File / Suite	Cas de tests	Status	Duration (ms)	Error Details
auth-user-integration.test.ts	should login successfully with correct credentials	PASSED	227.53	
auth-user-integration.test.ts	should reject login with non-existent user email	PASSED	88.87	
auth-user-integration.test.ts	should reject login with incorrect password	PASSED	167.07	
auth-user-integration.test.ts	should generate valid JWT token with correct expiration	PASSED	168.91	
auth-user-integration.test.ts	should be case-sensitive for email login	PASSED	90.38	
auth-user-integration.test.ts	should handle special characters in password	PASSED	263.55	
auth-user-integration.test.ts	should return consistent token for same user	PASSED	1357.91	
auth-user-integration.test.ts	should include correct user ID in token	PASSED	175.55	
auth-user-integration.test.ts	should handle multiple users independently	PASSED	350.73	
auth-user-integration.test.ts	should work with ADMIN role users	PASSED	265.34	
auth-user-integration.test.ts	should reject login with empty password	PASSED	181.40	
auth-user-integration.test.ts	should reject login with empty email	PASSED	83.13	
auth-user-integration.test.ts	should handle whitespace in password correctly	PASSED	319.61	
auth-user-integration.test.ts	should allow up to 5 login requests per minute from same IP	PASSED	91.05	
auth-user-integration.test.ts	should reject login with 429 status when rate limit exceeded	PASSED	85.42	
auth-user-integration.test.ts	should track requests by IP address correctly	PASSED	90.57	
auth-user-integration.test.ts	should extract client IP from X-Forwarded-For header	PASSED	79.05	
auth-user-integration.test.ts	should fallback to direct IP when X-Forwarded-For not present	PASSED	84.45	
auth-user-integration.test.ts	should return retry-after value when rate limit reached	PASSED	87.04	
auth-user-integration.test.ts	should allow new requests after rate limit window resets	PASSED	83.93	
auth-user-integration.test.ts	should handle rapid sequential login attempts from same user	PASSED	86.20	
auth-user.test.ts	returns 404 when user is not found	PASSED	1.44	
auth-user.test.ts	returns 400 when password is invalid	PASSED	0.31	
auth-user.test.ts	returns 200 and a token on successful login	PASSED	0.66	
create-reservation-integration.test.ts	should create reservation successfully	PASSED	178.41	
create-reservation-integration.test.ts	should return HTTP 201 status on successful creation	PASSED	99.70	
create-reservation-integration.test.ts	should return created reservation with all fields	PASSED	121.25	
create-reservation-integration.test.ts	should create reservation with correct status	PASSED	107.07	
create-reservation-integration.test.ts	should create multiple reservations for same user in different rooms	PASSED	104.68	

File / Suite	Cas de tests	Status	Duration (ms)	Error Details
create-reservation-integration.test.ts	should create reservation with multiple hours duration	**PASSED**	108.21	
create-reservation-integration.test.ts	should return 400 when userId is not provided	**PASSED**	105.57	
create-reservation-integration.test.ts	should return 400 when roomId is not provided	**PASSED**	92.65	
create-reservation-integration.test.ts	should return 400 when startTime is after endTime	**PASSED**	118.60	
create-reservation-integration.test.ts	should return 400 when startTime is in the past	**PASSED**	106.76	
create-reservation-integration.test.ts	should return 400 when endTime is in the past	**PASSED**	88.05	
create-reservation-integration.test.ts	should return 400 for same startTime and endTime	**PASSED**	86.97	
create-reservation-integration.test.ts	should return 400 when room is already booked for time slot	**PASSED**	92.02	
create-reservation-integration.test.ts	should return 400 when new reservation partially overlaps	**PASSED**	110.83	
create-reservation-integration.test.ts	should return 400 when new reservation contains existing reservation	**PASSED**	91.72	
create-reservation-integration.test.ts	should allow reservation if room is free at different time	**PASSED**	97.74	
create-reservation-integration.test.ts	should allow reservation in different room at same time	**PASSED**	98.12	
create-reservation-integration.test.ts	should allow back-to-back reservations (no gap)	**PASSED**	97.02	
create-reservation-integration.test.ts	should return 500 on database error during overlap check	**PASSED**	104.32	
create-reservation-integration.test.ts	should return 500 on database error during reservation creation	**PASSED**	93.50	
create-reservation-integration.test.ts	should create reservation with very long duration	**PASSED**	97.48	
create-reservation-integration.test.ts	should create reservation with minimum duration (1 minute)	**PASSED**	98.43	
create-reservation-integration.test.ts	should create reservation far in the future	**PASSED**	112.77	
create-reservation-integration.test.ts	should accept various valid date formats	**PASSED**	108.84	
create-reservation-integration.test.ts	should create reservation with milliseconds in timestamp	**PASSED**	110.89	
create-reservation-integration.test.ts	should return created reservation with correct user and room IDs	**PASSED**	104.72	
create-reservation-integration.test.ts	should return correct timestamps in response	**PASSED**	109.60	
create-reservation-integration.test.ts	should create reservation for authenticated user	**PASSED**	102.79	
create-reservation-integration.test.ts	should use authenticated user's ID, not one from body	**PASSED**	198.71	
create-reservation.test.ts	should create reservation successfully	**PASSED**	2.90	
create-reservation.test.ts	should return 400 when user id is missing	**PASSED**	0.41	
create-reservation.test.ts	should return 400 when user.id is null	**PASSED**	0.39	
create-reservation.test.ts	should return 400 when roomId is missing	**PASSED**	0.61	

