Final Project Design

Airline Search

Jeremy Johnson

INEW 2332

Summary

This application is a flight reservation system that allows the user to search for and reserve flights. It provides a registration process and requires a log in to reserve flights. The user can search for flights based on airport codes and departure date. It returns a list of flights based on the search criteria. The user can then reserve the desired flight. Finally, it provides the option to show all of the users reserved flights.

Requirements

This application will be composed of 6 GUI screens.
Buttons will redirect user to the appropriate screen.

1. Home Screen
    a. This screen functions as the home screen for the app
    b. Buttons
        i. Register
        ii. Log In
        iii. Search Flights
        iv. Show Reservations

2. Register Screen
    a. This screen allows the user to register and have their data written to the users table of the database.
    b. Buttons
        i. Home
        ii. Register
    c. Input Fields
        i. Username / Email
        ii. Password
        iii. Verify Password
        iv. First Name
        v. Last Name

3. Log In / Log Out Screen
    a. This allows the user to log into or out of their account
    b. Buttons
        i. Log In
        ii. Log Out
        iii. Home
    c. Input Fields
        i. Username / Email
        ii. Password

4. Search Flights
    a. This screen allows the user to search for available flights
    b. Currently only searches for direct flights

c. Requires the use of airport codes
d. Flights will be searched based on
    i. Departure Code
    ii. Destination Code
    iii. Date
e. Buttons
    i. Home
    ii. Log In
    iii. Search Flights
f. Input Fields
    i. Departure Airport Code
    ii. Destination Airport Code
    iii. Date
    iv. Seats Requested
g. The Search Flights Button will open the Available Flights screen

5. Available Flights
    a. This screen shows the available flights based on the users search criteria
    b. The user can highlight the desired flight and reserve it
        i. Reserving a flight will decrease the available seats in the flights table by the number of requested seats and write the data to the reservations table in the database.
    c. Buttons
        i. Home
        ii. Search Flights
        iii. Reserve Flight
    d. Output Fields
        i. Airline
        ii. Flight Number
        iii. Departure Airport Code
        iv. Destination Airport Code
        v. Date
        vi. Time
        vii. Cost
        viii. Number of Seats
        ix. Number of Available Seats

6. Show Reservations
    a. This screen shows the users reserved flights
    b. Buttons
        i. Home
        ii. Log In
    c. Output Fields
        i. First Name
        ii. Last Name
        iii. Airline
        iv. Flight Number
        v. Departure Airport Code

vi. Destination Airport Code
vii. Date
viii. Time
ix. Number of Reserved Seats

d. Cancel Button
    i. Cancel button deletes the highlighted reservation from the reservations table and increases the available seats in the flights table by the number of seats currently reserved

Future Features

Features to be added to project time permitting

1. Implement feature to search for round trip flights
2. Implement feature to show the city and state based on airport codes
    a. Use it to search for flights based on city and state
    b. Use it to display city and state in the reservation summary screen
3. Implement feature to search for flights with multiple stops
4. Add additional User information and transfer all user data to reservation, i.e., address
5. Add multiple users' information for multiple reservations
    a. Add another flier Username, First Name, Last Name
6. Update user information
7. Delete user

