**Project Report on**

***Hotel Reservation System***

***Introduction***

The project is based on on-site hotel reservation system. It allows guests and members of the application to place reservation(s), and the admins may approve or disapprove the bookings. The application is written on Python 3.7 and uses the Kivy module for Graphical User Interface.

This project is aimed towards hotels where reservation may be allowed by on-site displayed tablets or computers connected to the same host, i.e. the hotels reservation system server.

***Structure of UI***

The following table shows the flow of screens/pages for the application;

|  |  |  |  |
| --- | --- | --- | --- |
| Guest | Reservation form screen | | |
| Member | Login | Reservation form Screen | Delete my account screen |
| Admin | Login | Confirmation Screen | Booked rooms Screen |
| Checked-in rooms Screen |
| New Admin Screen |
| Create room Screen |
| Delete my account Screen |
| New | Member create Screen | | |
| Rooms | View rooms Screen | | |

On launch of the app, the users are prompted to make a selection whether they want to reserve a hotel room as a guest or member, or check for available room types, its costs, and availability of those rooms. The administrative of the reservation system can log in through here as well.

To check for available rooms, prices, room capacity, guests can click on the ‘Rooms’ button.

The rooms are arranged in ascending order of room number.

Guest members are able to place bookings directly after filing out the reservation forms. The emails and phone numbers entered are checked, and the check-in and check-out dates are checked whether they are in correct format or not. These dates cannot be the same and cannot be in the past.

Member account holders can log into their accounts and place reservations similarly, but most of their common data will be saved, so they can make reservations quickly. They also have the ability to delete their accounts themselves.

Hotel guests wanting to make their accounts are able to do so themselves by clicking of the ‘New’ button on the home page. They will get an unique Member ID and their input data will also be checked to make sure they are in correct format. They will be prompted to re-enter their passwords to make sure they are aware of their own password.

Admins can log into administration account by logging in thorough their unique Admin ID and password. The first screen displayed after logging in to the administration account is the reservation rooms which are yet to be processed. The admin can select multiple rows of data and either accept them or decline the reservations. The processed reservations are moved to the booked rooms screen, from where the admins are able to check-in the guests. After check-in, the data is moved to the checked-in screen, where the admins are able to check out the guests and the cost of the total period of stay is displayed beside each guest data. After checking out the guests, the data is deleted. Admins are also able to create new rooms, new admins and may also delete their admin account.

***Databases***

The Database Management System used is MySQL. It is used to store data of member information, admin information, all available rooms, requested rooms, booked rooms and checked in rooms.

The database name is “hotel\_management” and the following are the tables included in it :

* admin\_info – For all the admin information
* booked\_rooms – For all the rooms which have been verified by admin
* checked\_in – For all the rooms which have been checked in by the guest
* member\_info – For all the member information
* requested\_rooms – For all the rooms which have been requested by guest/member users
* rooms – For the rooms that are created by admin or are already existing

mysql> use hotel\_management;

Reading table information for completion of table and column names

You can turn off this feature to get a quicker startup with -A

Database changed

mysql> show tables;

+----------------------------+

| Tables\_in\_hotel\_management |

+----------------------------+

| admin\_info |

| booked\_rooms |

| checked\_in |

| member\_info |

| requested\_rooms |

| rooms |

+----------------------------+

6 rows in set (0.00 sec)

1. *admin\_info structure*

mysql> desc admin\_info;

+-------------------+-------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+-------------------+-------------+------+-----+---------+-------+

| admin\_id | varchar(10) | NO | PRI | NULL | |

| admin\_phonenumber | varchar(15) | NO | | NULL | |

| admin\_email | varchar(50) | NO | | NULL | |

| admin\_name | varchar(25) | NO | | NULL | |

| admin\_password | varchar(25) | NO | | NULL | |

+-------------------+-------------+------+-----+---------+-------+

5 rows in set (0.00 sec)

1. *booked\_rooms structure*

mysql> desc booked\_rooms;

+-------------+-------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+-------------+-------------+------+-----+---------+-------+

| room\_no | varchar(5) | NO | MUL | NULL | |

| id | varchar(10) | NO | MUL | NULL | |

| phonenumber | varchar(15) | NO | | NULL | |

| email | varchar(50) | NO | | NULL | |

| name | varchar(25) | NO | | NULL | |

| check\_in | date | NO | | NULL | |

| check\_out | date | NO | | NULL | |

+-------------+-------------+------+-----+---------+-------+

7 rows in set (0.00 sec)

1. *checked\_in structure*

mysql> desc checked\_in;

+------------+-------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+------------+-------------+------+-----+---------+-------+

| room\_no | varchar(5) | NO | | NULL | |

| id | varchar(10) | NO | | NULL | |

| name | varchar(25) | NO | | NULL | |

| checked\_in | date | NO | | NULL | |

| check\_out | date | NO | | NULL | |

+------------+-------------+------+-----+---------+-------+

5 rows in set (0.00 sec)

1. *member\_info structure*

mysql> desc member\_info;

+--------------------+-------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+--------------------+-------------+------+-----+---------+-------+

| member\_id | varchar(10) | NO | PRI | NULL | |

| member\_phonenumber | varchar(15) | NO | | NULL | |

| member\_email | varchar(50) | NO | | NULL | |

| member\_name | varchar(25) | NO | | NULL | |

| member\_password | varchar(25) | NO | | NULL | |

+--------------------+-------------+------+-----+---------+-------+

5 rows in set (0.00 sec)

1. *requested\_rooms structure*

mysql> desc requested\_rooms;

+-------------+-------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+-------------+-------------+------+-----+---------+-------+

| room\_type | varchar(25) | NO | | NULL | |

| id | varchar(10) | NO | | NULL | |

| phonenumber | varchar(15) | NO | | NULL | |

| name | varchar(25) | NO | | NULL | |

| no\_guests | int | NO | | NULL | |

| check\_in | date | NO | | NULL | |

| check\_out | date | NO | | NULL | |

+-------------+-------------+------+-----+---------+-------+

7 rows in set (0.00 sec)

1. *rooms structure*

mysql> desc rooms;

+-------------------+-------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+-------------------+-------------+------+-----+---------+-------+

| room\_no | varchar(5) | NO | PRI | NULL | |

| room\_type | varchar(25) | NO | | NULL | |

| room\_availability | char(1) | YES | | NULL | |

| room\_capacity | int | NO | | NULL | |

| room\_price | int | NO | | NULL | |

+-------------------+-------------+------+-----+---------+-------+

5 rows in set (0.00 sec)

***Functions in the program***

The home screen has three large-sized function, namely ‘Guest’ (to call the screen for room reservation as guest), ‘Member’ (to call the member login screen), ‘Admin’ (to login to the application as an administrative member). Following these functions are the two functions ‘New’ and ‘Rooms’ for loading the screen where new member accounts can be created and the screen which displays all the room details respectively.

1. *The guest screen*

The guest is prompted to fill out a form which inputs his/her name, phone number, choice of room type, number of guests, expected check-in and check-out date. The phone number is checked where it is in proper format (xxx-xxxx-xxxx) and must not include alphabets. The room type is selected through a drop-down list of all the available types of rooms and by default is set to ‘Select’. Similarly, the number of guests is selected through a drop-down list ranging from one to 10. The check-in and the check-out text inputs have the same structure, where the inputs are checked and must be in the correct format (yyyy-mm-dd).

If the form has any empty or invalid inputs, also including leaving the drop-down entry set to the default value, the data is not processed and a popup is displayed. If the form is filled out successfully, a popup is displayed stating that the reservation was sent for approval.

1. *The member login screen*

This screen has two input boxes for the ‘Member ID’ and ‘Password’ for the member. The id is checked whether it is existing in the database and the password to the corresponding id. If any input is left empty or the id and the password don’t match the database values, a popup is displayed stating login error.

After the member is through with the login phase, he/she is presented to a similar form from the guest reservation screen. The member is to select preferable room type and number of guests, and fill out the check-in and check-out dates. If the member successfully fills out the form, a popup is displayed stating that the request has been sent for Admin confirmation and is returned to the Home Screen.

1. *The admin login screen*

Similar to the member login screen, here the admin is prompted to fill out his/her ‘Admin ID’ and ‘Password’ for the admin. The inputs are checked in the database whether they are present and correspond to the id. If any input is left empty or the id and the password don’t match the database values, a popup is displayed stating login error.

The admin confirmation screen is displayed after the admin is through the login phase. The screen has all the requested rooms by guests and members, where the admin can select them by clicking on the check-boxes beside each row of request and either decline the reservations or accept them. Either the case, a popup is displayed stating the fore-mentioned reply for the request is successfully deleted or confirmed. If the request is accepted, the selected rows of requests are moved to booked\_rooms table of the database. The room number is selected by the program on its own and marked as availability equals to ‘N’ in the table Rooms of the database.

The admin confirmation screen has five other functions in it which can be called by clicking on the respective buttons which are displayed on the top of the screen.

* + 1. The booked rooms screen can be called through the button on the upper right hand corner. This screen is similar to the data presented in the admin confirmation screen but these are already processed/accepted. The admin can check-in each row or multiple rows of reserved rooms by checking the check-box beside each room and clicking on the ‘Checked In Now’ button on the bottom of the screen. When the guest checks into their room, the records are deleted from the booked\_rooms table and moved to the checked\_in table in the databases. And similarly moved from the booked rooms screen to the checked-in rooms screen.
    2. The checked-in rooms screen can be called by clicking on the button on the upper right corner, below the booked rooms button. The rows of data presented here are from the guests/members who have checked-in to their rooms. The total cost is displayed up-to date beside each row. The admin can check-out each row or multiple rows of checked-in rooms by checking the check-box beside each room and clicking on the ‘Checked Out Now’ button on the bottom of the screen. When the guest checks out of their room, the records are deleted from the checked\_in table and also removed from the current screen.
    3. The create new admin screen can be called from the button on the top of the current screen. This screen has a form to be filled out which is for the creation of the new admin account. If the required inputs are empty or invalid, an error is popped out, and if the details are correct the account is created. The new account is saved in the admin\_info table of the database.
    4. To access the screen for creating a new room, the admin can click on the button ‘Create room’ on the top of the admin confirmation screen. The screen is displayed and prompts to fill out the input boxes , ‘Room Number’, ‘Room Type’, ‘Room Capacity’ and ‘Room Price’. If the inputs are left empty, an error popup is displayed stating that the room could not be created. After successful room creation, the details are inserted into the rooms table of the database and the rooms screen is refreshed.
    5. For deleting the admin account, there is a button ‘Delete My Account’ on the top of the admin confirmation page. On pressing the button, a popup is displayed asking whether you want to delete the account or not. If the delete button is pressed, the screen is brought back to the Home screen and the admin account is deleted from the admin\_info table.

1. *Member account creation screen*

The ‘New’ screen has a button to call the Member account creation page. Similar to the admin create page, this page has text input boxes which must be filled for creating a member account. The inputs are checked, and if they are empty or invalid, a popup is displayed stating account creation failure. If the inputs are valid, a new member account is created in the member\_info table in the database and a popup is displayed stating account creation was successful. The home screen is called back.

1. *Rooms screen*

On pressing the rooms button on the lower right corner of the home screen, the rooms screen is displayed. All the rooms are displayed on a scroll view.