File / Suite	Cas de tests	Status	Duration (ms)	Error Details
create-reservation.test.ts	should return 400 when roomId is empty string	**PASSED**	0.23	
create-reservation.test.ts	should return 400 when date range is invalid	**PASSED**	0.16	
create-reservation.test.ts	should return 400 when room is already booked for time slot	**PASSED**	0.21	
create-reservation.test.ts	should check for overlapping bookings correctly	**PASSED**	0.20	
create-reservation.test.ts	should return 500 when database error occurs during findMany	**PASSED**	0.32	
create-reservation.test.ts	should return 500 when database error occurs during create	**PASSED**	0.20	
create-reservation.test.ts	should handle multiple overlapping bookings	**PASSED**	0.18	
create-reservation.test.ts	should create reservation with different user and same room	**PASSED**	0.13	
create-reservation.test.ts	should create reservation with long duration	**PASSED**	0.17	
create-reservation.test.ts	should create reservation with short duration (15 minutes)	**PASSED**	0.10	
create-reservation.test.ts	should parse ISO date strings correctly	**PASSED**	0.10	
delete-reservation-integration.test.ts	should delete own reservation successfully	**PASSED**	275.15	
delete-reservation-integration.test.ts	should return HTTP 200 status on successful deletion	**PASSED**	215.82	
delete-reservation-integration.test.ts	should return success message in response	**PASSED**	211.32	
delete-reservation-integration.test.ts	should allow user to delete their own reservation	**PASSED**	199.99	
delete-reservation-integration.test.ts	should delete reservation regardless of time to event	**PASSED**	209.80	
delete-reservation-integration.test.ts	should delete reservation for different rooms	**PASSED**	211.53	
delete-reservation-integration.test.ts	should return 404 when reservation does not exist	**PASSED**	196.98	
delete-reservation-integration.test.ts	should return 404 for already deleted reservation	**PASSED**	192.07	
delete-reservation-integration.test.ts	should return 404 with specific error message	**PASSED**	185.62	
delete-reservation-integration.test.ts	should return 403 when user tries to delete other user's reservation	**PASSED**	196.93	
delete-reservation-integration.test.ts	should verify user owns the reservation before deleting	**PASSED**	179.32	
delete-reservation-integration.test.ts	should include authorization error message	**PASSED**	186.32	
delete-reservation-integration.test.ts	should return 400 when reservationId is not provided	**PASSED**	192.91	
delete-reservation-integration.test.ts	should return 400 when reservationId is empty string	**PASSED**	194.58	
delete-reservation-integration.test.ts	should return 400 when userId is not provided	**PASSED**	193.32	
delete-reservation-integration.test.ts	should return 500 on database error during existence check	**PASSED**	187.87	
delete-reservation-integration.test.ts	should return 500 on database error during deletion	**PASSED**	179.94	

File / Suite	Cas de tests	Status	Duration (ms)	Error Details
delete-reservation-integration.test.ts	should handle generic database errors gracefully	PASSED	170.52	
delete-reservation-integration.test.ts	should delete very old reservation	PASSED	195.11	
delete-reservation-integration.test.ts	should delete reservation with very long duration	PASSED	195.90	
delete-reservation-integration.test.ts	should delete multiple reservations sequentially	PASSED	240.88	
delete-reservation-integration.test.ts	should delete reservation with special characters in ID	PASSED	184.43	
delete-reservation-integration.test.ts	should return response with message property on success	PASSED	203.97	
delete-reservation-integration.test.ts	should return response with error property on failure	PASSED	189.58	
delete-reservation-integration.test.ts	should use authenticated user's ID for ownership check	PASSED	172.97	
delete-reservation-integration.test.ts	should not allow user to delete if user context is missing	PASSED	184.46	
delete-reservation-integration.test.ts	should compare user IDs correctly for authorization	PASSED	208.66	
delete-reservation.test.ts	should delete reservation successfully	PASSED	4.89	
delete-reservation.test.ts	should return 400 when reservationId is missing	PASSED	0.43	
delete-reservation.test.ts	should return 400 when reservationId is empty string	PASSED	0.33	
delete-reservation.test.ts	should return 404 when reservation does not exist	PASSED	0.73	
delete-reservation.test.ts	should return 403 when user is not the owner of reservation	PASSED	0.21	
delete-reservation.test.ts	should return 500 when database findUnique error occurs	PASSED	0.24	
delete-reservation.test.ts	should return 500 when database delete error occurs	PASSED	0.13	
delete-reservation.test.ts	should delete reservation with valid UUID	PASSED	0.14	
delete-reservation.test.ts	should handle user deletion of their own reservation	PASSED	0.26	
delete-reservation.test.ts	should verify ownership before deletion	PASSED	0.16	
delete-reservation.test.ts	should handle multiple deletion attempts by same user	PASSED	0.23	
delete-reservation.test.ts	should allow different users to delete their own reservations	PASSED	0.20	
get-reservation-integration.test.ts	should return user's reservations successfully	PASSED	256.17	
get-reservation-integration.test.ts	should return empty array when user has no reservations	PASSED	215.19	
get-reservation-integration.test.ts	should return 400 when userId is not provided	PASSED	195.16	
get-reservation-integration.test.ts	should return multiple reservations for same user	PASSED	210.01	
get-reservation-integration.test.ts	should return only user's reservations, not others	PASSED	287.33	
get-reservation-integration.test.ts	should return reservations with complete data	PASSED	181.05	