Application Decisions

Programming Language

- Python

Testing

- Pytest

GUI Library

- Tkinter

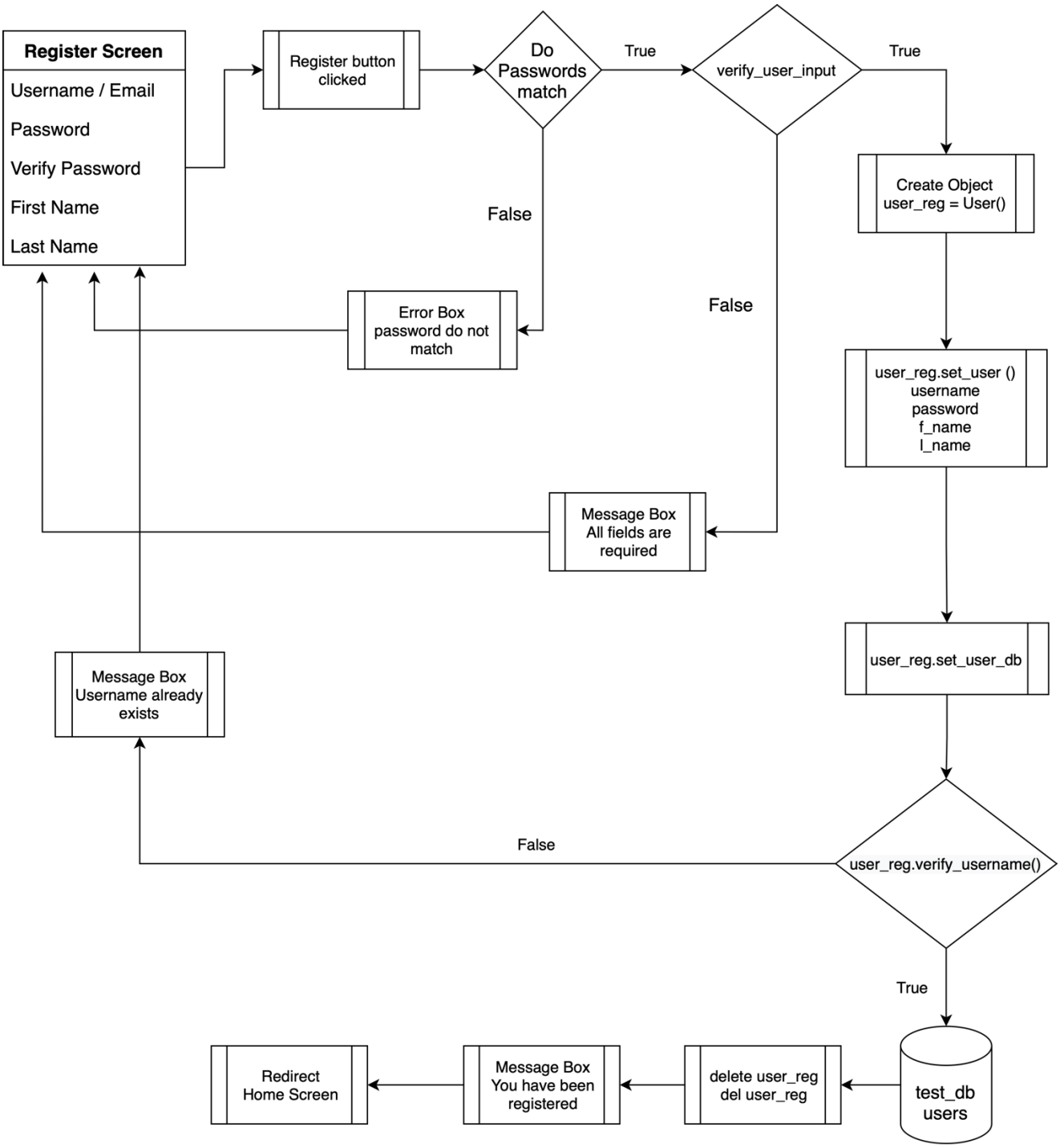
Database

- PostgreSQL
    - test_db
        - Users Table
        - Flights Table
        - Reservations Table

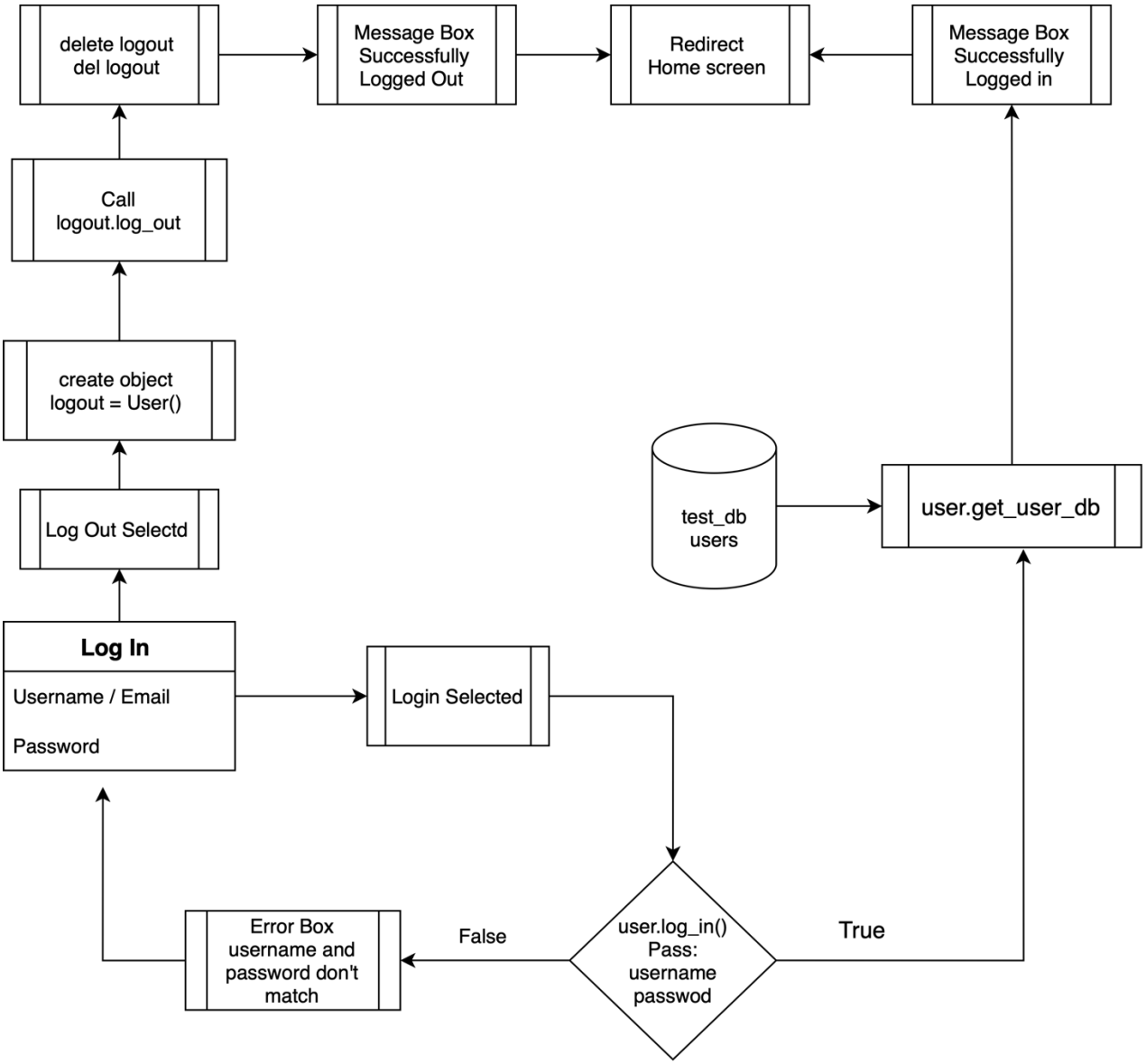
Flow Charts, Diagrams, Database Fields, Tests

