Test cases

1. Find free court in the given time interval

Tests method from ReservationRepository *findFreeCourtForReservation(LocalDateTime start, LocalDateTime end, Long reservation)*, which returns a list of tennis court IDs that can be assigned to a given reservation, i.e. courts that are free in the given time interval. The third parameter, reservationId is required to update the details of the given reservation.

Given the following records in the database:

Courts table

	id
court#1	1
court#2	2

Reservations table

	id	court_id	start	ending
reservation#1	1	1	5/12/2022 8:00	5/12/2022 9:00
reservation#2	2	2	5/12/2022 8:00	5/12/2022 9:00
reservation#3	3	1	5/12/2022 10:00	5/12/2022 12:00

Tests are run as follows:

Case no	start	end	reservation_id	Expected
1	5/12/2022 8:00	5/12/2022 10:00	4	Empty list
2	5/12/2022 10:00	5/12/2022 11:00	4	[2]
3	5/18/2022 10:00	5/18/2022 12:00	4	[1, 2]
4	5/12/2022 8:00	5/12/2022 9:00	1	[1]
5	5/12/2022 8:00	5/12/2022 9:00	2	[2]
6	5/12/2022 8:00	5/12/2022 9:00	4	Empty list
7	5/20/2022 10:00	5/20/2022 12:00	3	[1, 2]

2. Save reservation

Tests method from ReservationService class *saveReservation(Reservation reservation)* which saves a reservation into the database after validating the reservation details, that is, checks if the reservation does not start in the past, is during an open day and its duration is within the allowed length.

In the following tests, the current date-time is set to 2022-05-10T14:00:00 by ISO 8601 format.

Case no	start	end	duration	Expected
1	5/12/2022 8:00	5/12/2022 10:00	2	saved
2	5/12/2000 10:00	5/12/2000 11:00	1	Exception denoting past reservation thrown
3	5/18/2022 10:00	5/18/2022 22:00	0	Exception denoting invalid duration thrown
4	5/14/2022 8:00	5/14/2022 9:00	1	Exception denoting is not open day thrown
5	5/12/2022 8:00	5/12/2022 9:00	1	saved
6	5/12/2022 18:00	5/12/2022 9:00	-9	Invalid duration exception
7	5/20/2022 10:00	5/20/2022 12:00	2	saved

3. Cancel reservation

This method deletes a reservation from the database, however only if the user requesting the cancellation has created the reservation or the user has an admin role.

Users table

	id	role
user#1	1	user
user#2	2	user
user#3	3	admin

Reservations table

	id	user_id
reservation#1	1	1
reservation#2	2	1
reservation#3	3	2
reservation#4	4	2

Case no	Action	Expected
1	user#1 tries to cancel reservation#1	Success
2	user#2 tries to cancel reservation#1	Access denied exception
3	user#2 tries to cancel reservation#2	Access denied exception
4	user#3 tries to cancel reservation#1	Success
5	user#3 tries to cancel reservation#4	Success
6	user#1 tries to cancel reservation#3	Access denied exception
7	user#2 tries to cancel reservation#4	Success