Iteration 2 operation Contract

Contract CO4 — book_trip

Operation:

book_trip(travelers: list, connection: Connection)

Cross References:

Use Case: Book a Trip

Preconditions

- travelers is a non-empty list with valid name, age, and client_id
- connection is a valid Connection object existing in ConnectionDB

Postconditions

- Instance Creation: A Trip instance t is created with a unique alphanumeric ID
- Association Formed: t.connection is set to the provided Connection
- Instance Creation/Retrieval: For each traveler, a Client instance is created or retrieved and added to ClientDB
- Instance Creation: A Reservation instance is created and linked to the Client and Connection
- Instance Creation: A Ticket instance with a unique numerical ID is created and linked to the Reservation
- Association Formed: Each Reservation is added to t.reservations
- Association Formed: t is added to TripDB.trips

Contract CO5 — add_reservation

Operation:

add_reservation(client: Client)

Cross References:

Use Case: Book a Trip

Preconditions

- Trip instance exists
- client is a valid Client object
- client does not already have a reservation for this trip's connection

Postconditions

- Instance Creation: Reservation and Ticket instances are created
- Association Formed: Reservation is linked to client and connection
- Association Formed: Ticket is linked to Reservation and client
- Attribute Modification: trip.reservations count increases by 1

Contract CO6 — add_client

Operation:

add_client(name: string, age: integer, client_id: string)

Cross References:

Use Case: Maintain Client Records

Preconditions

- name is non-empty
- age is a positive integer
- client_id is unique

Postconditions

- Instance Creation: If client does not exist, create Client and set name, age, and client_id
- Association Formed: Add new Client to ClientDB.clients
- Instance Retrieval: If client exists, return existing Client (no duplication)

Contract CO7 — add_trip

Operation:

add_trip(trip: Trip)

Cross References:

Use Case: Book a Trip

Preconditions

- trip is a valid Trip object
- trip is not None
- trip.trip_id is unique

Postconditions

- Association Formed: trip is added to TripDB.trips
- Attribute Modification: trips collection size increases by 1

Contract CO8 — find_trip

Operation:

find_trip(trip_id: string)

Cross References:

Use Case: View Trip Details

Preconditions

- trip_id is a valid string identifier

Postconditions

- Instance Retrieval: If trip with trip_id exists, return Trip instance
- Null Result: If trip does not exist, return None
- Association Preserved: TripDB.trips remains unchanged

Contract CO9 — find_client

Operation:

find_client(client_id: string)

Cross References:

Use Case: View Client History

Preconditions

- client_id is a valid string identifier

Postconditions

- Instance Retrieval: If client with client_id exists, return Client instance
- Null Result: If client does not exist, return None
- Association Preserved: ClientDB.clients remains unchanged

Contract CO10 — find_trips_by_client

Operation:

find_trips_by_client(client: Client)

Cross References:

Use Case: View Client Trip History

Preconditions

- client is a valid Client object
- client exists in ClientDB

Postconditions

- Instance Creation: A list of Trip instances is created
- Association Formed: List includes trips where client appears in any reservation
- Association Preserved: TripDB.trips remains unchanged