Home Screen

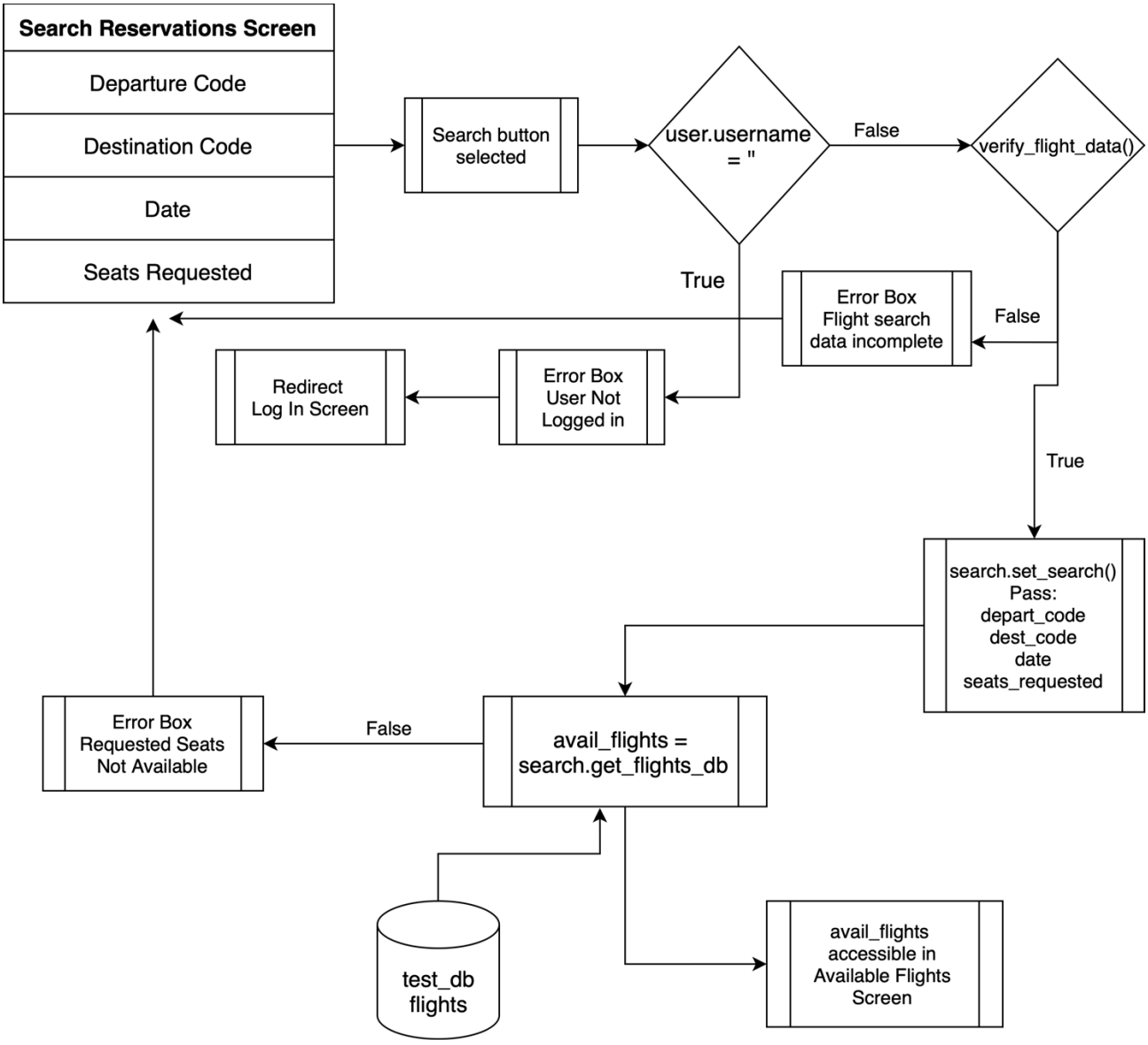
Register Screen



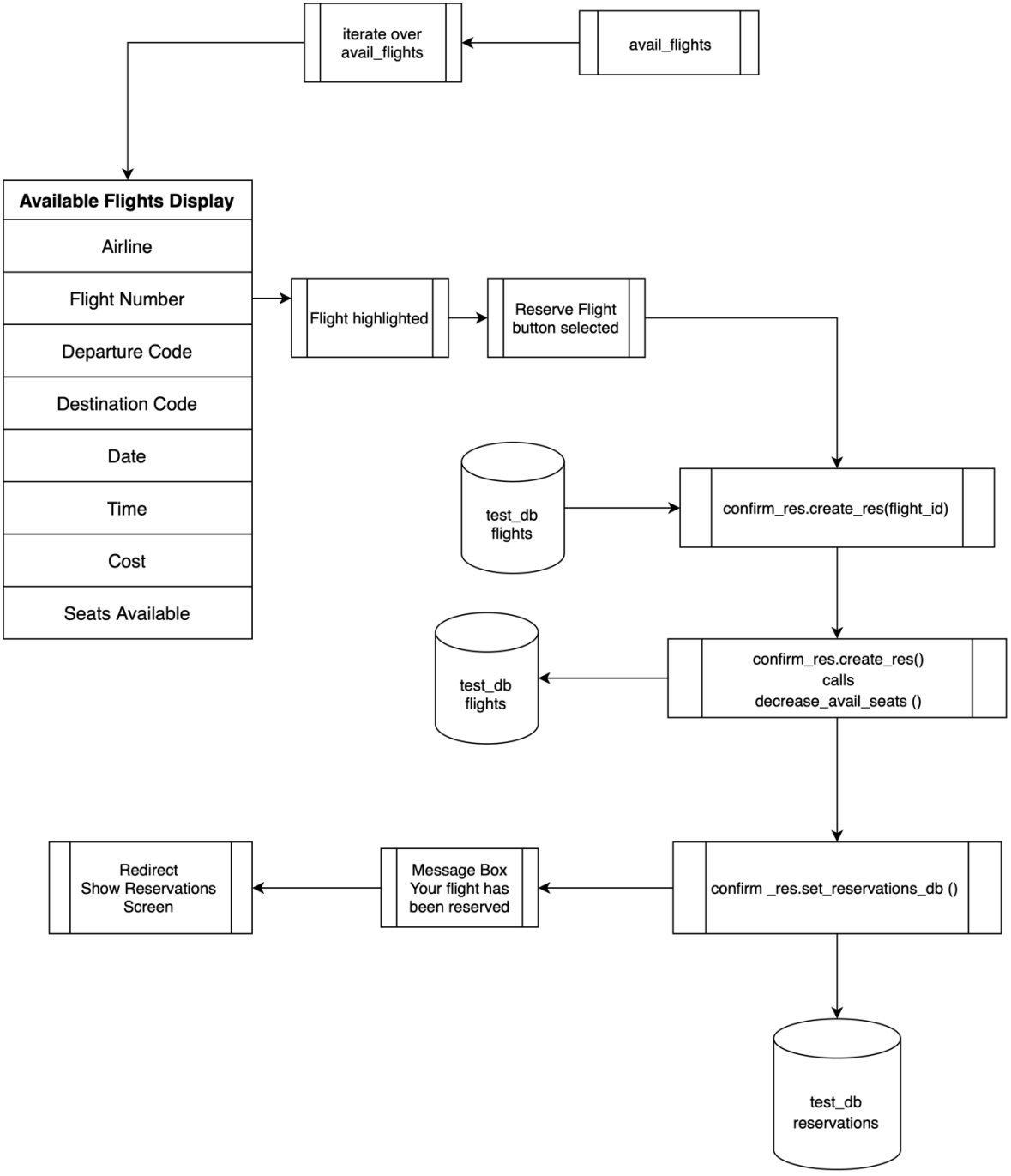
Log In Screen



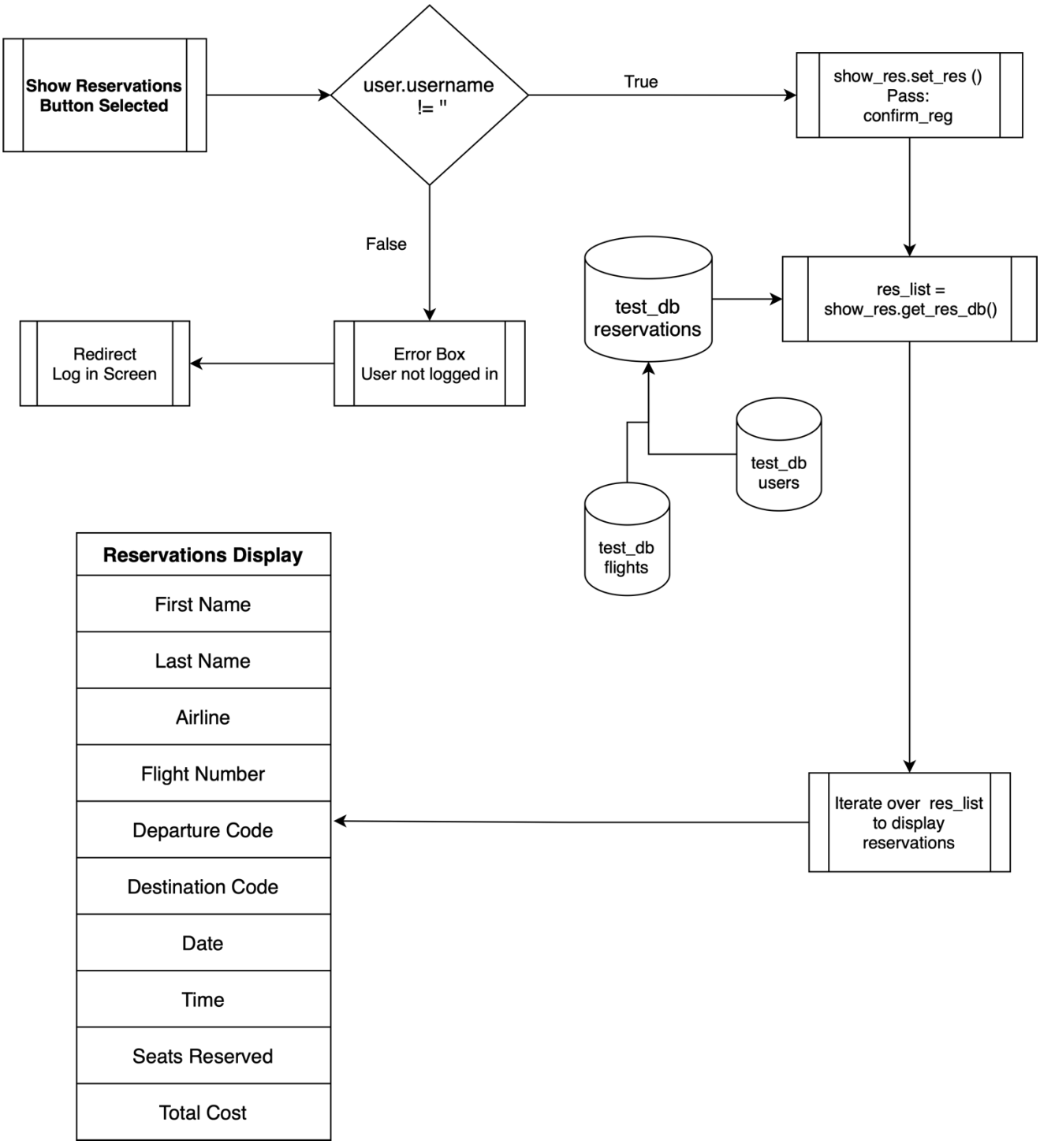
Search Flights