File / Suite	Cas de tests	Status	Duration (ms)	Error Details
get-reservation-integration.test.ts	should return reservations starting from given date	**PASSED**	188.24	
get-reservation-integration.test.ts	should return 400 for invalid date format	**PASSED**	187.64	
get-reservation-integration.test.ts	should return 400 when userId is not provided	**PASSED**	175.84	
get-reservation-integration.test.ts	should return empty array when no reservations after date	**PASSED**	188.87	
get-reservation-integration.test.ts	should accept ISO date format	**PASSED**	194.01	
get-reservation-integration.test.ts	should include reservations starting exactly at given date	**PASSED**	182.82	
get-reservation-integration.test.ts	should return all reservations for a room	**PASSED**	319.13	
get-reservation-integration.test.ts	should return 400 when roomId is not provided	**PASSED**	188.63	
get-reservation-integration.test.ts	should return empty array when room has no reservations	**PASSED**	189.52	
get-reservation-integration.test.ts	should only return reservations for specific room	**PASSED**	216.59	
get-reservation-integration.test.ts	should return all reservations from all users	**PASSED**	278.94	
get-reservation-integration.test.ts	should return 400 when userId is not provided	**PASSED**	170.87	
get-reservation-integration.test.ts	should return array with all reservations	**PASSED**	204.74	
get-reservation-integration.test.ts	should return 500 on database error in getReservationsByUser	**PASSED**	181.27	
get-reservation-integration.test.ts	should return 500 on database error in getReservationAfterDate	**PASSED**	177.32	
get-reservation-integration.test.ts	should return 500 on database error in getAllReservationsByRoomId	**PASSED**	206.52	
get-reservation-integration.test.ts	should return 500 on database error in getAllReservations	**PASSED**	180.29	
get-reservation-integration.test.ts	should return array of reservations with complete data	**PASSED**	181.10	
get-reservation-integration.test.ts	should handle multiple reservations at the same time in different rooms	**PASSED**	190.44	
get-reservation-integration.test.ts	should filter correctly by date with many reservations	**PASSED**	232.01	
get-reservation.test.ts	should return reservations for user	**PASSED**	6.14	
get-reservation.test.ts	should return empty array when user has no reservations	**PASSED**	1.31	
get-reservation.test.ts	should return 400 when user id is missing	**PASSED**	1.23	
get-reservation.test.ts	should return 400 when user.id is null	**PASSED**	1.42	
get-reservation.test.ts	should return 500 when database error occurs	**PASSED**	0.79	
get-reservation.test.ts	should handle multiple reservations for user	**PASSED**	1.00	
get-reservation.test.ts	should return reservations after specified date	**PASSED**	0.47	
get-reservation.test.ts	should return empty array when no reservations after date	**PASSED**	0.13	

File / Suite	Cas de tests	Status	Duration (ms)	Error Details
get-reservation.test.ts	should return 400 when user id is missing	**PASSED**	0.35	
get-reservation.test.ts	should return 400 when date format is invalid	**PASSED**	0.20	
get-reservation.test.ts	should return 400 when date is empty string	**PASSED**	0.10	
get-reservation.test.ts	should handle ISO date format correctly	**PASSED**	0.11	
get-reservation.test.ts	should return 500 when database error occurs	**PASSED**	0.12	
get-reservation.test.ts	should return all reservations for a room	**PASSED**	0.18	
get-reservation.test.ts	should return empty array when room has no reservations	**PASSED**	0.09	
get-reservation.test.ts	should return 400 when roomId is missing	**PASSED**	0.11	
get-reservation.test.ts	should return 400 when roomId is empty string	**PASSED**	0.09	
get-reservation.test.ts	should return 500 when database error occurs	**PASSED**	0.11	
get-reservation.test.ts	should handle multiple reservations from different users	**PASSED**	0.16	
get-reservation.test.ts	should return 401 when authorization header is missing	**PASSED**	0.20	
get-reservation.test.ts	should return 401 when bearer token format is invalid	**PASSED**	0.13	
get-reservation.test.ts	should return 401 when token is invalid	**PASSED**	0.15	
get-reservation.test.ts	should return 403 when user is not admin	**PASSED**	0.16	
get-reservation.test.ts	should call next and set req.user when user is admin	**PASSED**	0.15	
get-reservation.test.ts	should allow admin to get room reservations after authentication	**PASSED**	0.20	
get-reservation.test.ts	should deny non-admin user from getting all room reservations	**PASSED**	0.14	
create-room-integration.test.ts	should create room successfully with valid data	**PASSED**	137.62	
create-room-integration.test.ts	should trim whitespace from room name	**PASSED**	98.58	
create-room-integration.test.ts	should create room with minimum capacity of 1	**PASSED**	94.42	
create-room-integration.test.ts	should create room with large capacity	**PASSED**	92.56	
create-room-integration.test.ts	should create room with empty equipment array	**PASSED**	113.42	
create-room-integration.test.ts	should create room with single equipment item	**PASSED**	93.90	
create-room-integration.test.ts	should create room with multiple equipment items	**PASSED**	94.93	
create-room-integration.test.ts	should return correct HTTP status 201 for creation	**PASSED**	89.29	
create-room-integration.test.ts	should persist room in database	**PASSED**	102.27	
create-room-integration.test.ts	should return 400 when name is missing	**PASSED**	87.63	

