

Meeting Room Reservation System

Test Decomposition of the Product Brief

Prepared 18 September 2026 · Scope: create, view, edit, cancel, persistence · Editing assumed in scope

1. Source brief and scope

1.1 Product brief (verbatim)

Employees can reserve meeting rooms, view their reservations, and cancel their own reservations. Each reservation specifies a room, date, start time, end time, and attendee count. Rooms have a maximum capacity. Overlapping reservations for the same room are prohibited. Reservations must remain available after the user signs out and returns.

1.2 Scope assumptions

- Editing an existing reservation is absent from the brief. It is treated as in scope here, with all five fields editable, pending decision D14.
- Actors: employee A, employee B, unauthenticated user U. The brief defines no administrative or override role (D22).
- Mutations: create, edit, cancel. Reservation lifecycle: nonexistent → active → cancelled. Edit keeps the reservation active and, unless D17 decides otherwise, keeps its identity. This bounds the state space.
- Rooms exist with a capacity, but the brief defines no room administration. Room changes (capacity reduction, removal) are noted as risks to invariants, not as tested operations.

1.3 Conventions

- T-n: testable statement. I-n: invariant. D-n: unresolved decision.
- Basis column. **Stated**: the brief supports the statement directly. **Inferred**: the brief only implies it. **Assumed scope**: follows from treating edit as in scope. Inferred and assumed statements must be confirmed before they become pass/fail criteria rather than assumptions.

2. Testable statements

2.1 Access

ID	Statement	Basis
T1	Unauthenticated requests to create, view, edit, or cancel are rejected.	Inferred — “Employees can...”

2.2 Create

ID	Statement	Basis
T2	An authenticated employee can create a reservation specifying room, date, start time, end time, and attendee count.	Stated
T3	A create with any of the five fields missing is rejected, and no record is created.	Inferred — “specifies” read as required
T4	A create with start time $\geq$ end time is rejected.	Inferred — ordering is never stated (D5)
T5	A create with attendee count greater than the room's capacity is rejected.	Inferred — the brief says rooms have a capacity; it never says reservations are checked against it (D8)
T6	A create with attendee count equal to the room's capacity is accepted.	Inferred — boundary of T5; $\leq$ vs $<$ unresolved (D8)
T7	A create whose interval overlaps an active reservation for the same room and date is rejected, regardless of who owns the existing reservation, including the requester.	Stated
T8	Same interval, different room: accepted.	Stated — prohibition is scoped to the same room
T9	Same interval and room, different date: accepted.	Stated — intervals on different dates do not overlap
T10	A create in a slot vacated by a cancellation is accepted.	Inferred — the brief never says cancelled reservations stop blocking
T11	A rejected create leaves no partial or hidden record.	Inferred — atomicity is not stated

2.3 View

ID	Statement	Basis
T12	The owner's view lists every active reservation they own, with all five fields intact.	Stated
T13	The view excludes reservations the user does not own.	Inferred — “their reservations” (D11)

2.4 Edit

ID	Statement	Basis
T14	The owner can edit any of the five fields of an active reservation they own.	Assumed scope — D14
T15	A non-owner cannot edit another employee's reservation. Enforced server-side, including requests that reference the reservation by ID.	Assumed scope — parallel to cancel (T27)
T16	The post-edit record must satisfy every create rule. Each create rejection case (T3–T7 and the field boundaries in 4.2) replayed through edit is rejected, with the original untouched.	Assumed scope — D19
T17	The overlap check excludes the reservation under edit: shifting or resizing within its own current interval succeeds.	Assumed scope
T18	The overlap check still includes the owner's other reservations. Self-exclusion is by reservation ID, not by owner.	Assumed scope
T19	Changing room re-checks the new room's capacity and the new room's conflicts. The old slot is released only when the edit commits.	Assumed scope — D18
T20	A failed edit of any kind leaves the reservation exactly as it was: same room, same slot, still active.	Assumed scope — D18
T21	Edit cannot change ownership.	Assumed scope — D14
T22	A no-op edit (all values unchanged) succeeds with no side effects and no self-conflict error.	Assumed scope
T23	The same edit submitted twice is a no-op on the second submission — not an error, not a duplicate.	Assumed scope
T24	Editing a cancelled reservation is rejected.	Inferred — the alternative is reinstatement (D15)
T25	The edit form pre-populates from the targeted reservation's stored values, not an adjacent one.	Assumed scope