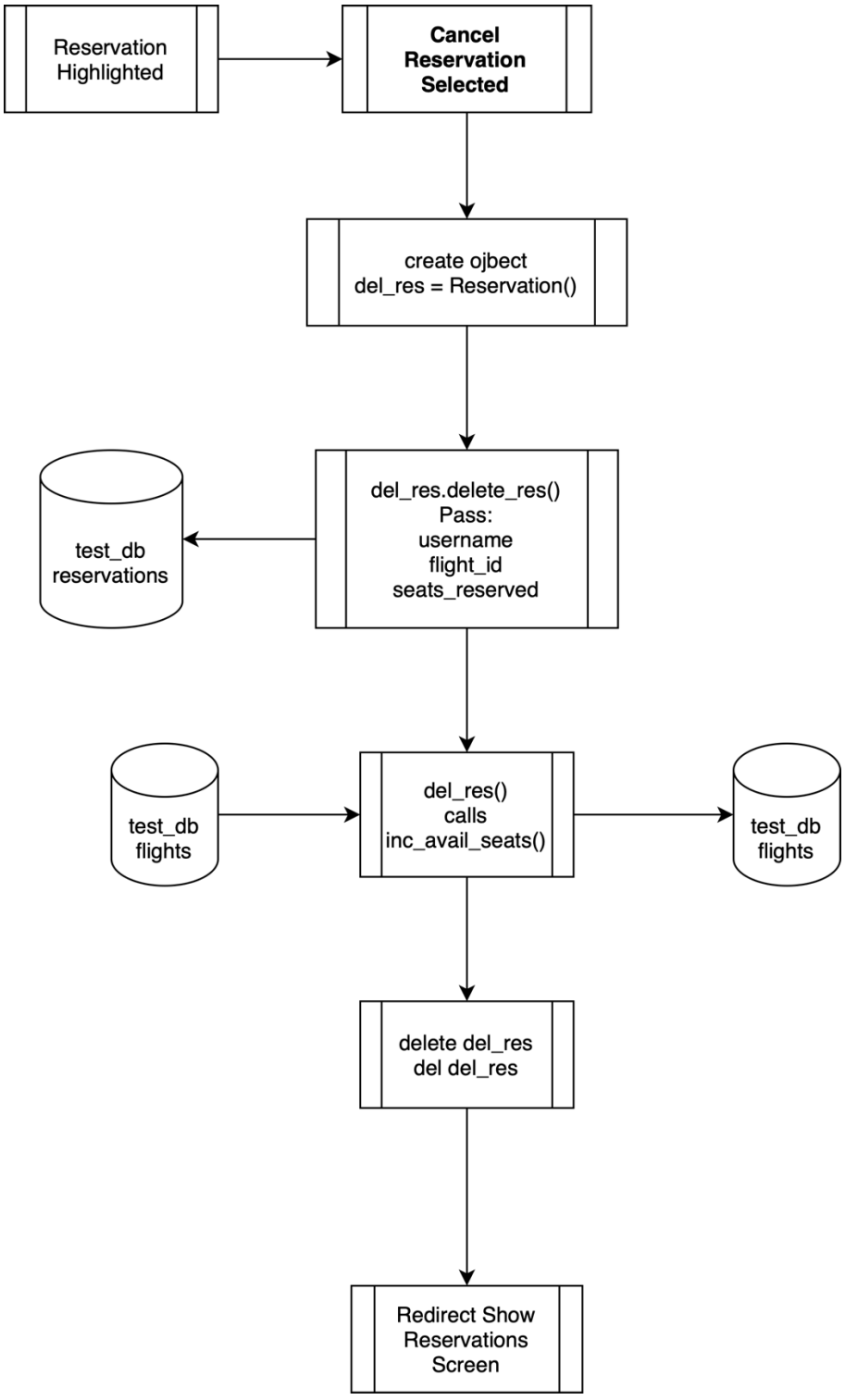
Available Flights



Show Reservations



Cancel Reservation



Database Fields

Flights Table

Table Fields	Datatype
flight_id	Serial Primary Key
Airline	Varchar – Not Null
Flight_num	Varchar – Not Null
Depart_code	Varchar – Not Null
Dest_code	Varchar – Not Null
Depart_Date	Date – Not Null
Depart_Time	Time – Not Null
Cost	int – Not Null
Avail_seats	Int – Not Null