File / Suite	Cas de tests	Status	Duration (ms)	Error Details
create-room-integration.test.ts	should return 400 when name is empty string	PASSED	86.07	
create-room-integration.test.ts	should return 400 when name is whitespace only	PASSED	91.27	
create-room-integration.test.ts	should accept room name with special characters	PASSED	86.11	
create-room-integration.test.ts	should accept room name with very long string	PASSED	98.71	
create-room-integration.test.ts	should return 400 when capacity is missing	PASSED	91.86	
create-room-integration.test.ts	should return 400 when capacity is zero	PASSED	93.29	
create-room-integration.test.ts	should return 400 when capacity is negative	PASSED	88.94	
create-room-integration.test.ts	should accept decimal capacity (converted to Int)	PASSED	116.23	
create-room-integration.test.ts	should return 400 when equipment is not an array	PASSED	87.45	
create-room-integration.test.ts	should return 400 when equipment is object	PASSED	85.48	
create-room-integration.test.ts	should return 400 when equipment array contains non-string	PASSED	83.22	
create-room-integration.test.ts	should return 400 when equipment array contains empty string	PASSED	86.54	
create-room-integration.test.ts	should return 400 when equipment array contains whitespace-only string	PASSED	88.33	
create-room-integration.test.ts	should accept equipment with special characters	PASSED	97.22	
create-room-integration.test.ts	should accept equipment with numbers and symbols	PASSED	92.50	
create-room-integration.test.ts	should return 500 on database error	PASSED	85.32	
create-room-integration.test.ts	should create multiple rooms in sequence	PASSED	99.42	
create-room-integration.test.ts	should create room with many equipment items	PASSED	116.86	
create-room-integration.test.ts	should create room with duplicate equipment names (if allowed)	PASSED	92.72	
create-room-integration.test.ts	should create room returns proper structure with ID and timestamp	PASSED	76.55	
create-room.test.ts	should create room successfully with valid data	PASSED	2.98	
create-room.test.ts	should trim whitespace from room name	PASSED	0.43	
create-room.test.ts	should return 400 when room name is empty	PASSED	0.73	
create-room.test.ts	should return 400 when room name is only whitespace	PASSED	1.88	
create-room.test.ts	should return 400 when room name is missing	PASSED	0.66	
create-room.test.ts	should return 400 when capacity is 0	PASSED	0.60	
create-room.test.ts	should return 400 when capacity is negative	PASSED	0.38	
create-room.test.ts	should return 400 when capacity is missing	PASSED	0.36	

File / Suite	Cas de tests	Status	Duration (ms)	Error Details
create-room.test.ts	should return 400 when equipment is not an array	PASSED	0.52	
create-room.test.ts	should return 400 when equipment is object instead of array	PASSED	0.15	
create-room.test.ts	should return 400 when equipment contains non-string items	PASSED	0.13	
create-room.test.ts	should return 400 when equipment contains empty strings	PASSED	0.10	
create-room.test.ts	should return 400 when equipment contains whitespace-only strings	PASSED	0.10	
create-room.test.ts	should create room with empty equipment array	PASSED	0.12	
create-room.test.ts	should return 500 when database error occurs	PASSED	0.20	
create-room.test.ts	should handle large capacity values	PASSED	0.12	
create-room.test.ts	should handle room with many equipment items	PASSED	0.11	
create-room.test.ts	should return 401 when authorization header is missing	PASSED	0.20	
create-room.test.ts	should return 401 when bearer token format is invalid	PASSED	0.12	
create-room.test.ts	should return 401 when token is invalid	PASSED	0.15	
create-room.test.ts	should return 403 when user is not admin	PASSED	0.17	
create-room.test.ts	should call next and set req.user when user is admin	PASSED	0.14	
create-room.test.ts	should allow admin to create room after authentication	PASSED	0.21	
create-room.test.ts	should deny non-admin user from creating room	PASSED	0.16	
delete-room-integration.test.ts	should delete room successfully	PASSED	254.85	
delete-room-integration.test.ts	should delete room with reservations (cascade delete)	PASSED	231.02	
delete-room-integration.test.ts	should return HTTP 200 status on successful deletion	PASSED	214.28	
delete-room-integration.test.ts	should return success message in response	PASSED	199.10	
delete-room-integration.test.ts	should delete room with empty equipment array	PASSED	215.70	
delete-room-integration.test.ts	should delete room regardless of room properties	PASSED	209.47	
delete-room-integration.test.ts	should return 404 when room does not exist	PASSED	177.30	
delete-room-integration.test.ts	should return 404 for already deleted room	PASSED	180.25	
delete-room-integration.test.ts	should return 404 with specific error message for non-existent room	PASSED	177.44	
delete-room-integration.test.ts	should delete associated reservations when room is deleted	PASSED	172.43	
delete-room-integration.test.ts	should delete multiple reservations for the same room	PASSED	195.83	
delete-room-integration.test.ts	should not affect reservations for other rooms	PASSED	199.95	