***Code***

import random

import mysql.connector as connector

from datetime import datetime, date

from kivy.app import App

from kivy.uix.gridlayout import GridLayout

from kivy.uix.floatlayout import FloatLayout

from kivy.uix.scrollview import ScrollView

from kivy.core.window import Window

from kivy.uix.textinput import TextInput

from kivy.uix.button import Button

from kivy.uix.label import Label

from kivy.graphics import Color, Rectangle

from kivy.uix.screenmanager import ScreenManager, Screen

from kivy.uix.popup import Popup

from kivy.uix.checkbox import CheckBox

from kivy.uix.dropdown import DropDown

"""

/usr/local/mysql/bin/mysql -u root -p

"""

# creating database if not exists

def mysql():

mycon = connector.connect(user='root', password='avishek2002')

cursor = mycon.cursor()

cursor.execute('show databases;')

databases = cursor.fetchall()

condition = ''

for database in databases:

if "hotel\_management" == database[0]:

condition = "Exists"

break

else:

condition = "Does not exist"

if condition != "Exists":

cursor.execute('create database if not exists hotel\_management;')

cursor.execute('use hotel\_management;')

cursor.execute('create table admin\_info(admin\_id varchar(10) primary key not null);')

cursor.execute('alter table admin\_info add(admin\_phonenumber varchar(15) not null);')

cursor.execute('alter table admin\_info add(admin\_email varchar(50) not null);')

cursor.execute('alter table admin\_info add(admin\_name varchar(25) not null);')

cursor.execute('alter table admin\_info add(admin\_password varchar(25) not null)')

cursor.execute('create table member\_info (member\_id varchar(10) primary key not null);')

cursor.execute('alter table member\_info add(member\_phonenumber varchar(15) not null);')

cursor.execute('alter table member\_info add(member\_email varchar(50) not null);')

cursor.execute('alter table member\_info add(member\_name varchar(25) not null);')

cursor.execute('alter table member\_info add(member\_password varchar(25) not null);')

cursor.execute('create table rooms (room\_no varchar(5) primary key not null);')

cursor.execute("alter table rooms add(room\_type varchar(25) not null);")

cursor.execute('alter table rooms add(room\_availability char(1));')

cursor.execute('alter table rooms add(room\_capacity int(2) not null);')

cursor.execute('alter table rooms add(room\_price int(5) not null);')

cursor.execute('create table booked\_rooms(room\_no varchar(5) not null);')

cursor.execute('alter table booked\_rooms add(foreign key(room\_no) references rooms(room\_no));')

cursor.execute('alter table booked\_rooms add(id varchar(10) not null);')

cursor.execute('alter table booked\_rooms add(foreign key(id) references member\_info(member\_id));')

cursor.execute('alter table booked\_rooms add(phonenumber varchar(15) not null);')

cursor.execute('alter table booked\_rooms add(email varchar(50) not null);')

cursor.execute('alter table booked\_rooms add(name varchar(25) not null);')

cursor.execute('alter table booked\_rooms add(check\_in date not null);')

cursor.execute('alter table booked\_rooms add(check\_out date not null);')

cursor.execute('create table requested\_rooms(room\_type varchar(25) not null);')

cursor.execute('alter table requested\_rooms add(id varchar(10) not null);')

cursor.execute('alter table requested\_rooms add(phonenumber varchar(15) not null);')

cursor.execute('alter table requested\_rooms add(name varchar(25) not null);')

cursor.execute('alter table requested\_rooms add(no\_guests int(2) not null);')

cursor.execute('alter table requested\_rooms add(check\_in date not null);')

cursor.execute('alter table requested\_rooms add(check\_out date not null);')

cursor.execute("create table checked\_in(room\_no varchar(5) not null);")

cursor.execute("alter table checked\_in add(id varchar(10) not null);")

cursor.execute("alter table checked\_in add(name varchar(25) not null);")

cursor.execute("alter table checked\_in add(checked\_in date not null);")

cursor.execute("alter table checked\_in add(check\_out date not null);")

mycon.commit()

mycon.close()

else:

return

mysql()

mycon = connector.connect(user='root', password='avishek2002', database='hotel\_management')

cursor = mycon.cursor()

# getting available rooms

def available\_rooms():

cursor.execute("select \* from rooms;")

rooms = cursor.fetchall()

return rooms

# getting requested rooms

def requested\_rooms():

cursor.execute("select \* from requested\_rooms;")

rooms = cursor.fetchall()

return rooms

# getting booked rooms

def booked\_rooms():

cursor.execute("select \* from booked\_rooms;")

rooms = cursor.fetchall()

return rooms

# getting checked in rooms

def checked\_rooms():

cursor.execute("select \* from checked\_in;")

rooms = cursor.fetchall()

return rooms

# getting room type

def room\_types():

cursor.execute("select distinct(room\_type) from rooms;")

room\_type = cursor.fetchall()

return room\_type

# getting admin information

def admin\_info():

cursor.execute("select \* from admin\_info;")

data = cursor.fetchall()

return data

# getting member information

def member\_info():

cursor.execute("select \* from member\_info;")

data = cursor.fetchall()

return data

# incorrect login details

def popup\_incorrect():

# incorrect login details screen

class popup\_incorrect\_screen(FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(popup\_incorrect\_screen, self).\_\_init\_\_(\*\*kwargs)

self.label = Label(text="ID or Password is incorrect!", font\_size=50, color=(1, 0, 0, 1),

size\_hint=(.4, .2), pos\_hint={"x": 0.3, "top": .75}, text\_size=(600, None),

halign="center")

self.add\_widget(self.label)

self.back = Button(text="Try Again", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1),

size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": 0.3})

self.back.bind(on\_release=self.call\_back)

self.add\_widget(self.back)

def call\_back(self, instances):

popupWindow.dismiss()

pass

show = popup\_incorrect\_screen()

popupWindow = Popup(title="Login Error", content=show, size\_hint=(None, None), size=(650, 600),

pos=(0, 0), auto\_dismiss=False)

popupWindow.open()

# invalid room reservation submission

def popup\_invalid\_submission():

# invalid room reservation submission screen

class popup\_invalid\_screen(FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(popup\_invalid\_screen, self).\_\_init\_\_(\*\*kwargs)

self.label = Label(text="The text boxes must not be left empty and must be in correct format!",

font\_size=50, color=(1, 0, 0, 1), size\_hint=(.4, .2), pos\_hint={"x": 0.3, "top": .75},

text\_size=(600, None), halign="center")

self.add\_widget(self.label)

self.back = Button(text="Re-try submitting form", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1),

size\_hint=(.42, .15), pos\_hint={"x": 0.3, "top": 0.3})

self.back.bind(on\_release=self.call\_back)

self.add\_widget(self.back)

def call\_back(self, instances):

invalidWindow.dismiss()

pass

show = popup\_invalid\_screen()

invalidWindow = Popup(title="Submission Error", content=show, size\_hint=(None, None), size=(650, 600),

pos=(0, 0), auto\_dismiss=False)

invalidWindow.open()

# valid room reservation submission

def popup\_valid\_submission():

# valid room reservation submission screen

class popup\_valid\_screen(FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(popup\_valid\_screen, self).\_\_init\_\_(\*\*kwargs)

self.label = Label(text="Successfully sent request for room reservation!", text\_size=(600, None),

halign="center", font\_size=50, color=(1, 0, 0, 1), size\_hint=(.4, .2),

pos\_hint={"x": 0.3, "top": .75})

self.add\_widget(self.label)

self.back = Button(text="Return to Home Screen", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1),

size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": 0.3})

self.back.bind(on\_release=self.call\_back)

self.add\_widget(self.back)

def call\_back(self, instances):

validWindow.dismiss()

sm.transition.direction = 'right'

sm.current = 'Home'

pass

show = popup\_valid\_screen()

validWindow = Popup(title="Submission success", content=show, size\_hint=(None, None), size=(650, 600),

pos=(0, 0), auto\_dismiss=False)

validWindow.open()

# confirm reservations as Admin

def popup\_confirm():

# confirm reservation screen

class popup\_confirm\_screen(FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(popup\_confirm\_screen, self).\_\_init\_\_(\*\*kwargs)

self.label = Label(text="Confirmed reservations for the selected rooms!", text\_size=(600, None),

halign="center", font\_size=50, color=(1, 0, 0, 1), size\_hint=(.4, .2),

pos\_hint={"x": 0.3, "top": .75})

self.add\_widget(self.label)

self.back = Button(text="Return to your Page", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1),

size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": 0.3})

self.back.bind(on\_release=self.call\_back)

self.add\_widget(self.back)

def call\_back(self, instances):

confirmWindow.dismiss()

pass

show = popup\_confirm\_screen()

confirmWindow = Popup(title="Confirmation success", content=show, size\_hint=(None, None), size=(650, 600),

pos=(0, 0), auto\_dismiss=False)

confirmWindow.open()

# deleteing reservations as Admin

def popup\_confirm\_delete():

# delete reservation screen

class popup\_confirm\_screen(FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(popup\_confirm\_screen, self).\_\_init\_\_(\*\*kwargs)

self.label = Label(text="Deleted reservations for the selected rooms!", text\_size=(600, None),

halign="center", font\_size=50, color=(1, 0, 0, 1), size\_hint=(.4, .2),

pos\_hint={"x": 0.3, "top": .75})

self.add\_widget(self.label)

self.back = Button(text="Return to your Page", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1),

size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": 0.3})

self.back.bind(on\_release=self.call\_back)

self.add\_widget(self.back)

def call\_back(self, instances):

confirmWindow.dismiss()

pass

show = popup\_confirm\_screen()

confirmWindow = Popup(title="Deletion success", content=show, size\_hint=(None, None), size=(650, 600),

pos=(0, 0), auto\_dismiss=False)

confirmWindow.open()

# generating admin id

def makeid():

cursor.execute('select admin\_id from admin\_info;')

data = cursor.fetchall()

if data != "":

ID = 'A'

while True:

for i in range(0, 3):

ID += str(random.randrange(0, 10))

if ID not in data:

break

else:

continue

else:

ID = "A001"

return ID

# creating admin acoount

def create\_admin(id\_, phone, email, name, password):

cursor.execute("insert into admin\_info(admin\_id,admin\_phonenumber,admin\_email,admin\_name,admin\_password) "

"values('{}','{}','{}','{}','{}')".format(id\_, phone, email, name, password))

mycon.commit()

return

# deleting admin account

def adminaccountdelete(admin\_identifier):

cursor.execute("delete from admin\_info where admin\_id = '{}'".format(admin\_identifier))

mycon.commit()

return

# generating member id

def makememberid():

cursor.execute('select member\_id from member\_info;')

data = cursor.fetchall()

if data != "":

ID = 'M'

while True:

for i in range(0,3):

ID += str(random.randrange(0,10))

if ID not in data:

break

else:

continue

else:

ID = "M001"

return ID

# creating member account

def create\_member(ID, phone\_number, email, name, password):

cursor.execute("insert into member\_info(member\_id,member\_phonenumber,member\_email,member\_name,"

"member\_password) values('{}','{}','{}','{}','{}')"

.format(ID, phone\_number, email, name, password))

mycon.commit()

return

# deleting member account and its data

def member\_delete(ID):

cursor.execute("set foreign\_key\_checks = 0")

cursor.execute("delete from member\_info where member\_id='{}'".format(ID))

cursor.execute("delete from requested\_rooms where id = '{}'".format(ID))

cursor.execute("delete from booked\_rooms where id = '{}'".format(ID))

cursor.execute("set foreign\_key\_checks = 1")

mycon.commit()

return

# account created

def popup\_account\_created():

# account created screen

class account\_created\_screen(FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(account\_created\_screen, self).\_\_init\_\_(\*\*kwargs)

self.label = Label(text="Your Account has been successfully created!", text\_size=(600, None),

halign="center", font\_size=50, color=(1, 0, 0, 1), size\_hint=(.4, .2),

pos\_hint={"x": 0.3, "top": .75})

self.add\_widget(self.label)

self.back = Button(text="Home", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1),

size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": 0.3})

self.back.bind(on\_release=self.call\_back)

self.add\_widget(self.back)

def call\_back(self, instances):

accountWindow.dismiss()

sm.transition.direction = 'right'

sm.current = 'Home'

pass

show = account\_created\_screen()

accountWindow = Popup(title="Account Created", content=show, size\_hint=(None, None), size=(650, 600),

pos=(0, 0), auto\_dismiss=False)

accountWindow.open()

# account creation failure

def popup\_account\_notcreated():

# account creation failure screen

class account\_notcreated\_screen(FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(account\_notcreated\_screen, self).\_\_init\_\_(\*\*kwargs)

self.label = Label(text="Your Account could not be processed!", text\_size=(600, None), halign="center",

font\_size=50, color=(1, 0, 0, 1), size\_hint=(.4, .2), pos\_hint={"x": 0.3, "top": .75})

self.add\_widget(self.label)

self.back = Button(text="Try Again", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1),

size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": 0.3})

self.back.bind(on\_release=self.call\_back)

self.add\_widget(self.back)

def call\_back(self, instances):

failureWindow.dismiss()

pass

show = account\_notcreated\_screen()

failureWindow = Popup(title="ERROR", content=show, size\_hint=(None, None), size=(650, 600),

pos=(0, 0), auto\_dismiss=False)

failureWindow.open()