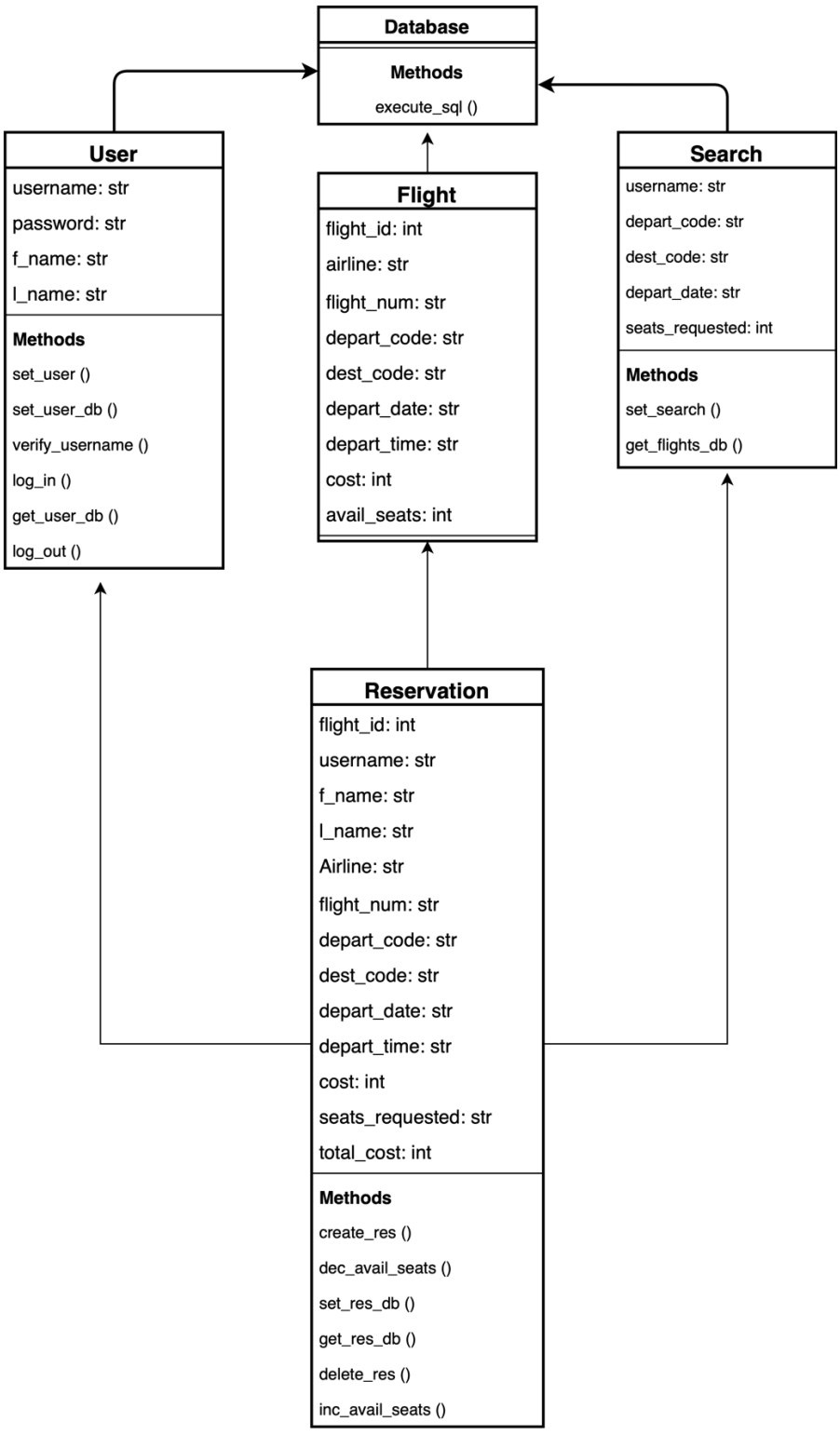
Users Table

Table Fields	Datatype
username	Varchar - Primary Key
Pass	Varchar – Not Null
F_Name	Varchar – Not Null
L_Name	Varchar – Not Null

Reservations Table

Table Fields	Datatype
Flight_id	Int – Primary Key
Username	Varchar – Foreign Key
Seats_requested	Int – Not Null
Total_Cost	Int
Seats_reserved	Int – Not Null

UML Diagrams



Global Variable

1. Global user
    a. User = User ()
2. Global search
    a. Search = Search ()
3. Global confirm_res
    a. Confirm_res = Reservation ()
4. Global show_res
    a. Show_res = Reservation ()
5. Global avail_flights
    a. Avail_flights = []
6. Global res_list
    a. Res_list = []

Class Summary

1. Database
2. User Inherits:
    a. Database
3. Flight Inherits:
    a. Database
4. Search Inherits:
    a. Database
5. Reservation Inherits
    a. User
    b. Flight
    c. Search

Method Summaries

Database ()

1. Connect_db
    a. Input: sql, update
    b. Creates a connection to test_db
    c. If update is True – any command other than SELECT
        i. Runs SQL
        ii. Returns None
    d. If update is False -- for SELECT statements
        i. Runs SQL
        ii. Returns results of SQL query
    e. Closes connection with test_db

User ()

1. Set_users (username, password, f_name, l_name)
    a. Sets data for a user object
    b. Uses user_reg
    c. Sets object values
    d. Returns None
2. set_users_db ()
    a. Updates user table with user registration information
    b. Uses user_reg
    c. Calls verify = self. verify_username ()
    d. If verify = True write SQL and UPDATE users table, return None
    e. If verify = False, return False
3. Verify_username ()
    a. Verifies requested username is unique
    b. Queries the users table to determine if username has been used
    c. Return Boolean
4. Log_in (username, password)
    a. Verifies username and password match
    b. Uses global variable user
    c. Queries users table
    d. Return Boolean
5. Get_user_db ()
    a. Extracts user information based on login credentials
    b. Uses global variable user
    c. Query users table for user information
    d. Returns None

6. Log_out ()
    a. Resets all global variables at log out
    b. Uses global variable user
    c. Rewrites all global variable to initialized state
7. Update_user ()
    a. Future feature
8. Delete_user ()
    a. Future feature

Search ()

1. set_search (depart_code, dest_code, depart_date, seats_requestes)
    a. sets data for the search object
    b. uses global variable search
    c. returns None
2. get_flights_db ()
    a. queris flights table to determine available flights
    b. applies logic to ensure there are enough available seats
    c. returns False if there are no seats
    d. returns list of flights if there are flights available