2.5 Cancel

ID	Statement	Basis
T26	The owner can cancel a reservation they own; it is then absent from their view.	Stated
T27	A non-owner cannot cancel another employee's reservation. Enforced server-side, not by hiding the control.	Stated — “their own”
T28	Cancel affects exactly the targeted reservation.	Inferred — implicit in per-reservation cancel

2.6 Persistence

ID	Statement	Basis
T29	After sign-out and sign-in, the owner's reservations, including any edited values, are present and unchanged.	Stated
T30	The same holds from a different browser or device with no local state.	Inferred — “remain available” is hollow if it holds on one client only; server-side storage is not stated (D26)

3. Invariants

Each must hold after every operation, successful or not.

ID	Invariant
I1	No two active reservations for the same room have overlapping intervals. The invariant most likely to break under concurrency.
I2	Every active reservation's attendee count is $\leq$ its room's current capacity. Breakable if capacity is editable (D10).
I3	Every reservation is well-formed: existing room, valid date, start < end, attendee count within the permitted range (D9).
I4	Every reservation has exactly one owner, the owner is an employee, and the owner never changes.
I5	A user's view equals the set of active reservations they own — nothing missing, nothing extra, nothing stale.
I6	Cancelled reservations never block new ones.
I7	Only the owner can modify or remove a reservation from the active set. The brief defines no other role (D22).
I8	Sign-out, sign-in, and session expiry never change the reservation set.
I9	Create, edit, and cancel are atomic: full effect or none. An edit is never observable mid-flight — a concurrent reader sees old values or new values, never an absent reservation, never a mix of fields, never a state that violates I1–I3.
I10	The active-reservation count changes by exactly +1 on a successful create, -1 on a successful cancel, and 0 in every other case, including every edit and every failed operation. A cheap automation oracle: evaluate it after every step.
I11	The reservation ID is immutable across edits, provided the design keeps IDs stable (D17).
I12	An edit's overlap set is the active reservations for the target room and date, minus the one being edited.

4. Test combinations

4.1 Interval placement on create

Existing reservation: room R, date D, 09:00–10:00, owned by A. New request as below.

New request	Expected
08:00–08:30	Accept
08:00–09:00	Unresolved — touching (D1)
08:30–09:30	Reject
09:00–10:00	Reject
09:15–09:45	Reject
08:00–11:00	Reject
09:30–10:30	Reject
10:00–11:00	Unresolved — touching (D1)
10:30–11:00	Accept
09:00–10:00, room S	Accept
09:00–10:00, date D+1	Accept
09:00–10:00, by A herself	Reject (T7)
09:00–10:00, after A cancels	Accept (T10)

4.2 Field boundaries

- Attendee count against room capacity N: 0, 1, N–1, N, N+1, negative, non-integer, blank, overflow-sized value.
- Times: start = end; start > end; 00:00 start; 23:59 end; 24:00; interval crossing midnight; sub-minute precision if the field permits.
- Dates: past date; today with start already passed; today with start passed but end not; 29 February and 30 February; month 13; the DST transition day if the deployment time zone observes it.
- All fields: leading and trailing whitespace; injection strings.
- Every case above replayed as an edit against an existing valid reservation (T16).

4.3 Checks that must fire per edited field

Changed field	Must re-check
attendee count	Capacity of the current room
start / end	Ordering; overlap, with the reservation itself excluded

Changed field	Must re-check
date	Overlap on the new date
room	Capacity of the new room; overlap in the new room
nothing	Nothing — the edit succeeds (T22)

Defects this matrix catches: only the changed field is validated (a room change skips the capacity check); the overlap check includes the reservation itself (a five-minute shift is rejected); self-exclusion by owner rather than by ID (T18 fails).

4.4 Interval placement under edit

Own reservation X: room R, date D, 09:00–10:00. Another user's reservation Y: room R, date D, 11:00–12:00. Edit X to the values below.