# member account delete

def popup\_member\_delete():

# member account delete screen

class member\_delete\_screen(FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(member\_delete\_screen, self).\_\_init\_\_(\*\*kwargs)

self.label = Label(text="Your Account has been deleted!", text\_size=(600, None), halign="center",

font\_size=50, color=(1, 0, 0, 1), size\_hint=(.4, .2), pos\_hint={"x": 0.3, "top": .75})

self.add\_widget(self.label)

self.back = Button(text="Back", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1),

size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": 0.3})

self.back.bind(on\_release=self.call\_back)

self.add\_widget(self.back)

def call\_back(self, instances):

memberdeleteWindow.dismiss()

sm.transition.direction = 'right'

sm.current = 'Home'

pass

show = member\_delete\_screen()

memberdeleteWindow = Popup(title="DELETED", content=show, size\_hint=(None, None), size=(650, 600),

pos=(0, 0), auto\_dismiss=False)

memberdeleteWindow.open()

# member account delete failure

def popup\_member\_delete\_failure():

# member account delete failure screen

class member\_delete\_failure\_screen(FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(member\_delete\_failure\_screen, self).\_\_init\_\_(\*\*kwargs)

self.label = Label(text="Your Account could not deleted!", text\_size=(600, None), halign="center",

font\_size=50, color=(1, 0, 0, 1), size\_hint=(.4, .2), pos\_hint={"x": 0.3, "top": .75})

self.add\_widget(self.label)

self.back = Button(text="Retry", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1),

size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": 0.3})

self.back.bind(on\_release=self.call\_back)

self.add\_widget(self.back)

def call\_back(self, instances):

memberdeletefailWindow.dismiss()

pass

show = member\_delete\_failure\_screen()

memberdeletefailWindow = Popup(title="ERROR", content=show, size\_hint=(None, None), size=(650, 600),

pos=(0, 0), auto\_dismiss=False)

memberdeletefailWindow.open()

# admin account delete cornfirmation

def popup\_confirm\_admindelete():

# admin account delete confirmation screen

class admin\_delete\_confirmation\_screen(FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(admin\_delete\_confirmation\_screen, self).\_\_init\_\_(\*\*kwargs)

self.label = Label(text="Are you sure you want to delete this Admin Account!", text\_size=(600, None),

halign="center", font\_size=50, color=(1, 0, 0, 1), size\_hint=(.4, .2),

pos\_hint={"x": 0.3, "top": .75})

self.add\_widget(self.label)

self.back = Button(text="Cancel", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.4, .15), pos\_hint={"x": 0.1, "top": 0.3})

self.back.bind(on\_release=self.call\_back)

self.add\_widget(self.back)

self.delete = Button(text="Delete", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.4, .15), pos\_hint={"x": 0.5, "top": 0.3})

self.delete.bind(on\_release=self.call\_delete)

self.add\_widget(self.delete)

def call\_back(self, instances):

admindeleteconfirmationWindow.dismiss()

def call\_delete(self, instances):

global admin\_identifier

adminaccountdelete(admin\_identifier)

admindeleteconfirmationWindow.dismiss()

sm.transition.direction = "right"

sm.current = "Home"

pass

show = admin\_delete\_confirmation\_screen()

admindeleteconfirmationWindow = Popup(title="Confirm", content=show, size\_hint=(None, None), size=(650, 600),

pos=(0, 0), auto\_dismiss=False)

admindeleteconfirmationWindow.open()

# creating room details

def create\_room(roomno, roomcap, roomtype, roomprice):

roomavailability = "Y"

cursor.execute("insert into rooms(room\_no,room\_type,room\_availability,room\_capacity,room\_price)"

"values('{}','{}','{}','{}','{}')".format(roomno, roomtype, roomavailability, roomcap, roomprice))

mycon.commit()

return

# room created

def popup\_room\_created():

# room created screen

class room\_created\_screen(FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(room\_created\_screen, self).\_\_init\_\_(\*\*kwargs)

self.label = Label(text="Room Created!", halign="center", text\_size=(600, None),

font\_size=50, color=(1, 0, 0, 1), size\_hint=(.4, .2), pos\_hint={"x": 0.3, "top": .75})

self.add\_widget(self.label)

self.back = Button(text="Back", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1),

size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": 0.3})

self.back.bind(on\_release=self.call\_back)

self.add\_widget(self.back)

def call\_back(self, instances):

roomWindow.dismiss()

sm.transition.direction = 'right'

sm.current = 'AdminConfirmation'

pass

show = room\_created\_screen()

roomWindow = Popup(title="Success", content=show, size\_hint=(None, None), size=(650, 600),

pos=(0, 0), auto\_dismiss=False)

roomWindow.open()

# room creation failure

def popup\_room\_notcreated():

# room creation failure screen

class room\_notcreated\_screen(FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(room\_notcreated\_screen, self).\_\_init\_\_(\*\*kwargs)

self.label = Label(text="Room could not be created!", halign="center", text\_size=(600, None),

font\_size=50, color=(1, 0, 0, 1), size\_hint=(.4, .2), pos\_hint={"x": 0.3, "top": .75})

self.add\_widget(self.label)

self.back = Button(text="Try Again", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1),

size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": 0.3})

self.back.bind(on\_release=self.call\_back)

self.add\_widget(self.back)

def call\_back(self, instances):

failureWindow.dismiss()

pass

show = room\_notcreated\_screen()

failureWindow = Popup(title="ERROR", content=show, size\_hint=(None, None), size=(650, 600),

pos=(0, 0), auto\_dismiss=False)

failureWindow.open()

# inserting value into requested\_rooms as guest

def request\_room(name, phone, room\_type, noofquests, check\_in, check\_out):

cursor.execute("insert into requested\_rooms(room\_type,id,phonenumber,name,no\_guests,check\_in,check\_out)"

"values('{}','{}','{}','{}',{},'{}','{}')"

.format(room\_type, 'GUEST', phone, name, noofquests, check\_in, check\_out))

mycon.commit()

return

# inserting value into requested\_rooms as member

def request\_room\_asmember(ID, room\_type, noofquests, check\_in, check\_out):

cursor.execute("select \* from member\_info where member\_id = '{}';".format(ID))

data = cursor.fetchall()

phone = data[0][1]

name = data[0][3]

cursor.execute("insert into requested\_rooms(room\_type,id,phonenumber,name,no\_guests,check\_in,check\_out)"

"values('{}','{}','{}','{}',{},'{}','{}')"

.format(room\_type, ID, phone, name, noofquests, check\_in, check\_out))

mycon.commit()

return

# moving requested\_room to booked\_rooms

def move\_to\_booked\_rooms(idx):

name = idx

cursor.execute("set foreign\_key\_checks = 0")

cursor.execute("select \* from requested\_rooms where name='{}'".format(name))

data = cursor.fetchone()

if data[1] != "GUEST":

cursor.execute("select \* from member\_info where member\_name='{}'".format(name))

more\_data = cursor.fetchone()

id\_ = data[1]

phone = data[2]

email = more\_data[2]

name = name

check\_in = data[5]

check\_out = data[6]

room\_type = data[0]

# checking if there is free room for the above room type

cursor.execute("select room\_no,room\_availability from rooms where room\_type='{}'".format(room\_type))

rooms\_data = cursor.fetchall()

for room\_data in rooms\_data:

room\_no = room\_data[0]

room\_availability = room\_data[1]

if room\_availability == 'Y':

cursor.execute("insert into booked\_rooms(room\_no,id,phonenumber,email,name,check\_in,check\_out)"

"values('{}','{}','{}','{}','{}','{}','{}')"

.format(room\_no, id\_, phone, email, name, check\_in, check\_out))

cursor.execute("alter table booked\_rooms order by check\_in")

cursor.execute("delete from requested\_rooms where name='{}'".format(name))

cursor.execute("update rooms set room\_availability='N' where room\_no='{}'".format(room\_no))

mycon.commit()

break

else:

continue

else:

print("NO ROOM")

elif data[1] == "GUEST":

id\_ = data[1]

phone = data[2]

name = name

check\_in = data[5]

check\_out = data[6]

room\_type = data[0]

email = "NULL"

# checking if there is free room for the above room type

cursor.execute("select room\_no,room\_availability from rooms where room\_type='{}'".format(room\_type))

rooms\_data = cursor.fetchall()

for room\_data in rooms\_data:

room\_no = room\_data[0]

room\_availability = room\_data[1]

if room\_availability == 'Y':

cursor.execute("insert into booked\_rooms(room\_no,id,phonenumber,email,name,check\_in,check\_out)"

"values('{}','{}','{}','{}','{}','{}','{}')"

.format(room\_no, id\_, phone, email, name, check\_in, check\_out))

cursor.execute("alter table booked\_rooms order by check\_in")

cursor.execute("delete from requested\_rooms where name='{}'".format(name))

cursor.execute("update rooms set room\_availability='N' where room\_no='{}'".format(room\_no))

mycon.commit()

break

else:

continue

else:

print("NO ROOM")

cursor.execute("set foreign\_key\_checks = 1")

return

# deleting requests from requested\_rooms

def delete\_requestedroom(idx):

name = idx

cursor.execute("delete from requested\_rooms where name='{}'".format(name))

mycon.commit()

return

# moving booked\_rooms to checked\_in

def checked\_in\_rooms(idx):

name = idx

cursor.execute("select \* from booked\_rooms where name = '{}'".format(name))

data = cursor.fetchone()

room\_no = data[0]

ID = data[1]

checked\_in = datetime.today().strftime('%Y-%m-%d')

check\_out = data[6]

cursor.execute("insert into checked\_in(room\_no,id,name,checked\_in,check\_out)"

"values('{}','{}','{}','{}','{}')".format(room\_no, ID, name, checked\_in, check\_out))

cursor.execute("delete from booked\_rooms where name = '{}'".format(name))

mycon.commit()

return

# checking out guests and opening the room

def check\_out(idx):

name = idx

cursor.execute("select \* from checked\_in where name = '{}'".format(name))

data = cursor.fetchone()

room\_no = data[0]

cursor.execute("delete from checked\_in where name = '{}'".format(name))

cursor.execute("update rooms set room\_availability = 'Y' where room\_no = '{}'".format(room\_no))

mycon.commit()

return

# Home page

class Home(Screen, FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(Home, self).\_\_init\_\_(\*\*kwargs)

with self.canvas:

Color(1, 1, 0, 1)

self.rect = Rectangle(size=(3000, 2000))

self.label = Label(text="Select an option: ", font\_size=75, color=(225, 225, 225, 1), size\_hint=(.4, .3),

pos\_hint={"x": 0.31, "top": 1})

self.add\_widget(self.label)

self.guest = Button(text="Guest", font\_size=40, background\_color=(0, 225, 225, 1), color=(225, 225, 225, 1),

size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": .7})

self.guest.bind(on\_release=self.call\_guest)

self.add\_widget(self.guest)

self.member = Button(text="Member", font\_size=40, background\_color=(0, 225, 225, 1), color=(225, 225, 225, 1),

size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": 0.5})

self.member.bind(on\_release=self.call\_member)

self.add\_widget(self.member)

self.admin = Button(text="Admin", font\_size=40, background\_color=(0, 225, 225, 1), color=(225, 225, 225, 1),

size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": 0.3})

self.admin.bind(on\_release=self.call\_admin)

self.add\_widget(self.admin)

self.create = Button(text='NEW', font\_size=30, background\_color=(0, 225, 225, 1), color=(225, 225, 225, 1),

size\_hint=(.15, .1), pos\_hint={"x": 0.05, "top": 0.15})

self.create.bind(on\_release=self.call\_create)

self.add\_widget(self.create)

self.rooms = Button(text='ROOMS', font\_size=30, background\_color=(0, 225, 225, 1), color=(225, 225, 225, 1),

size\_hint=(.15, .1), pos\_hint={"x": 0.8, "top": .15})

self.rooms.bind(on\_release=self.call\_rooms)

self.add\_widget(self.rooms)

def call\_guest(self, instances):

sm.transition.direction = 'left'

sm.current = 'GuestReservation'

def call\_member(self, instances):

sm.transition.direction = 'left'

sm.current = 'MemberLogin'

def call\_admin(self, instances):

sm.transition.direction = 'left'

sm.current = 'AdminLogin'

def call\_create(self, instances):

sm.transition.direction = 'up'

sm.current = 'Create'

def call\_rooms(self, instances):

sm.transition.direction = 'up'

sm.current = 'Rooms'

pass

# Create account screen (middle screen for Member signup screen)

class Create(Screen, FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(Create, self).\_\_init\_\_(\*\*kwargs)

with self.canvas:

Color(1, 1, 0, 1)

self.rect = Rectangle(size=(3000, 2000))

self.label = Label(text="Account Creation Page", font\_size=75, color=(225, 225, 225, 1), size\_hint=(.4, .3),

pos\_hint={"x": 0.31, "top": 1})

self.add\_widget(self.label)

self.createmember = Button(text="Become a member", font\_size=40, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": 0.55})

self.createmember.bind(on\_release=self.call\_createmember)

self.add\_widget(self.createmember)

self.home = Button(text="Home", font\_size=25, background\_color=(0, 225, 225, 1), color=(225, 225, 225, 1),

size\_hint=(.15, .025), pos\_hint={"x": 0.01, "top": .99})

self.home.bind(on\_release=self.call\_home)

self.add\_widget(self.home)

def call\_home(self, instances):

sm.transition.direction = 'down'

sm.current = 'Home'

def call\_createmember(self, instances):

sm.transition.direction = 'left'

sm.current = 'Createmember'

pass

# Rooms screen

class Rooms(Screen, FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(Rooms, self).\_\_init\_\_(\*\*kwargs)

with self.canvas:

Color(1, 1, 0, 1)

self.rect = Rectangle(size=(3000, 2000))

self.label = Label(text="Room Selection", font\_size=75, color=(225, 225, 225, 1), size\_hint=(.4, .3),

pos\_hint={"x": 0.31, "top": 1})

self.add\_widget(self.label)

x = [0.2, 0.35, 0.5, 0.65, 0.8]

top = 0.75

header\_string = ['Room No.', 'Room Type', 'Available', 'Capacity', 'Price(¥)']

for i in range(0, len(header\_string)):

self.header = Label(text=header\_string[i], font\_size=40, color=(225, 225, 225, 1), size\_hint=(0, 0),

pos\_hint={'x': x[i], 'top': top})

self.add\_widget(self.header)

self.layout = GridLayout(cols=5, spacing=40, size\_hint\_y=None)

self.layout.bind(minimum\_height=self.layout.setter('height'))

rooms = available\_rooms()

x = [0.2, 0.35, 0.5, 0.65, 0.8]

top = 0.8

def lining():

for i in range(0, 5):

self.room = Label(text="", font\_size=25, color=(225, 225, 225, 1),

pos\_hint={'x': x[i], 'top': top})

self.layout.add\_widget(self.room)

return

lining()

for room in rooms:

for i in range(0, len(room)):

self.room = Label(text=str(room[i]), font\_size=25, color=(225, 225, 225, 1),

pos\_hint={'x': x[i], 'top': top})

self.layout.add\_widget(self.room)

top -= .05

lining()

self.root = ScrollView(size\_hint=(None, None), size=(Window.width \* .75, Window.height \* .6),

pos\_hint={"x": 0.125, 'top': 0.675})

self.root.add\_widget(self.layout)

self.add\_widget(self.root)

self.home = Button(text="Home", font\_size=25, background\_color=(0, 225, 225, 1), color=(225, 225, 225, 1),

size\_hint=(.15, .025), pos\_hint={"x": 0.01, "top": .99})

self.home.bind(on\_release=self.call\_home)

self.add\_widget(self.home)

def call\_home(self, instances):

sm.transition.direction = 'down'

sm.current = 'Home'

pass

# Booked rooms screen

class BookedRooms(Screen, FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(BookedRooms, self).\_\_init\_\_(\*\*kwargs)

with self.canvas:

Color(1, 1, 0, 1)

self.rect = Rectangle(size=(3000, 2000))

self.label = Label(text="Booked Rooms", font\_size=75, color=(225, 225, 225, 1), size\_hint=(.4, .3),

pos\_hint={"x": 0.31, "top": 1})

self.add\_widget(self.label)

x = [0.1, 0.175, 0.3, 0.45, 0.6, 0.75, 0.9]

top = 0.75

header\_string = ['Room No.', 'ID', 'Phone Number', 'Email', 'Name', 'Check-in', 'Check-out']

for i in range(0, len(header\_string)):

self.header = Label(text=header\_string[i], font\_size=40, color=(225, 225, 225, 1), size\_hint=(0, 0),

pos\_hint={'x': x[i], 'top': top})

self.add\_widget(self.header)

rooms = booked\_rooms()

x = [0.1, 0.175, 0.3, 0.45, 0.6, 0.75, 0.9, 0.8]

top = 0.7

self.checkref = {}

for room in rooms:

for i in range(0, len(room)):

self.room = Label(text=str(room[i]), font\_size=25, color=(225, 225, 225, 1), size\_hint=(.0, .0),

pos\_hint={'x': x[i], 'top': top})

self.add\_widget(self.room)

self.check = CheckBox(size\_hint=(.02, .02), pos\_hint={'x': x[7], 'top': top + 0.01}, active=False)

self.add\_widget(self.check)

self.checkref[room[4]] = self.check

top -= 0.05

self.home = Button(text="Back", font\_size=25, background\_color=(0, 225, 225, 1), color=(225, 225, 225, 1),

size\_hint=(.15, .025), pos\_hint={"x": 0.01, "top": .99})

self.home.bind(on\_release=self.call\_home)

self.add\_widget(self.home)

self.submit = Button(text="Checked In Now", font\_size=40, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.4, .1), pos\_hint={"x": 0.3, "top": 0.125})

self.submit.bind(on\_release=self.call\_confirm)

self.add\_widget(self.submit)

def call\_home(self, instances):

sm.transition.direction = 'right'

sm.current = 'AdminConfirmation'

def call\_confirm(self, instances):

for idx, wgt in self.checkref.items():

if wgt.active:

checked\_in\_rooms(idx)

self.parent.get\_screen('BookedRooms').\_\_init\_\_()

self.parent.get\_screen('CheckedInRooms').\_\_init\_\_()

pass

# CheckedIn rooms screen

class CheckedInRooms(Screen, FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(CheckedInRooms, self).\_\_init\_\_(\*\*kwargs)

with self.canvas:

Color(1, 1, 0, 1)

self.rect = Rectangle(size=(3000, 2000))

self.label = Label(text="Checked-In Rooms", font\_size=75, color=(225, 225, 225, 1), size\_hint=(.4, .3),

pos\_hint={"x": 0.31, "top": 1})

self.add\_widget(self.label)

x = [0.1, 0.25, 0.4, 0.55, 0.7, 0.9]

top = 0.75

header\_string = ['Room No.', 'ID', 'Name', 'Checked-in', 'Check-out', 'Current Cost(¥)']

for i in range(0, len(header\_string)):

self.header = Label(text=header\_string[i], font\_size=40, color=(225, 225, 225, 1), size\_hint=(0, 0),

pos\_hint={'x': x[i], 'top': top})

self.add\_widget(self.header)

rooms = checked\_rooms()

x = [0.1, 0.25, 0.4, 0.55, 0.7, 0.8]

top = 0.7

self.checkref = {}

for room in rooms:

for i in range(0, len(room)):

self.room = Label(text=str(room[i]), font\_size=25, color=(225, 225, 225, 1), size\_hint=(.0, .0),

pos\_hint={'x': x[i], 'top': top})

self.add\_widget(self.room)

self.check = CheckBox(size\_hint=(.02, .02), pos\_hint={'x': x[5], 'top': top + 0.01}, active=False)

self.add\_widget(self.check)

self.checkref[room[2]] = self.check

cursor.execute("select room\_price from rooms where room\_no='{}'".format(room[0]))

price = cursor.fetchone()

days\_stayed = (date.today() - room[3]).days

curr\_cost = days\_stayed \* price[0]

self.cost = Label(text=str(curr\_cost), font\_size=25, color=(225, 225, 225, 1), size\_hint=(.0, .0),

pos\_hint={'x': 0.9, 'top': top})

self.add\_widget(self.cost)

top -= 0.05

self.home = Button(text="Back", font\_size=25, background\_color=(0, 225, 225, 1), color=(225, 225, 225, 1),

size\_hint=(.15, .025), pos\_hint={"x": 0.01, "top": .99})

self.home.bind(on\_release=self.call\_home)

self.add\_widget(self.home)

self.refresh = Button(text="Refresh Page", font\_size=35, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.2, .05), pos\_hint={"x": 0.76, "top": .1})

self.refresh.bind(on\_release=self.call\_refresh)

self.add\_widget(self.refresh)

self.submit = Button(text="Checked Out Now", font\_size=40, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.4, .1), pos\_hint={"x": 0.3, "top": 0.125})

self.submit.bind(on\_release=self.call\_confirm)

self.add\_widget(self.submit)

def call\_home(self, instances):

sm.transition.direction = 'right'

sm.current = 'AdminConfirmation'

def call\_refresh(self, instances):

self.parent.get\_screen('CheckedInRooms').\_\_init\_\_()

def call\_confirm(self, instances):

for idx, wgt in self.checkref.items():

if wgt.active:

check\_out(idx)

self.parent.get\_screen('CheckedInRooms').\_\_init\_\_()

self.parent.get\_screen('Rooms').\_\_init\_\_()

pass

# Guest reservation screen

class GuestReservation(Screen, FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(GuestReservation, self).\_\_init\_\_(\*\*kwargs)

with self.canvas:

Color(1, 1, 0, 1)

self.rect = Rectangle(size=(3000, 2000))

self.label = Label(text="Guest Reservation", font\_size=75, color=(225, 225, 225, 1), size\_hint=(.4, .3),

pos\_hint={"x": 0.31, "top": 1})

self.add\_widget(self.label)

self.name\_label = Label(text="Name", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.8})

self.add\_widget(self.name\_label)

self.nameofguest = TextInput(multiline=False, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.8})

self.add\_widget(self.nameofguest)

self.phone\_label = Label(text="Phone Number", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.7})

self.add\_widget(self.phone\_label)

self.phoneofguest = TextInput(multiline=False, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.7})

self.add\_widget(self.phoneofguest)

room\_type = room\_types()

self.room\_label = Label(text="Room Type", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.6})

self.add\_widget(self.room\_label)

self.dropdown = DropDown()

for room\_type\_ in room\_type:

type = room\_type\_[0]

self.val = Button(text=type, color=(225, 225, 1, 1), background\_color=(0, 225, 225, 1),

size\_hint\_y=None, height=44)

self.val.bind(on\_release=lambda val: self.dropdown.select(val.text))

self.dropdown.add\_widget(self.val)

self.roomofguest = Button(text='Select', color=(225, 225, 1, 1), background\_color=(0, 225, 225, 1),

size\_hint=(.35, .05), pos\_hint={"x": 0.5, "top": 0.6})

self.roomofguest.bind(on\_release=self.dropdown.open)

self.add\_widget(self.roomofguest)

self.dropdown.bind(on\_select=lambda instance, x: setattr(self.roomofguest, 'text', x))

self.noofguests\_label = Label(text="Number of Guests", font\_size=40, color=(225, 225, 1, 1),

size\_hint=(.35, .05), pos\_hint={"x": 0.15, "top": 0.5})

self.add\_widget(self.noofguests\_label)

self.dropdown2 = DropDown()

for num in range(1, 11):

self.val = Button(text=str(num), color=(225, 225, 1, 1), background\_color=(0, 225, 225, 1),

size\_hint\_y=None, height=44)

self.val.bind(on\_release=lambda val: self.dropdown2.select(val.text))

self.dropdown2.add\_widget(self.val)

self.noofguest = Button(text='Select', color=(225, 225, 1, 1), background\_color=(0, 225, 225, 1),

size\_hint=(.35, .05), pos\_hint={"x": 0.5, "top": 0.5})

self.noofguest.bind(on\_release=self.dropdown2.open)

self.add\_widget(self.noofguest)

self.dropdown2.bind(on\_select=lambda instance, x: setattr(self.noofguest, 'text', x))

self.checkin\_label = Label(text="Check-in Date", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.4})

self.add\_widget(self.checkin\_label)