Reservations ()

1. set_res (confirm_res)
    a. sets values of reservation
    b. users global variable confirm_res
    c. returns None
2. create_res (flight_id)
    a. creates a list of reservation details
    b. uses global variable confirm_res
    c. queries flights table for flight data
    d. incorporates global user data, username, f_nam, l_name
    e. incorporates global search data, seats_requested
    f. calculates and writes total_cost to list
    g. Returns list of flight data
3. Dec_avail_seats
    a. Decreases the available seats field in the flights table by the number of seats selected
    b. Called by create_res ()
    c. UPDATES flights table to the correct number of avail_seats based on seats_requested
    d. Returns None
4. Set_res_db ()

    a. Updates reservations table with the reserved flight information
    b. Uses global variable confirm_res
    c. UPDATE operation on reservations table
    d. Returns None
5. Get_res_db ()
    a. Gets the reservation from the user, flights and users table to display on the Show Reservations screen
    b. Uses global variable show_res
    c. Queries 3 tables to get all reservation information
    d. Returns a list of reserved flights
6. Delete_res (username, flight_id, seats_reserved)
    a. Deletes a reservation from the reservations table
    b. Uses del_res object
    c. DELETE operation on requested reservations table row
    d. Calls inc_avail_seats (flight_id, seats_reserved)
    e. Returns None
7. Inc_avail_seats (flight_id, seats_reserved)
    a. Increases avail_seats in the flights table by the number of seats_reserved in the reservations table
    b. Queries flights table to get avail_seats
    c. Calculates new avail_seats
    d. UPDATE flights table avail_seats by new value, queries by flight_id
    e. Returns None

Flights Table Data Set

flights table data set							
airline	flight_num	depart_code	dest_code	depsrt_date	depart_time	Cost	avail_seats
United	UA1234	IAH	PDX	2-16-2022	9:00	200.00	10
United	UA5678	IAH	SFO	2-16-2022	12:00	300.00	20
United	UA9876	IAH	PDX	2-16-2022	18:00	200.00	30
United	UA9123	IAH	LAX	2-17-2022	14:00	400.00	40
American AIrlines	AA1234	IAH	PDX	2-16-2022	11:00	200.00	50
American Airlines	AA9876	IAH	PDX	2-16-2022	20:00	200.00	60
American Airlines	AA5678	IAH	SFO	2-16-2022	22:00	300.00	70
American Airlines	AA9123	IAH	LAX	2-19-2022	8:00	400.00	80
United	UA2345	SFO	IAH	2-22-2022	14:00	300.00	90
United	UA3456	LAX	IAH	2-22-2022	14:00	400.00	100
United	UA4567	PDX	IAH	2-22-2022	14:00	200.00	0
American Airlines	AA2345	SFO	IAH	2-22-2022	14:00	300.00	120
American Airlines	AA3456	LAX	IAH	2-22-2022	14:00	400.00	1
American Airlines	AA4567	PDX	IAH	2-22-2022	14:00	200.00	0

Unit Testing

Unit Tests

1. User Class
    a. Set_user ()
    b. Set_user_db ()
    c. Verify_username ()
    d. Log_in ()
    e. Get_user_db ()
    f. Log_out ()

2. Search Class
    a. Seat_search ()
    b. Get_flights_db ()

3. Reservations Class
    a. Set_res ()
    b. Create_res ()
    c. Dec_avail_seats ()
    d. Set_res_db ()
    e. Get_res_db ()
    f. Delete_res ()
    g. Inc_avail_seats ()

1) User Class

1) Create object user_reg = User ()
2) Create user1
    a. set_user (username, password, f_name, l_name)
        i. User1, username = johndoe, password = 1234, f_name = john, l_name = doe
3) Create user2
    a. Set_user (username, password, f_name, l_name)
        i. User2, username = tombrown, password = 6789, f_name = tom, l_name = brown
4) Create user3
    a. Set_user (username, password, f_name, l_name)
        i. User3, username = johndoe, password = 4567, f_name = jane, l_name = Jackson
5) Call user1.set_user_db, user2.set_user_db, user3.set_user_db
    a. User1 and user2 return None
    b. Tests verify_username ()
        i. User3 returns False because the username is already taken
6) Delete user_reg – del user_reg
    a. Verify
        i. User_reg.username = ‘’
        ii. User_reg.password = ‘’
        iii. User_reg.f_name = ‘’
        iv. User_reg.l_name =’’
7) Test log_in (username, password)
    a. Create global variable user = User ()
    b. Test 1 – username and password do not match
        i. Log in with user.log_in (‘jamedoe’, ‘1234’)
        ii. Return False
    c. Test 2 – username and password do not match
        i. Log in with user.log_in (‘johndoe’, ‘4567’)
        ii. Return False
    d. Test 3 – username and password match
        i. Log in with user.log_in (‘johndoe’, ‘1234’)
        ii. Return True
8) User1 is logged in with the object user
9) Test get_user_db ()
    a. User1 is logged in
        i. User.get_user_db ()
        ii. Verify
            1. Self.username = johndoe
            2. Self.f_name = john
            3. Self.l_name = doe

b. Test log_out
    i. User.log_out ()
    ii. Verify
        1. user.username = ‘’
        2. user.password = ‘’
        3. user.f_name = ‘’
        4. user.l_name = ‘’
c. Log in as User2 – verify you can log in as a second user
    i. User.log_in (‘tombrown’, ‘6789’)
        1. Return True
    ii. User.get_user_db ()
    iii. Verify
        1. user.username = tombrown
        2. user.password = 6789
        3. user.f_name = tom
        4. user.l_name = brown