File / Suite	Cas de tests	Status	Duration (ms)	Error Details
delete-room-integration.test.ts	should return 500 on database error during existence check	**PASSED**	191.29	
delete-room-integration.test.ts	should return 500 on database error during reservation deletion	**PASSED**	182.03	
delete-room-integration.test.ts	should return 500 on database error during room deletion	**PASSED**	201.35	
delete-room-integration.test.ts	should handle generic database errors gracefully	**PASSED**	191.78	
delete-room-integration.test.ts	should delete room with special characters in name	**PASSED**	194.70	
delete-room-integration.test.ts	should delete room with many equipment items	**PASSED**	176.90	
delete-room-integration.test.ts	should delete room with minimum capacity	**PASSED**	189.43	
delete-room-integration.test.ts	should delete room with large capacity	**PASSED**	181.02	
delete-room-integration.test.ts	should delete room with reservations at different times	**PASSED**	206.37	
delete-room-integration.test.ts	should return response with message property	**PASSED**	187.59	
delete-room-integration.test.ts	should return error response with error property on failure	**PASSED**	187.77	
delete-room-integration.test.ts	should delete multiple different rooms	**PASSED**	205.49	
delete-room.test.ts	should delete room successfully	**PASSED**	1.95	
delete-room.test.ts	should return 404 when room does not exist	**PASSED**	0.55	
delete-room.test.ts	should return 500 when database findUnique error occurs	**PASSED**	3.00	
delete-room.test.ts	should return 500 when database delete error occurs	**PASSED**	0.68	
delete-room.test.ts	should delete room with valid id parameter	**PASSED**	0.27	
delete-room.test.ts	should handle deletion of room with complex equipment list	**PASSED**	0.14	
delete-room.test.ts	should return 401 when authorization header is missing	**PASSED**	0.29	
delete-room.test.ts	should return 401 when bearer token format is invalid	**PASSED**	0.15	
delete-room.test.ts	should return 401 when token is invalid	**PASSED**	0.31	
delete-room.test.ts	should return 403 when user is not admin	**PASSED**	0.20	
delete-room.test.ts	should call next and set req.user when user is admin	**PASSED**	0.15	
delete-room.test.ts	should allow admin to delete room after authentication	**PASSED**	0.22	
delete-room.test.ts	should deny non-admin user from deleting room	**PASSED**	0.20	
delete-room.test.ts	should verify admin role check uses database lookup	**PASSED**	0.12	
get-room-integretion.test.ts	should return all rooms successfully	**PASSED**	147.89	
get-room-integretion.test.ts	should return array with correct room properties	**PASSED**	101.51	

File / Suite	Cas de tests	Status	Duration (ms)	Error Details
get-room-integretion.test.ts	should include both test rooms in results	PASSED	104.51	
get-room-integretion.test.ts	should return correct equipment arrays for each room	PASSED	99.71	
get-room-integretion.test.ts	should return correct capacity for each room	PASSED	99.44	
get-room-integretion.test.ts	should return rooms with proper createdAt timestamps	PASSED	95.03	
get-room-integretion.test.ts	should handle database error gracefully	PASSED	94.59	
get-room-integretion.test.ts	should return room by id successfully	PASSED	102.15	
get-room-integretion.test.ts	should return room with all correct fields	PASSED	100.16	
get-room-integretion.test.ts	should return 404 for non-existent room	PASSED	139.50	
get-room-integretion.test.ts	should return correct room among multiple rooms	PASSED	110.48	
get-room-integretion.test.ts	should handle database error for getRoomById	PASSED	102.39	
get-room-integretion.test.ts	should return available time slots when room is fully available	PASSED	103.46	
get-room-integretion.test.ts	should return available time slots with correct start and end times	PASSED	98.31	
get-room-integretion.test.ts	should return 404 for non-existent room	PASSED	98.36	
get-room-integretion.test.ts	should exclude reserved time slots from availability	PASSED	109.61	
get-room-integretion.test.ts	should handle multiple reservations correctly	PASSED	113.24	
get-room-integretion.test.ts	should handle room fully booked	PASSED	103.02	
get-room-integretion.test.ts	should handle availability check for different dates	PASSED	135.57	
get-room-integretion.test.ts	should handle database error gracefully	PASSED	85.69	
get-room-integretion.test.ts	should handle room with no equipment	PASSED	80.25	
get-room-integretion.test.ts	should handle room with capacity of 1	PASSED	77.95	
get-room-integretion.test.ts	should handle room with very large capacity	PASSED	77.88	
get-room-integretion.test.ts	should handle invalid date format for availability	PASSED	81.48	
get-room.test.ts	should return all rooms successfully	PASSED	4.60	
get-room.test.ts	should return empty array when no rooms exist	PASSED		01.02
get-room.test.ts	should return 500 when database error occurs	PASSED	0.97	
get-room.test.ts	should return room successfully when found	PASSED	1.72	
get-room.test.ts	should return 404 when room is not found	PASSED	0.40	
get-room.test.ts	should return 500 when database error occurs	PASSED	0.20	