self.checkinofguest = TextInput(multiline=False, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.4})

self.add\_widget(self.checkinofguest)

self.checkout\_label = Label(text="Check-out Date", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.3})

self.add\_widget(self.checkout\_label)

self.checkoutofguest = TextInput(multiline=False, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.3})

self.add\_widget(self.checkoutofguest)

self.home = Button(text="Home", font\_size=25, background\_color=(0, 225, 225, 1), color=(225, 225, 225, 1),

size\_hint=(.15, .025), pos\_hint={"x": 0.01, "top": 0.99})

self.home.bind(on\_release=self.call\_home)

self.add\_widget(self.home)

self.submit = Button(text="Submit form as Guest", font\_size=40, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": 0.2})

self.submit.bind(on\_release=self.guest\_submit)

self.add\_widget(self.submit)

def call\_home(self, instances):

self.nameofguest.text = ""

self.phoneofguest.text = ""

self.roomofguest.text = "Select"

self.noofguest.text = "Select"

self.checkinofguest.text = ""

self.checkoutofguest.text = ""

sm.transition.direction = 'right'

sm.current = 'Home'

def guest\_submit(self, instances):

name = self.nameofguest.text

phone = self.phoneofguest.text

room\_type = self.roomofguest.text

noofquests = self.noofguest.text

check\_in = self.checkinofguest.text

check\_out = self.checkoutofguest.text

phoneCheck = ''

for i in range(0, len(phone)):

if phone[i] == "-":

phoneCheck = phoneCheck

else:

phoneCheck += phone[i]

continue

today = date.today()

if name != "" and phoneCheck.isnumeric() and room\_type != "Select" and noofquests != "Select" and\

check\_in > str(today) and check\_out > str(today) and check\_out > check\_in:

request\_room(name, phone, room\_type, noofquests, check\_in, check\_out)

self.parent.get\_screen('AdminConfirmation').\_\_init\_\_()

popup\_valid\_submission()

else:

popup\_invalid\_submission()

self.nameofguest.text = ""

self.phoneofguest.text = ""

self.roomofguest.text = "Select"

self.noofguest.text = "Select"

self.checkinofguest.text = ""

self.checkoutofguest.text = ""

pass

# Member login screen

class MemberLogin(Screen, FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(MemberLogin, self).\_\_init\_\_(\*\*kwargs)

with self.canvas:

Color(1, 1, 0, 1)

self.rect = Rectangle(size=(3000, 2000))

self.label = Label(text="Member Login", font\_size=75, color=(225, 225, 225, 1), size\_hint=(.4, .3),

pos\_hint={"x": 0.31, "top": 1})

self.add\_widget(self.label)

self.id\_label = Label(text="Member ID", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.65})

self.add\_widget(self.id\_label)

self.idofguest = TextInput(multiline=False, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.65})

self.add\_widget(self.idofguest)

self.password\_label = Label(text="Password", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.55})

self.add\_widget(self.password\_label)

self.passwordofguest = TextInput(multiline=False, password=True, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.55})

self.add\_widget(self.passwordofguest)

self.home = Button(text="Home", font\_size=25, background\_color=(0, 225, 225, 1), color=(225, 225, 225, 1),

size\_hint=(.15, .025), pos\_hint={"x": 0.01, "top": 0.99})

self.home.bind(on\_release=self.call\_home)

self.add\_widget(self.home)

self.memberregistration = Button(text="Login", font\_size=40, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": 0.3})

self.memberregistration.bind(on\_release=self.call\_memberreservation)

self.add\_widget(self.memberregistration)

def call\_home(self, instances):

self.idofguest.text = ""

self.passwordofguest.text = ""

sm.transition.direction = 'right'

sm.current = 'Home'

def call\_memberreservation(self, instances):

ID = self.idofguest.text

global member\_identifier

member\_identifier = ID

password = self.passwordofguest.text

memberinfo = member\_info()

state = ''

for member in memberinfo:

if ID in member and password in member:

self.parent.get\_screen("MemberReservation").\_\_init\_\_()

sm.transition.direction = 'left'

sm.current = 'MemberReservation'

state = 'correct'

break

else:

state = 'incorrect'

if state == 'incorrect':

popup\_incorrect()

self.idofguest.text = ""

self.passwordofguest.text = ""

pass

member\_identifier = ""

mID = ''

# Member reservation screen

class MemberReservation(Screen, FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(MemberReservation, self).\_\_init\_\_(\*\*kwargs)

with self.canvas:

Color(1, 1, 0, 1)

self.rect = Rectangle(size=(3000, 2000))

self.label = Label(text='Member Reservation', font\_size=75, color=(225, 225, 225, 1), size\_hint=(.4, .3),

pos\_hint={'x': 0.31, 'top': 1})

self.add\_widget(self.label)

global member\_identifier

self.id\_label = Label(text="Member ID", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.8})

self.add\_widget(self.id\_label)

self.idofmember = Label(text=member\_identifier, color=(225, 225, 225, 1), font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.8})

self.add\_widget(self.idofmember)

room\_type = room\_types()

self.room\_label = Label(text="Room Type", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.7})

self.add\_widget(self.room\_label)

self.dropdown = DropDown()

for room\_type\_ in room\_type:

type = room\_type\_[0]

self.val = Button(text=type, color=(225, 225, 1, 1), background\_color=(0, 225, 225, 1),

size\_hint\_y=None, height=44)

self.val.bind(on\_release=lambda val: self.dropdown.select(val.text))

self.dropdown.add\_widget(self.val)

self.room = Button(text='Select', color=(225, 225, 1, 1), background\_color=(0, 225, 225, 1),

size\_hint=(.35, .05), pos\_hint={"x": 0.5, "top": 0.7})

self.room.bind(on\_release=self.dropdown.open)

self.add\_widget(self.room)

self.dropdown.bind(on\_select=lambda instance, x: setattr(self.room, 'text', x))

self.noofguests\_label = Label(text="Number of Guests", font\_size=40, color=(225, 225, 1, 1),

size\_hint=(.35, .05), pos\_hint={"x": 0.15, "top": 0.6})

self.add\_widget(self.noofguests\_label)

self.dropdown2 = DropDown()

for num in range(1, 11):

self.val = Button(text=str(num), color=(225, 225, 1, 1), background\_color=(0, 225, 225, 1),

size\_hint\_y=None, height=44)

self.val.bind(on\_release=lambda val: self.dropdown2.select(val.text))

self.dropdown2.add\_widget(self.val)

self.noofguests = Button(text='Select', color=(225, 225, 1, 1), background\_color=(0, 225, 225, 1),

size\_hint=(.35, .05), pos\_hint={"x": 0.5, "top": 0.6})

self.noofguests.bind(on\_release=self.dropdown2.open)

self.add\_widget(self.noofguests)

self.dropdown2.bind(on\_select=lambda instance, x: setattr(self.noofguests, 'text', x))

self.checkin\_label = Label(text="Check-in Date", font\_size=40, color=(225, 225, 1, 1),

size\_hint=(.35, .05), pos\_hint={"x": 0.15, "top": 0.5})

self.add\_widget(self.checkin\_label)

self.checkin = TextInput(multiline=False, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.5})

self.add\_widget(self.checkin)

self.checkout\_label = Label(text="Check-out Date", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.4})

self.add\_widget(self.checkout\_label)

self.checkout = TextInput(multiline=False, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.4})

self.add\_widget(self.checkout)

self.memberlogin = Button(text="Home", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.15, .025), pos\_hint={"x": 0.01, "top": 0.99})

self.memberlogin.bind(on\_release=self.call\_memberlogin)

self.add\_widget(self.memberlogin)

self.member\_submit = Button(text="Submit form as Member", font\_size=40, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": 0.3})

self.member\_submit.bind(on\_release=self.submit)

self.add\_widget(self.member\_submit)

self.delete = Button(text="Delete Acoount", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.15, .025), pos\_hint={"x": 0.83, "top": 0.99})

self.delete.bind(on\_release=self.call\_delete)

self.add\_widget(self.delete)

def call\_memberlogin(self, instances):

self.idofmember.text = ""

self.room.text = "Select"

self.noofguests.text = "Select"

self.checkin.text = ""

self.checkout.text = ""

sm.transition.direction = 'right'

sm.current = 'Home'

def call\_delete(self, instances):

self.parent.get\_screen('MemberDelete').\_\_init\_\_()

sm.transition.direction = 'left'

sm.current = 'MemberDelete'

def submit(self, instances):

ID = member\_identifier

room\_type = self.room.text

noofquests = self.noofguests.text

check\_in = self.checkin.text

check\_out = self.checkout.text

today = date.today()

if ID != "" and room\_type != "Select" and noofquests != "Select" and check\_in > str(today) and\

check\_out > str(today) and check\_out > check\_in:

request\_room\_asmember(ID, room\_type, noofquests, check\_in, check\_out)

self.parent.get\_screen('AdminConfirmation').\_\_init\_\_()

popup\_valid\_submission()

else:

popup\_invalid\_submission()

self.idofmember.text = ""

self.room.text = "Select"

self.noofguests.text = "Select"

self.checkin.text = ""

self.checkout.text = ""

pass

# Member signup screen

class Createmember(Screen, FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(Createmember, self).\_\_init\_\_(\*\*kwargs)

with self.canvas:

Color(1, 1, 0, 1)

self.rect = Rectangle(size=(3000, 2000))

self.label = Label(text="Member Sign Up", font\_size=75, color=(225, 225, 225, 1), size\_hint=(.4, .3),

pos\_hint={"x": 0.31, "top": 1})

self.add\_widget(self.label)

self.name\_label = Label(text="Name", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.8})

self.add\_widget(self.name\_label)

self.nameofmember = TextInput(multiline=False, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.8})

self.add\_widget(self.nameofmember)

self.phone\_label = Label(text="Phone Number", font\_size=40, color=(225, 225, 1, 1),

size\_hint=(.35, .05), pos\_hint={"x": 0.15, "top": 0.7})

self.add\_widget(self.phone\_label)

self.phoneofmember = TextInput(multiline=False, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.7})

self.add\_widget(self.phoneofmember)

self.email\_label = Label(text="Email", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.6})

self.add\_widget(self.email\_label)

self.emailofmember = TextInput(multiline=False, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.6})

self.add\_widget(self.emailofmember)

self.pass\_label = Label(text="Password", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.5})

self.add\_widget(self.pass\_label)

self.passofmember = TextInput(multiline=False, password=True, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.5})

self.add\_widget(self.passofmember)

self.pass2\_label = Label(text="Confirm Password", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.4})

self.add\_widget(self.pass2\_label)

self.pass2ofmember = TextInput(multiline=False, password=True, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.4})

self.add\_widget(self.pass2ofmember)

global mID

mID = makememberid()

self.id\_label = Label(text="Your ID will be", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.3})

self.add\_widget(self.id\_label)

self.id\_value = Label(text=mID, font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.3})

self.add\_widget(self.id\_value)

self.home = Button(text="Home", font\_size=25, background\_color=(0, 225, 225, 1), color=(225, 225, 225, 1),

size\_hint=(.15, .025), pos\_hint={"x": 0.01, "top": .99})

self.home.bind(on\_release=self.call\_home)

self.add\_widget(self.home)

self.submit = Button(text="Done", font\_size=40, background\_color=(0, 225, 225, 1), color=(225, 225, 225, 1),

size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": .2})

self.submit.bind(on\_release=self.call\_create\_member)

self.add\_widget(self.submit)

def call\_home(self, instances):

self.nameofmember.text = ""

self.phoneofmember.text = ""

self.emailofmember.text = ""

self.passofmember.text = ""

self.pass2ofmember.text = ""

sm.transition.direction = 'right'

sm.current = 'Home'

def call\_create\_member(self, instances):

name = self.nameofmember.text

phone = self.phoneofmember.text

email = self.emailofmember.text

password = self.passofmember.text

password\_check = self.pass2ofmember.text

global mID

phoneCheck = ''

for i in range(0, len(phone)):

if phone[i] == "-":

phoneCheck = phoneCheck

else:

phoneCheck += phone[i]

continue

mail\_list = ['@gmail.com', '@yahoo.com', '@hotmail.com', '@aol.com', '@msn.com', '@icloud.com']

if '@' in email:

mail = email[email.index('@'):len(email)+1]

else:

mail = email

if name != '' and phoneCheck.isnumeric() and mail in mail\_list and password != '' and\

password == password\_check:

create\_member(mID, phone, email, name, password)

self.parent.get\_screen('Createmember').\_\_init\_\_()

popup\_account\_created()

else:

popup\_account\_notcreated()

self.nameofmember.text = ""

self.phoneofmember.text = ""

self.emailofmember.text = ""

self.passofmember.text = ""

self.pass2ofmember.text = ""

pass

# Delete my member account screen

class MemberDelete(Screen, FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(MemberDelete, self).\_\_init\_\_(\*\*kwargs)

with self.canvas:

Color(1, 1, 0, 1)

self.rect = Rectangle(size=(3000, 2000))

self.label = Label(text='Delete my Member account', font\_size=75, color=(225, 225, 225, 1), size\_hint=(.4, .3),

pos\_hint={'x': 0.31, 'top': 1})

self.add\_widget(self.label)

global member\_identifier

self.id\_label = Label(text="Member ID", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.65})

self.add\_widget(self.id\_label)

self.idofmember = TextInput(text=member\_identifier, multiline=False, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.65})

self.add\_widget(self.idofmember)

self.password\_label = Label(text="Password", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.55})

self.add\_widget(self.password\_label)

self.password = TextInput(multiline=False, password=True, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.55})

self.add\_widget(self.password)

self.memberconfirm = Button(text="Back", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.15, .025), pos\_hint={"x": 0.01, "top": 0.99})

self.memberconfirm.bind(on\_release=self.call\_memberconfirmation)

self.add\_widget(self.memberconfirm)

self.delete = Button(text="Confirm Delete", font\_size=40, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": 0.3})

self.delete.bind(on\_release=self.call\_deleter)

self.add\_widget(self.delete)

def call\_memberconfirmation(self, instances):

self.idofmember.text = ""

self.password.text = ""

sm.transition.direction = 'right'

sm.current = 'MemberReservation'

def call\_deleter(self, instances):

ID = self.idofmember.text

password = self.password.text

memberinfo = member\_info()

state = ''

for member in memberinfo:

if ID in member and password in member:

member\_delete(ID)

self.parent.get\_screen("AdminConfirmation").\_\_init\_\_()

self.parent.get\_screen("BookedRooms").\_\_init\_\_()

popup\_member\_delete()

state = 'correct'

self.idofmember.text = ""

break

else:

state = 'incorrect'

if state == 'incorrect':

popup\_member\_delete\_failure()

self.password.text = ""

pass

# Admin login screen

class AdminLogin(Screen, FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(AdminLogin, self).\_\_init\_\_(\*\*kwargs)

with self.canvas:

Color(1, 1, 0, 1)

self.rect = Rectangle(size=(3000, 2000))

self.label = Label(text="Admin Login", font\_size=75, color=(225, 225, 225, 1), size\_hint=(.4, .3),

pos\_hint={"x": 0.31, "top": 1})

self.add\_widget(self.label)

self.id\_label = Label(text="Admin ID", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.65})

self.add\_widget(self.id\_label)

self.idofadmin = TextInput(multiline=False, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.65})

self.add\_widget(self.idofadmin)

self.password\_label = Label(text="Password", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.55})

self.add\_widget(self.password\_label)

self.passwordofadmin = TextInput(multiline=False, password=True, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.55})

self.add\_widget(self.passwordofadmin)

self.home = Button(text="Home", font\_size=25, background\_color=(0, 225, 225, 1), color=(225, 225, 225, 1),

size\_hint=(.15, .025), pos\_hint={"x": 0.01, "top": 0.99})

self.home.bind(on\_release=self.call\_home)

self.add\_widget(self.home)

self.adminconfirmation = Button(text="Login", font\_size=40, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.4, .15), pos\_hint={"x": 0.3, "top": 0.3})

self.adminconfirmation.bind(on\_release=self.call\_adminconfirmation)

self.add\_widget(self.adminconfirmation)

def call\_home(self, instances):

self.idofadmin.text = ""

self.passwordofadmin.text = ""

sm.transition.direction = 'right'

sm.current = 'Home'

def call\_adminconfirmation(self, instances):

ID = self.idofadmin.text

global admin\_identifier

admin\_identifier = ID

password = self.passwordofadmin.text

admininfo = admin\_info()

state = ''

for admin in admininfo:

if ID in admin and password in admin:

self.parent.get\_screen("AdminConfirmation").\_\_init\_\_()

sm.transition.direction = 'left'

sm.current = 'AdminConfirmation'

state = 'correct'

break

else:

state = 'incorrect'

if state == 'incorrect':

popup\_incorrect()

self.idofadmin.text = ""

self.passwordofadmin.text = ""

pass

admin\_identifier = ""

id\_ = ''

# Admin room confirmation screen

class AdminConfirmation(Screen, FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(AdminConfirmation, self).\_\_init\_\_(\*\*kwargs)

with self.canvas:

Color(1, 1, 0, 1)

self.rect = Rectangle(size=(3000, 2000))

global admin\_identifier

self.label = Label(text=admin\_identifier, font\_size=75, color=(225, 225, 225, 1),

size\_hint=(.4, .1), pos\_hint={'x': 0.31, 'top': 0.95})

self.add\_widget(self.label)

self.label = Label(text='Admin Confirmation', font\_size=75, color=(225, 225, 225, 1),

size\_hint=(.4, .1), pos\_hint={'x': 0.31, 'top': 0.875})

self.add\_widget(self.label)

x = [0.15, 0.25, 0.335, 0.45, 0.55, 0.65, 0.75, 0.85]

top = 0.75

header\_string = ['Room Type', 'ID', 'Phone No.', 'Name', 'No. of guests', 'Check-in', 'Check-out', 'Approve']

for i in range(0, len(header\_string)):

self.header = Label(text=header\_string[i], font\_size=30, color=(225, 225, 225, 1), size\_hint=(0, 0),

pos\_hint={'x': x[i], 'top': top})

self.add\_widget(self.header)

requested = requested\_rooms()

self.checkref = {}

x = [0.15, 0.25, 0.335, 0.465, 0.55, 0.65, 0.75, 0.85]

top = 0.7

for request in requested:

for i in range(0, len(request)):

self.value = Label(text=str(request[i]), font\_size=25, color=(225, 225, 225, 1), size\_hint=(.0, .0),

pos\_hint={'x': x[i], 'top': top})

self.add\_widget(self.value)

self.check = CheckBox(size\_hint=(.02, .02), pos\_hint={'x': x[7], 'top': top + 0.01}, active=False)

self.add\_widget(self.check)

self.checkref[request[3]] = self.check

top -= 0.05

self.submit = Button(text="Confirm reservation", font\_size=40, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.4, .1), pos\_hint={"x": 0.3, "top": 0.125})

self.submit.bind(on\_release=self.call\_confirm)

self.add\_widget(self.submit)

self.delete = Button(text="Delete reservation", font\_size=30, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.2, .05), pos\_hint={"x": 0.05, "top": 0.1})

self.delete.bind(on\_release=self.call\_delete)

self.add\_widget(self.delete)

self.home = Button(text="Home", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.15, .025), pos\_hint={"x": 0.01, "top": 0.99})

self.home.bind(on\_release=self.call\_home)

self.add\_widget(self.home)

self.view = Button(text="Booked rooms", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.15, .025), pos\_hint={"x": 0.83, "top": 0.99})

self.view.bind(on\_release=self.call\_booked)

self.add\_widget(self.view)

self.view2 = Button(text="Checked-In rooms", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.15, .025), pos\_hint={"x": 0.83, "top": 0.96})

self.view2.bind(on\_release=self.call\_booked2)

self.add\_widget(self.view2)

self.new\_admin = Button(text="New Admin", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.15, .025), pos\_hint={"x": 0.63, "top": 0.99})

self.new\_admin.bind(on\_release=self.call\_newadmin)

self.add\_widget(self.new\_admin)

self.new\_room = Button(text="Create room", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.15, .025), pos\_hint={"x": 0.43, "top": 0.99})

self.new\_room.bind(on\_release=self.call\_newroom)

self.add\_widget(self.new\_room)

self.admindelete = Button(text="Delete My Account", font\_size=25, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.15, .025), pos\_hint={"x": 0.23, "top": 0.99})

self.admindelete.bind(on\_release=self.call\_admindelete)

self.add\_widget(self.admindelete)

def call\_home(self, instances):

sm.transition.direction = 'right'

sm.current = 'Home'

def call\_booked(self, instances):

sm.transition.direction = 'left'

sm.current = 'BookedRooms'

def call\_booked2(self, instances):

sm.transition.direction = 'left'

sm.current = 'CheckedInRooms'

def call\_newadmin(self, instanes):

sm.transition.direction = 'left'

sm.current = 'NewAdmin'

def call\_newroom(self, instances):

sm.transition.direction = 'left'

sm.current = 'NewRoom'

def call\_confirm(self, instances):

for idx, wgt in self.checkref.items():

if wgt.active:

move\_to\_booked\_rooms(idx)

self.parent.get\_screen('AdminConfirmation').\_\_init\_\_()

self.parent.get\_screen('BookedRooms').\_\_init\_\_()

self.parent.get\_screen('Rooms').\_\_init\_\_()

popup\_confirm()

def call\_delete(self, instances):

for idx, wgt in self.checkref.items():

if wgt.active:

delete\_requestedroom(idx)

self.parent.get\_screen('AdminConfirmation').\_\_init\_\_()

self.parent.get\_screen('BookedRooms').\_\_init\_\_()

popup\_confirm\_delete()

def call\_admindelete(self, instances):

popup\_confirm\_admindelete()

pass

# New admin account screen

class NewAdmin(Screen, FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(NewAdmin, self).\_\_init\_\_(\*\*kwargs)

with self.canvas:

Color(1, 1, 0, 1)

self.rect = Rectangle(size=(3000, 2000))

self.label = Label(text='Create New Admin', font\_size=75, color=(225, 225, 225, 1),

size\_hint=(.4, .3), pos\_hint={'x': 0.31, 'top': 1})

self.add\_widget(self.label)

self.phone\_label = Label(text="Phone number", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.8})

self.add\_widget(self.phone\_label)

self.phone = TextInput(multiline=False, font\_size=40, size\_hint=(.35, .05), pos\_hint={"x": 0.5, "top": 0.8})

self.add\_widget(self.phone)

self.email\_label = Label(text="Email", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.7})

self.add\_widget(self.email\_label)

self.email = TextInput(multiline=False, font\_size=40, size\_hint=(.35, .05), pos\_hint={"x": 0.5, "top": 0.7})

self.add\_widget(self.email)

self.name\_label = Label(text="Name", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.6})

self.add\_widget(self.name\_label)

self.name\_ = TextInput(multiline=False, font\_size=40, size\_hint=(.35, .05), pos\_hint={"x": 0.5, "top": 0.6})

self.add\_widget(self.name\_)

self.password\_label = Label(text="Password", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.5})

self.add\_widget(self.password\_label)

self.password = TextInput(multiline=False, password=True, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.5})

self.add\_widget(self.password)

self.pass2\_label = Label(text="Confirm Password", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.4})

self.add\_widget(self.pass2\_label)

self.pass2 = TextInput(multiline=False, password=True, font\_size=40, size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.4})

self.add\_widget(self.pass2)

global id\_

id\_ = makeid()

self.id\_label = Label(text="Your ID will be", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.3})

self.add\_widget(self.id\_label)

self.id\_value = Label(text=id\_, font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.5, "top": 0.3})

self.add\_widget(self.id\_value)

self.submit = Button(text="Create", font\_size=40, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.4, .1), pos\_hint={"x": 0.3, "top": 0.2})

self.submit.bind(on\_release=self.call\_confirm)

self.add\_widget(self.submit)

self.home = Button(text="Back", font\_size=25, background\_color=(0, 225, 225, 1), color=(225, 225, 225, 1),

size\_hint=(.15, .025), pos\_hint={"x": 0.01, "top": 0.99})

self.home.bind(on\_release=self.call\_home)

self.add\_widget(self.home)

def call\_home(self, instances):

self.phone.text = ""

self.email.text = ""

self.name\_.text = ""

self.password.text = ""

self.pass2.text = ""

sm.transition.direction = 'right'

sm.current = 'AdminConfirmation'

def call\_confirm(self, instances):

phone = self.phone.text

email = self.email.text

name = self.name\_.text

password = self.password.text

pass2 = self.pass2.text

global id\_

phoneCheck = ''

for i in range(0, len(phone)):

if phone[i] == '-':

phoneCheck = phoneCheck

else:

phoneCheck += phone[i]

continue

mail\_list = ['@gmail.com', '@yahoo.com', '@hotmail.com', '@aol.com', '@msn.com', '@icloud.com']

if '@' in email:

mail = email[email.index('@'):len(email) + 1]

else:

mail = email

if name != '' and phoneCheck.isnumeric() and mail in mail\_list and password != '' and password == pass2:

create\_admin(id\_, phone, email, name, password)

self.parent.get\_screen('NewAdmin').\_\_init\_\_()

popup\_account\_created()

else:

popup\_account\_notcreated()

self.phone.text = ""

self.email.text = ""

self.name\_.text = ""

self.password.text = ""

self.pass2.text = ""

pass

# New room create screen

class NewRoom(Screen, FloatLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(NewRoom, self).\_\_init\_\_(\*\*kwargs)

with self.canvas:

Color(1, 1, 0, 1)

self.rect = Rectangle(size=(3000, 2000))

self.label = Label(text='Create New Room', font\_size=75, color=(225, 225, 225, 1),

size\_hint=(.4, .3), pos\_hint={'x': 0.31, 'top': 1})

self.add\_widget(self.label)

self.roomno\_label = Label(text="Room Number", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.7})

self.add\_widget(self.roomno\_label)

self.roomno = TextInput(multiline=False, font\_size=40, size\_hint=(.35, .05), pos\_hint={"x": 0.5, "top": 0.7})

self.add\_widget(self.roomno)

self.roomtype\_label = Label(text="Room Type", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.6})

self.add\_widget(self.roomtype\_label)

self.roomtype = TextInput(multiline=False, font\_size=40, size\_hint=(.35, .05), pos\_hint={"x": 0.5, "top": 0.6})

self.add\_widget(self.roomtype)

self.roomcap\_label = Label(text="Room Capacity", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.5})

self.add\_widget(self.roomcap\_label)

self.roomcap = TextInput(multiline=False, font\_size=40, size\_hint=(.35, .05), pos\_hint={"x": 0.5, "top": 0.5})

self.add\_widget(self.roomcap)

self.roomprice\_label = Label(text="Room Price", font\_size=40, color=(225, 225, 1, 1), size\_hint=(.35, .05),

pos\_hint={"x": 0.15, "top": 0.4})

self.add\_widget(self.roomprice\_label)

self.roomprice = TextInput(multiline=False, font\_size=40, size\_hint=(.35, .05), pos\_hint={"x": 0.5, "top": 0.4})

self.add\_widget(self.roomprice)

self.submit = Button(text="Create", font\_size=40, background\_color=(0, 225, 225, 1),

color=(225, 225, 225, 1), size\_hint=(.4, .1), pos\_hint={"x": 0.3, "top": 0.2})

self.submit.bind(on\_release=self.call\_confirm)

self.add\_widget(self.submit)

self.home = Button(text="Back", font\_size=25, background\_color=(0, 225, 225, 1), color=(225, 225, 225, 1),

size\_hint=(.15, .025), pos\_hint={"x": 0.01, "top": 0.99})

self.home.bind(on\_release=self.call\_home)

self.add\_widget(self.home)

def call\_home(self, instances):

self.roomno.text = ""

self.roomcap.text = ""

self.roomtype.text = ""

self.roomprice.text = ""

sm.transition.direction = 'right'

sm.current = 'AdminConfirmation'

def call\_confirm(self, instances):

roomno = self.roomno.text

roomcap = self.roomcap.text

roomtype = self.roomtype.text

roomprice = self.roomprice.text

if roomno != '' and roomcap != '' and roomtype != '' and roomprice:

create\_room(roomno, roomcap, roomtype, roomprice)

popup\_room\_created()

self.parent.get\_screen('Rooms').\_\_init\_\_()

self.parent.get\_screen('GuestReservation').\_\_init\_\_()

self.parent.get\_screen('MemberReservation').\_\_init\_\_()

else:

popup\_room\_notcreated()

self.roomno.text = ""

self.roomcap.text = ""

self.roomtype.text = ""

self.roomprice.text = ""

pass

# Screen manager

sm = ScreenManager()

sm.add\_widget(Home(name='Home'))

sm.add\_widget(Create(name='Create'))

sm.add\_widget(Rooms(name='Rooms'))

sm.add\_widget(BookedRooms(name='BookedRooms'))

sm.add\_widget(CheckedInRooms(name="CheckedInRooms"))

sm.add\_widget(GuestReservation(name='GuestReservation'))

sm.add\_widget(MemberLogin(name='MemberLogin'))

sm.add\_widget(MemberReservation(name='MemberReservation'))

sm.add\_widget(Createmember(name='Createmember'))

sm.add\_widget(MemberDelete(name='MemberDelete'))

sm.add\_widget(AdminLogin(name='AdminLogin'))

sm.add\_widget(AdminConfirmation(name='AdminConfirmation'))

sm.add\_widget(NewAdmin(name='NewAdmin'))

sm.add\_widget(NewRoom(name='NewRoom'))

class HotelReservationApp(App):

def build(self):

return sm

if \_\_name\_\_ == "\_\_main\_\_":

HotelReservationApp().run()