10) Logged in as user2

2) Search Class

1) Currently logged in as user2
2) Set global variable search = Search ()
3) Test set_search (depart_code, dest_code, depart_date, seats_requested)
    a. Depart_code = PDX
    b. Dest_code = IAH
    c. Depart_date = 2/22/2022
    d. Seats_requested = 2
4) Verify set_search
    a. Search.depart_code = IAH
    b. Search.dest_code = PDX
    c. Search.depart_date = 2/16/2022
    d. Search.seats_requested = 2
5) Need to reset search data for each flight test search.set_search (new flight data)
6) Test 1 get_flights_db () – no available flights
    a. Avail_flights = Search.get_flights_db (‘PDX’, ‘IAH’, ‘2/22/2022’, 1)
        i. Verify returns False
7) Test 2 get_flights_db () – not enough flights for seats requested
    a. Avail_flights = search.get_flights_db (‘LAX’, ‘IAH’, ‘2/22/2022, 2)
        i. verify returns False
8) Test 3 get_flights_db () – 2 available flights
    a. Search.get_flights_db (‘IAH’, ‘SFO’, ‘2/16/2022’, 1)
    b. Avail_flights = Verify returns 2 flights

        i. [(2, ‘United’, ‘UA5678’, ‘IAH’, ‘SFO’, ‘2022/02/22’, ’12:00’, 300, 20), (7, ‘American Airlines’, ‘AA5678’, ‘IAH’, ‘SFO’, ‘2022/02/22’, ’22:00’, 300, 70)]
9) Test 3 get_flights_db () – 1 available flight
    a. Avail_flights = Search.get_flights_db (‘LAX’, ‘IAH’, ‘2/22/2022’, 1)
    b. Verify returns 1 flight
        i. [(13,’American Airlines’, ‘AA3456’, ‘LAX’, ‘IAH’, ‘2022/02/22’, ’14:00’, 400, 1)]

3) Reservation Class

1) Logged in as user2, tombrown, tom, brown
2) Set global variable confirm_reg = Registration ()
3) Set global variable show_res
4) Tests create_res (flight_id)
    a. Flight_id = 13
    b. Confirm_reg.create_res (13)
    c. Verify
        i. Confirm_reg.username = tombrown
        ii. Confirm_reg.f_name = tom
        iii. Confirm_reg.l_name = brown
        iv. Confirm_reg.flight_id = 13
        v. Confirm_reg.airline = American Airlines
        vi. Confirm_reg.flight_num = AA3456
        vii. Confirm_reg.depart_code = LAX
        viii. Confirm_reg.dest_code = IAH
        ix. Confirm_reg.depart_date = 2/22/2022
        x. Confirm_reg.depart_time = 14:00
        xi. Confirm_reg.cost = 400
        xii. Confirm_reg.avail_seats = 1
        xiii. Confirm_reg.seats_requested = 1
        xiv. Confirm_reg.total_cost = 400
5) Verify dec_avail_seats () based on previous test
    a. Verify – SQL query of flights table?
        i. Flight_id = 13
        ii. New avail_seats = 0
        iii. Seats_reserved = 1
6) Create global variable res_list = []
7) Create global variable show_res = Reservation ()
8) Test get_res_db () – with one flight reserved
    a. Res_list = show_res.get_res_db ()
    b. Verify res_list =

i. [('tom', 'brown', 'American Airlines', 'AA3456', 'LAX', 'IAH', '2022/02/22', '14:00', 1, 400)]

9) Test set_res_db ()
   a. Use registration with flight_id = 13 above
   b. Show_res.set_res_db (confirm_res)
   c. Verify show_res =
      i. Flight_id = 13
      ii. Username = 'tombrown'
      iii. Seats_reserved = 1
      iv. Total_cost = 400
10) Test get_res_db () – with two flights reserved, and 2 seats reserved on second flight
   a. Res_list = Show_res.get_res_db ()
   b. Verify res_list =
      i. [('tom', 'brown', 'American Airlines', 'AA3456', 'LAX', 'IAH', '2022/02/22', '14:00', 1, 400)]
11) Test delete_res (username, flight_id, seats_requested)
   a. Delete flight_id =1
   b. Show_res.delete_res (user.username, 1, search.seats_requested)
      i. Verify
         1. Res_list = show_res.get_res_db ()
         2. Res_list =
         3. [('tom', 'brown', 'American Airlines', 'AA3456', 'LAX', 'IAH', '2022/02/22', '14:00', 1, 400)]
   c. Tests inc_avail_seats (flight_id, seats_reserved)
      i. Verify – SQL of flights table?
         1. Avail_seats = 10
12) Test log_out
   a. Verify
      i. User = User ()
         1. User.username = ''
         2. User.password = ''
         3. User.f_name = ''
         4. User.l_name = ''
      ii. Search = Search ()
         1. Search.username = ''
         2. Search.depart_code = ''
         3. Search.dest_code = ''
         4. Search.depart_date = ''
         5. Search.seats_requested = ''
      iii. Confirm_res = Reservation ()
         1. Confirm_reg.username = ''
         2. Confirm.reg.f_name = ''
         3. Confirm.reg.l_name = ''
         4. Confirm_reg.flight_id = 0

5. Confirm_reg.airline = ‘’
6. Confirm_reg.flight_num = ‘’
7. Confirm_reg.depart_code = ‘’
8. Confirm_reg.dest_code = ‘’
9. Confirm_reg.depart_date = ‘’
10. Confirm_reg.depart_time = ‘’
11. Confirm_reg.cost = 0
12. Confirm_reg.avail_seats = 0

iv. Show_res = Reservation ()
    1. Show_res.total_cost = 0
v. Avail_flights = []
vi. Res_list = []