File / Suite	Cas de tests	Status	Duration (ms)	Error Details
get-room.test.ts	should handle different room types with various equipment	**PASSED**	0.13	
get-room.test.ts	should return 400 when date query parameter is missing	**PASSED**	0.19	
get-room.test.ts	should return 404 when room is not found	**PASSED**	0.29	
get-room.test.ts	should return full day available when no reservations exist	**PASSED**	1.83	
get-room.test.ts	should return available time slots with one reservation in the middle	**PASSED**	0.33	
get-room.test.ts	should return available time slots with multiple reservations	**PASSED**	0.18	
get-room.test.ts	should not include gaps between consecutive reservations	**PASSED**	0.15	
get-room.test.ts	should handle reservation that starts at beginning of day	**PASSED**	0.15	
get-room.test.ts	should return 500 when database error occurs	**PASSED**	0.13	
get-room.test.ts	should handle invalid date format	**PASSED**	0.14	
update-room-integration.test.ts	should update room name successfully	**PASSED**	173.05	
update-room-integration.test.ts	should update room capacity successfully	**PASSED**	96.32	
update-room-integration.test.ts	should update room equipment successfully	**PASSED**	109.81	
update-room-integration.test.ts	should update multiple fields at once	**PASSED**	106.53	
update-room-integration.test.ts	should preserve original values when only updating one field	**PASSED**	95.24	
update-room-integration.test.ts	should update with empty equipment array	**PASSED**	94.53	
update-room-integration.test.ts	should update room name with special characters	**PASSED**	107.50	
update-room-integration.test.ts	should return HTTP 200 on successful update	**PASSED**	98.79	
update-room-integration.test.ts	should return updated room with all fields	**PASSED**	103.55	
update-room-integration.test.ts	should return 400 when name is empty string	**PASSED**	97.27	
update-room-integration.test.ts	should return 400 when capacity is zero	**PASSED**	88.06	
update-room-integration.test.ts	should return 400 when capacity is negative	**PASSED**	94.10	
update-room-integration.test.ts	should return 404 when room does not exist	**PASSED**	91.15	
update-room-integration.test.ts	should allow null values for partial updates (skip null fields)	**PASSED**	93.42	
update-room-integration.test.ts	should allow undefined fields (not included in update)	**PASSED**	92.61	
update-room-integration.test.ts	should handle update with whitespace-only name	**PASSED**	86.88	
update-room-integration.test.ts	should update capacity with minimum value of 1	**PASSED**	82.52	
update-room-integration.test.ts	should update capacity with large value	**PASSED**	93.69	

File / Suite	Cas de tests	Status	Duration (ms)	Error Details
update-room-integration.test.ts	should keep original name when name not provided	PASSED	92.33	
update-room-integration.test.ts	should keep original capacity when capacity not provided	PASSED	94.37	
update-room-integration.test.ts	should keep original equipment when equipment not provided	PASSED	86.20	
update-room-integration.test.ts	should handle empty body (no updates)	PASSED	88.33	
update-room-integration.test.ts	should handle consecutive updates to same room	PASSED	98.33	
update-room-integration.test.ts	should update different rooms independently	PASSED	111.66	
update-room-integration.test.ts	should return 500 on database error during update	PASSED	91.84	
update-room-integration.test.ts	should return 500 on database error during existence check	PASSED	102.82	
update-room-integration.test.ts	should update room name with very long string	PASSED	102.38	
update-room-integration.test.ts	should update equipment with many items	PASSED	101.23	
update-room-integration.test.ts	should preserve createdAt timestamp during update	PASSED	91.53	
update-room-integration.test.ts	should maintain room ID during update	PASSED	96.03	
update-room-integration.test.ts	should update with numeric capacity string	PASSED	99.18	
update-room-integration.test.ts	should return updated room with correct structure	PASSED	96.79	
update-room.test.ts	should update room successfully with all fields	PASSED	5.12	
update-room.test.ts	should update room with only name	PASSED	1.13	
update-room.test.ts	should update room with only capacity	PASSED	01.03	
update-room.test.ts	should update room with only equipment	PASSED	1.30	
update-room.test.ts	should return 400 when room name is empty string	PASSED	0.40	
update-room.test.ts	should return 400 when capacity is 0 (validation rejects 0 and negative)	PASSED	0.17	
update-room.test.ts	should return 400 when capacity is negative	PASSED	0.12	
update-room.test.ts	should return 404 when room does not exist	PASSED	0.14	
update-room.test.ts	should return 500 when database update error occurs	PASSED	0.34	
update-room.test.ts	should handle empty request body	PASSED	0.16	
update-room.test.ts	should handle update with capacity = 1	PASSED	0.14	
update-room.test.ts	should handle update with large capacity	PASSED	0.09	
update-room.test.ts	should update equipment to empty array	PASSED	0.13	
update-room.test.ts	should return 401 when authorization header is missing	PASSED	0.21	