***Output Screen***

*Home Screen*

Graphical user interface

Description automatically generated

*Guest Reservation Screen*

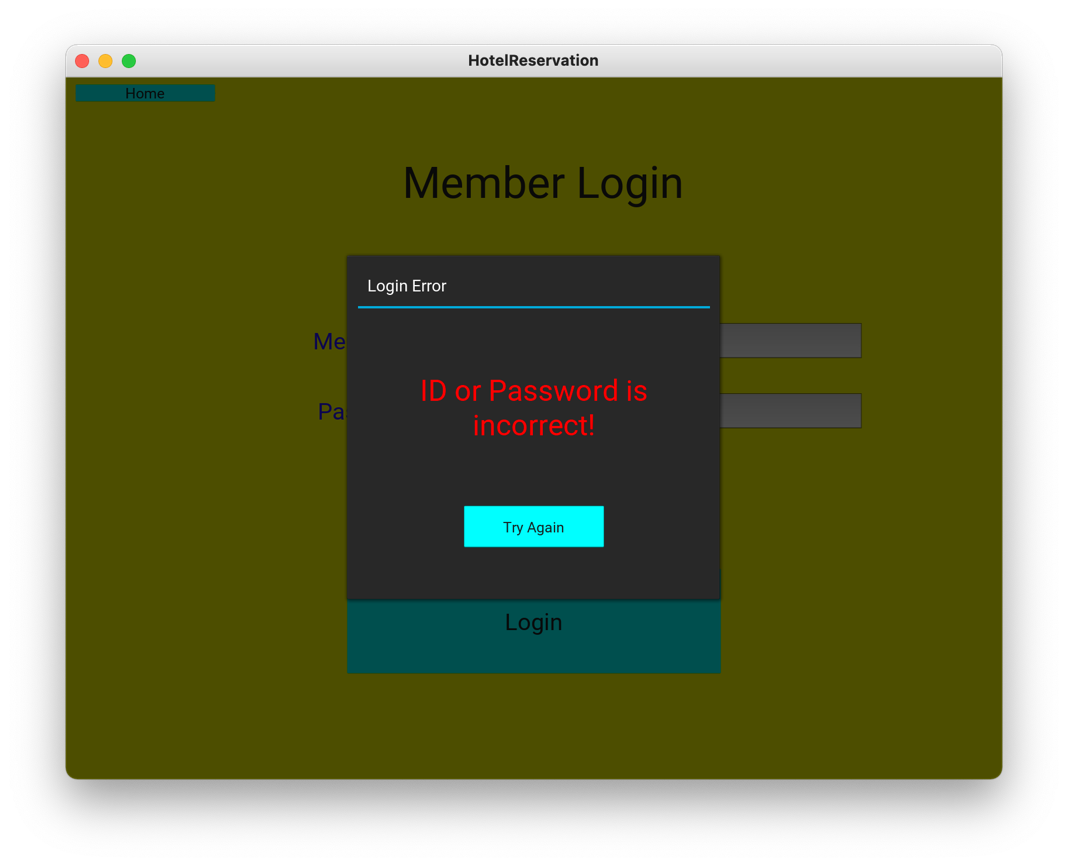
*Graphical user interface

Description automatically generated*

*Member Login Screen*

*Graphical user interface, website

Description automatically generated*

**

*Graphical user interface

Description automatically generated*

*Graphical user interface

Description automatically generated*

*Graphical user interface

Description automatically generated*

*Graphical user interface, application

Description automatically generated*

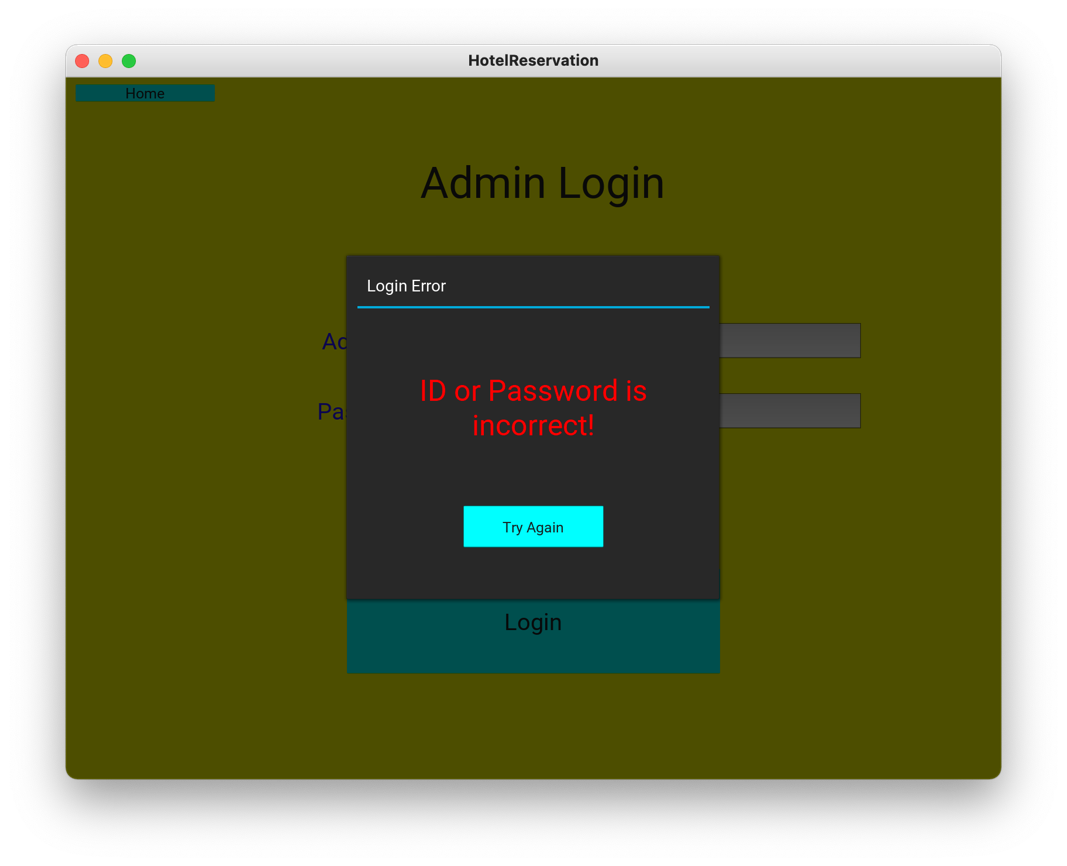
*Graphical user interface

Description automatically generated*

*Admin Login Screen*

*Graphical user interface, website

Description automatically generated*

**

*Graphical user interface, text, website

Description automatically generated*

*Graphical user interface

Description automatically generated*

*Graphical user interface

Description automatically generated*

* *Booked rooms screen*

*Graphical user interface, text, website

Description automatically generated*

* *Checked-in rooms screen*

*Graphical user interface, text

Description automatically generated*

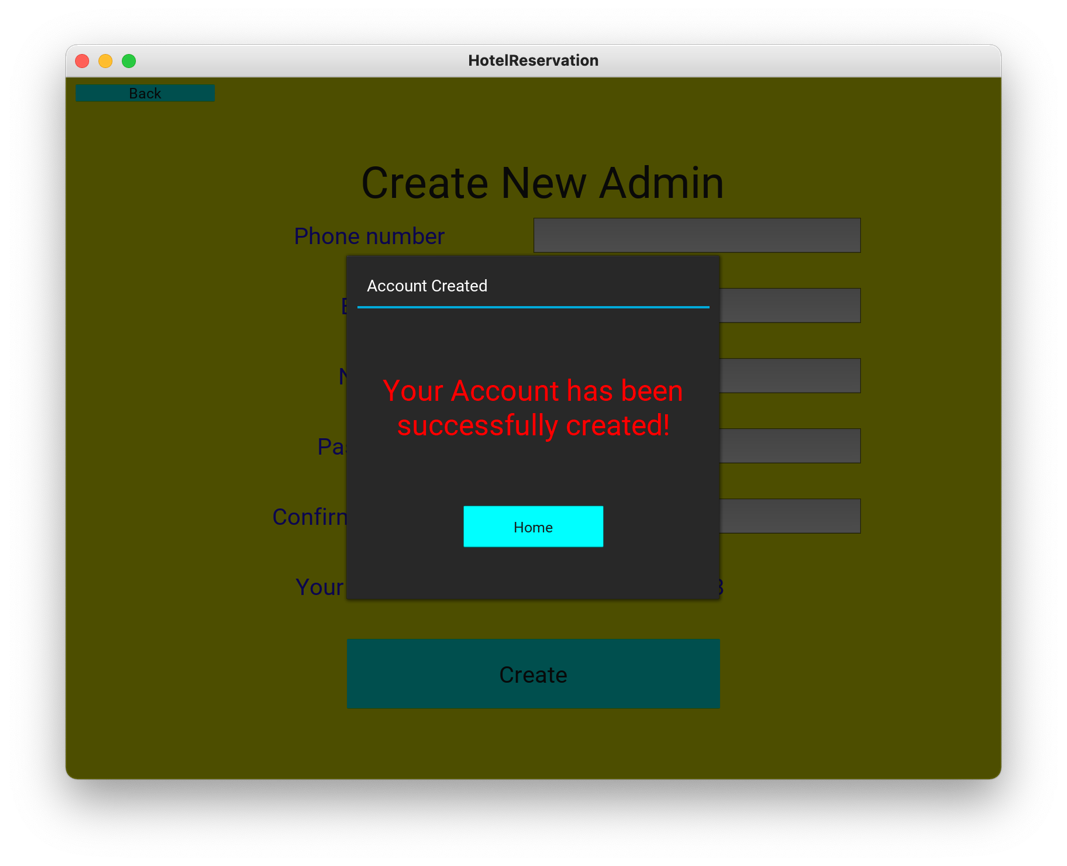
* *Create new admin screen*

*Graphical user interface

Description automatically generated*

*A screenshot of a computer

Description automatically generated*

**

* *Create new room screen*

*Graphical user interface

Description automatically generated*

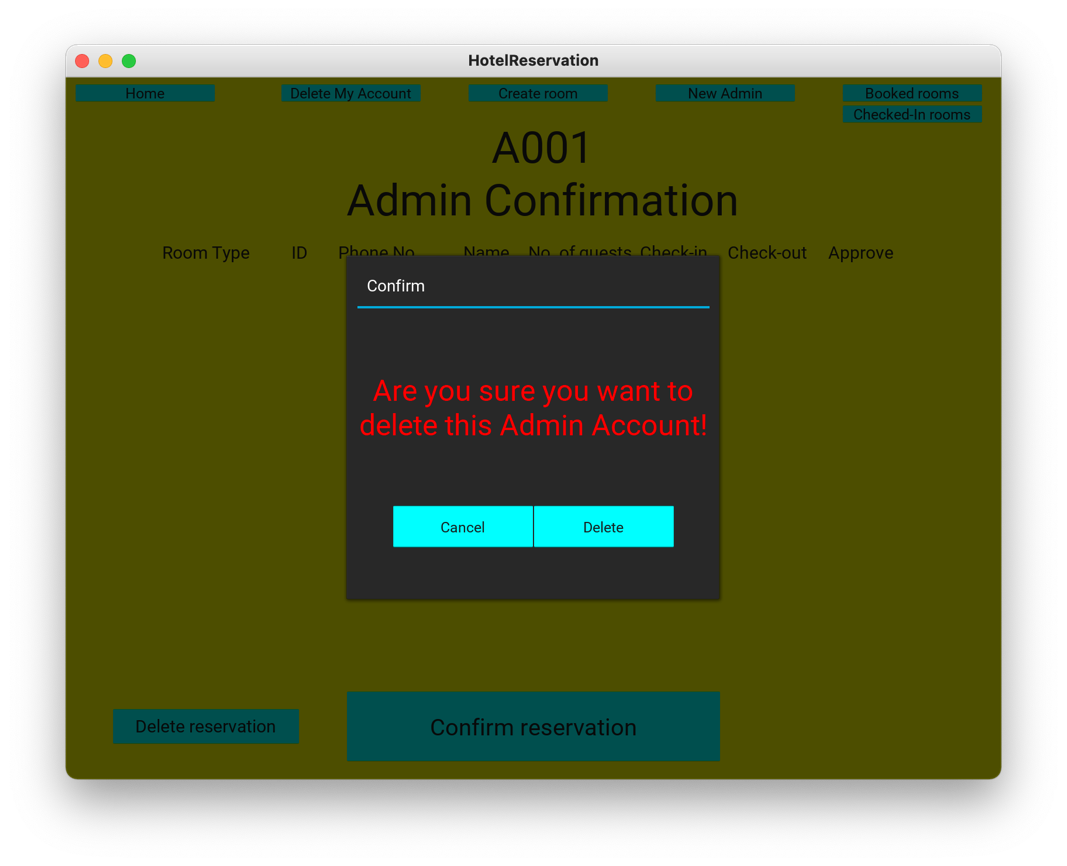
*A screenshot of a computer

Description automatically generated*

*A screenshot of a computer

Description automatically generated*

* *Admin account delete screen*

**

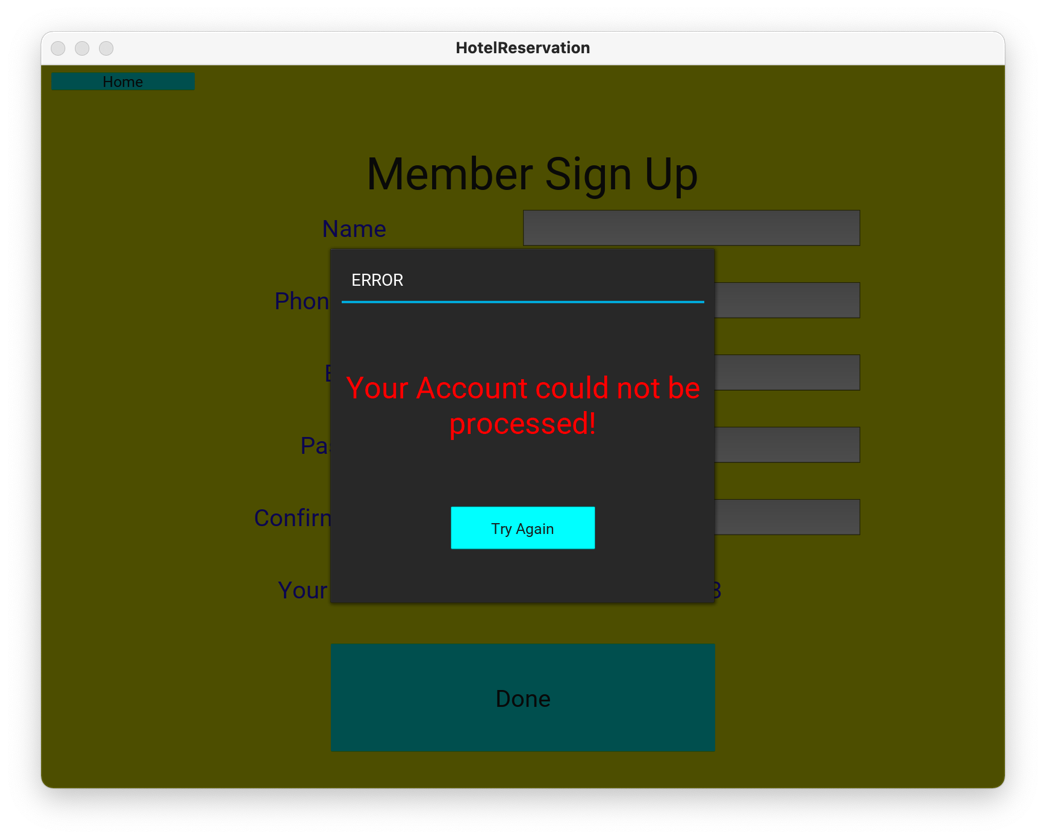
*Member Account Create Screen*

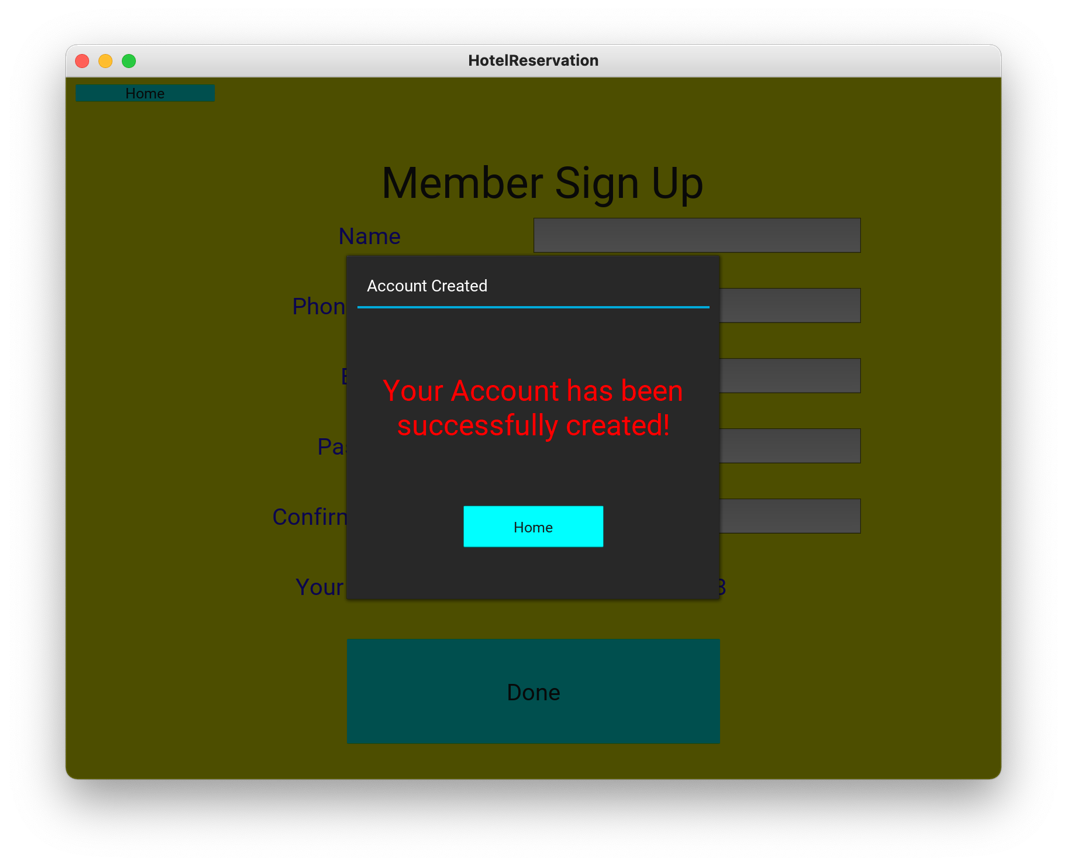
*Graphical user interface

Description automatically generated*

*Graphical user interface

Description automatically generated*

**

**

*Rooms Screen*

*A picture containing table

Description automatically generated*