File / Suite	Cas de tests	Status	Duration (ms)	Error Details
update-room.test.ts	should return 401 when bearer token format is invalid	PASSED	0.14	
update-room.test.ts	should return 401 when token is invalid	PASSED	0.15	
update-room.test.ts	should return 403 when user is not admin	PASSED	0.16	
update-room.test.ts	should call next and set req.user when user is admin	PASSED	0.14	
update-room.test.ts	should allow admin to update room after authentication	PASSED	0.35	
update-room.test.ts	should deny non-admin user from updating room	PASSED	0.15	
create-user-integration.test.ts	should create user successfully with valid email and password	PASSED	118.69	
create-user-integration.test.ts	should reject user creation with invalid email	PASSED	90.43	
create-user-integration.test.ts	should reject user creation with weak password (less than 12 characters)	PASSED	83.97	
create-user-integration.test.ts	should reject password without uppercase letter	PASSED	80.93	
create-user-integration.test.ts	should reject password without lowercase letter	PASSED	82.03	
create-user-integration.test.ts	should reject password without number	PASSED	80.57	
create-user-integration.test.ts	should reject password without special character	PASSED	85.68	
create-user-integration.test.ts	should reject invalid role (only ADMIN allowed when specified)	PASSED	0.75	
create-user-integration.test.ts	should allow role to be omitted (defaults to USER)	PASSED	87.11	
create-user-integration.test.ts	should hash password before storing in database	PASSED	188.69	
create-user-integration.test.ts	should reject duplicate email addresses	PASSED	191.54	
create-user-integration.test.ts	should exclude password from response	PASSED	115.40	
create-user-integration.test.ts	should handle missing email field	PASSED	93.36	
create-user-integration.test.ts	should handle missing password field	PASSED	0.17	
create-user-integration.test.ts	should handle missing name field	PASSED	92.16	
create-user-integration.test.ts	should accept special characters in name	PASSED	86.11	
create-user-integration.test.ts	should handle very long email addresses	PASSED	92.45	
create-user-integration.test.ts	should handle email with plus addressing	PASSED	89.89	
create-user-integration.test.ts	should validate password requirements comprehensively	PASSED	361.87	
create-user-integration.test.ts	should return user with all expected fields	PASSED	93.38	
create-user-integration.test.ts	should create multiple users independently	PASSED	268.14	
create-user.test.ts	should create user successfully with valid email and password	PASSED		02.06

File / Suite	Cas de tests	Status	Duration (ms)	Error Details
create-user.test.ts	should return 400 when invalid role is provided	**PASSED**	0.55	
create-user.test.ts	should return 400 when email is invalid	**PASSED**	0.48	
create-user.test.ts	should return 400 when password is too short	**PASSED**		01.01
create-user.test.ts	should return 400 when password is missing uppercase letter	**PASSED**	0.70	
create-user.test.ts	should return 400 when password is missing special character	**PASSED**	0.24	
create-user.test.ts	should hash password with bcrypt before creating user	**PASSED**	0.28	
create-user.test.ts	should not return password in response	**PASSED**	0.44	
create-user.test.ts	should return 500 when database error occurs	**PASSED**	0.38	
create-user.test.ts	should handle bcrypt hash error gracefully	**PASSED**	0.15	
create-user.test.ts	should accept ADMIN role when valid	**PASSED**	0.17	
delete-user-integration.test.ts	should delete user successfully	**PASSED**	114.31	
delete-user-integration.test.ts	should return confirmation message on successful deletion	**PASSED**	117.89	
delete-user-integration.test.ts	should access req.user set by userIsAuth middleware	**PASSED**	99.08	
delete-user-integration.test.ts	should delete only the authenticated user	**PASSED**	161.44	
delete-user-integration.test.ts	should delete user with ADMIN role	**PASSED**	184.06	
delete-user-integration.test.ts	should handle idempotent delete (delete already deleted user)	**PASSED**	80.90	
delete-user-integration.test.ts	should return 500 on database error	**PASSED**	87.24	
delete-user-integration.test.ts	should handle missing req.user gracefully	**PASSED**	85.86	
delete-user-integration.test.ts	should handle null req.user.id gracefully	**PASSED**	84.04	
delete-user-integration.test.ts	should work correctly with different user IDs	**PASSED**	195.91	
delete-user-integration.test.ts	should delete user and remove from database immediately	**PASSED**	79.76	
delete-user-integration.test.ts	should respond with success after deletion	**PASSED**	74.42	
delete-user-integration.test.ts	should delete user even if deletion takes time	**PASSED**	125.85	
delete-user-integration.test.ts	should handle invalid user ID format gracefully	**PASSED**	73.52	
delete-user.test.ts	should delete user successfully	**PASSED**	1.62	
delete-user.test.ts	should return 500 when user not found	**PASSED**	0.43	
delete-user.test.ts	should return 500 when database delete error occurs	**PASSED**	0.30	
delete-user.test.ts	should handle missing user id parameter	**PASSED**	0.56	

File / Suite	Cas de tests	Status	Duration (ms)	Error Details
delete-user.test.ts	should return deleted user data after successful deletion	**PASSED**	0.91	
delete-user.test.ts	should return 401 when authorization header is missing	**PASSED**	0.35	
delete-user.test.ts	should return 401 when bearer token format is invalid	**PASSED**	0.19	
delete-user.test.ts	should return 401 when token is missing after Bearer	**PASSED**	0.13	
delete-user.test.ts	should return 401 when token is invalid	**PASSED**	0.32	
delete-user.test.ts	should return 403 when user is not admin	**PASSED**	0.22	
get-user-integration.test.ts	should get current user profile with valid auth token	**PASSED**	130.54	
get-user-integration.test.ts	should return 404 when user not found in database	**PASSED**	88.99	
get-user-integration.test.ts	should handle missing user.id in request	**PASSED**	95.65	
get-user-integration.test.ts	should return 500 on database error	**PASSED**	88.12	
get-user-integration.test.ts	should successfully return different users with different tokens	**PASSED**	191.96	
get-user-integration.test.ts	should return all user fields in response	**PASSED**	81.24	
get-user-integration.test.ts	should work with both USER and ADMIN roles	**PASSED**	183.08	
get-user-integration.test.ts	should return all users when called by admin with valid token	**PASSED**	344.62	
get-user-integration.test.ts	should return array of users with all fields	**PASSED**	335.62	
get-user-integration.test.ts	should include test users in the list	**PASSED**	345.66	
get-user-integration.test.ts	should handle empty database gracefully	**PASSED**	331.92	
get-user-integration.test.ts	should return 500 on database error	**PASSED**	349.10	
get-user-integration.test.ts	should differentiate between USER and ADMIN in returned list	**PASSED**	355.37	
get-user-integration.test.ts	should handle multiple admin requests	**PASSED**	349.87	
get-user-integration.test.ts	should use req.user set by userIsAuth middleware for getUser	**PASSED**	85.62	
get-user-integration.test.ts	should access req.user.id property in getUser	**PASSED**	90.17	
get-user-integration.test.ts	should accept any user role for getUser (USER or ADMIN)	**PASSED**	167.62	
get-user.test.ts	should return user when user is found	**PASSED**	4.16	
get-user.test.ts	should return 404 when user is not found	**PASSED**	1.15	
get-user.test.ts	should return 500 when database error occurs	**PASSED**	1.36	
get-user.test.ts	should handle undefined user id gracefully	**PASSED**	1.00	
get-user.test.ts	should handle missing user object in request	**PASSED**	0.27	

File / Suite	Cas de tests	Status	Duration (ms)	Error Details
get-user.test.ts	should return 401 when authorization header is missing	PASSED	0.28	
get-user.test.ts	should return 401 when bearer token format is invalid	PASSED	0.20	
get-user.test.ts	should return 401 when token is missing after Bearer	PASSED	0.14	
get-user.test.ts	should return 401 when token is invalid	PASSED	0.33	
get-user.test.ts	should return all users when query is successful	PASSED	0.23	
get-user.test.ts	should return empty array when no users exist	PASSED	0.10	
get-user.test.ts	should return 500 when database error occurs	PASSED	0.12	
get-user.test.ts	should return 401 when authorization header is missing	PASSED	0.18	
get-user.test.ts	should return 401 when bearer token format is invalid	PASSED	0.12	
get-user.test.ts	should return 401 when token is missing after Bearer	PASSED	0.11	
get-user.test.ts	should return 401 when token is invalid	PASSED	0.14	
get-user.test.ts	should return 403 when user is not admin	PASSED	0.20	
update-user-integration.test.ts	should update user email successfully	PASSED	131.18	
update-user-integration.test.ts	should update user name successfully	PASSED	89.53	
update-user-integration.test.ts	should update user password successfully	PASSED	213.97	
update-user-integration.test.ts	should update multiple fields at once	PASSED	171.83	
update-user-integration.test.ts	should update with empty body (no changes)	PASSED	84.03	
update-user-integration.test.ts	should access req.user set by userIsAuth middleware	PASSED	82.74	
update-user-integration.test.ts	should not expose password in response	PASSED	172.03	
update-user-integration.test.ts	should maintain user role during update	PASSED	86.84	
update-user-integration.test.ts	should maintain timestamps correctly	PASSED	83.85	
update-user-integration.test.ts	should handle update for different users independently	PASSED	205.57	
update-user-integration.test.ts	should handle update with special characters in name	PASSED	79.21	
update-user-integration.test.ts	should handle update with very long name	PASSED	74.44	
update-user-integration.test.ts	should handle update with very long email	PASSED	74.88	
update-user-integration.test.ts	should handle update with null values (set to undefined)	PASSED	78.65	
update-user-integration.test.ts	should return 500 on database error	PASSED	74.61	
update-user-integration.test.ts	should return all user fields in response	PASSED	71.44	

File / Suite	Cas de tests	Status	Duration (ms)	Error Details
update-user-integration.test.ts	should handle consecutive updates	**PASSED**	73.83	
update-user-integration.test.ts	should work with ADMIN role users	**PASSED**	142.25	
update-user.test.ts	should update user successfully with all fields	**PASSED**	3.55	
update-user.test.ts	should update user with only email	**PASSED**	0.49	
update-user.test.ts	should update user with only name	**PASSED**	0.43	
update-user.test.ts	should update user with only password	**PASSED**	0.41	
update-user.test.ts	should return 500 when database update error occurs	**PASSED**	0.55	
update-user.test.ts	should handle missing user id gracefully	**PASSED**	0.16	
update-user.test.ts	should handle empty request body	**PASSED**	0.17	
update-user.test.ts	should update user email and name together	**PASSED**	0.12	
update-user.test.ts	should return 401 when authorization header is missing	**PASSED**	0.42	
update-user.test.ts	should return 401 when bearer token format is invalid	**PASSED**	0.15	
update-user.test.ts	should return 401 when token is missing after Bearer	**PASSED**	0.13	
update-user.test.ts	should return 401 when token is invalid	**PASSED**	0.15	