New X	Expected
09:15–10:15	Accept — overlaps self only (T17)
08:30–09:30	Accept
09:00–11:00	Unresolved — touches Y (D1)
09:00–11:30	Reject; X unchanged
11:00–12:00	Reject; X unchanged
10:30–11:30	Reject
09:00–10:00 (no change)	Accept (T22)
Room S, free, capacity $\geq$ count	Accept; B can now book R 09:00–10:00 (T19)
Room S, capacity $<$ count	Reject; X stays in R (T20)
Room S with Z at 09:30–10:30	Reject; X stays in R (T20)
Date D+1, R free	Accept
Slot of A's own other reservation W	Reject (T18)

4.5 Two-user scenarios

- A creates; B attempts each row of 4.1. Check whether B's rejection message discloses A's identity (D13).
- A creates; B views. Is A's reservation visible (D11)?
- A creates; B attempts to cancel via the UI (control should be absent) and via a direct request carrying A's reservation ID. Expected: rejected; reservation still active; A's view unchanged. Repeat for edit (T15).
- A creates, then cancels; B books the same slot. Expected: accepted.

- A creates on device 1; A signs in on device 2: present. A cancels on device 2; device 1 refreshes: gone. Repeat with an edit in place of the cancel.
- A edits X from room R to room S; B creates R at X's old slot. Expected: accepted.

4.6 Concurrency

Highest-value group. Check-then-insert races are the most probable defect class in this brief, and edit's release-then-acquire window is where they fail hardest — especially if edit is implemented as cancel + recreate (D17).

- A and B submit overlapping creates for the same room within milliseconds. Expected: exactly one succeeds. Repeat many times; verify I1 against the data store directly, not through the UI.
- Same user double-clicks submit, or submits from two tabs. Expected: exactly one record.
- A cancels while B's create for the same slot is in flight. Either order is acceptable; I1 and I10 hold.
- The same reservation cancelled twice concurrently. Expected: count changes by -1, not -2; no half-state.
- A edits X to R 14:00–15:00 while B creates R 14:00–15:00. Expected: exactly one holds the slot; if A loses, X is still at 09:00–10:00.
- A edits X to a slot that will conflict while B creates R 09:00–10:00. B may win X's old slot only if A's edit committed. Failure mode: cancel releases 09:00–10:00, recreate fails, B has taken the slot, X is gone.
- Lost update: A opens the edit form for X in two tabs. Tab 1 changes the time and saves. Tab 2 (stale) changes attendee count and saves. Acceptable: tab 2 is rejected as stale, or the result carries both changes. Tab 2 silently reverting the time is the defect. This also exposes full-record vs delta payload semantics (D16).
- Edit vs cancel, same user, two tabs. Cancel then edit: edit rejected, no resurrection. Edit then cancel: cancelled. Both in flight: one consistent terminal state.
- Two edits attempting to swap the slots of A's own reservations X and W. The first must fail on T18; no transactional swap is specified.
- Poll X via the API continuously while edits run. Expected: never absent, never partially updated (I9).

4.7 Authorization

- B cancels or edits X by ID. Expected: rejected; X unchanged.
- Mass assignment: A's edit payload includes owner, ID, or creation-timestamp fields. Expected: ignored or rejected, never applied (T21, I4).
- Body ID differs from URL ID. Expected: only the authorized target changes, or the request is rejected.
- Edit and cancel enforce the same owner-only boundary.
- U (never authenticated) calls the create, view, edit, and cancel endpoints directly. Expected: rejected (T1).

4.8 Session

- Create → sign out → sign in on the same browser: present. Then clear cookies and local storage → sign in: still present. The second step separates server persistence from a local cache that would pass T29 alone (T30).
- Sign out → browser back. The view must not render from cache; API calls fail.
- Two tabs; sign out in one; create or edit in the other. Expected: rejected; no record or change.
- Session expires with the create or edit form open → submit. Expected: clean rejection; no partial record; original intact.
- A signs out; B signs in on the same browser. Expected: none of A's data visible.

4.9 Repeated and degenerate operations

- Cancel twice sequentially. Expected: second is a clean no-op or an explicit error; count unchanged.
- Cancel or edit a nonexistent or malformed ID.
- Cancel or edit a reservation that has ended, or one in progress (D6, D15).
- Recreate an identical reservation immediately after cancelling it. Soft-delete implementations sometimes collide on unique constraints.
- No-op edit (T22); identical edit submitted twice (T23).

4.10 Multi-step sequences

- create → view: present with correct fields.
- create → cancel → view: absent → create same slot: accepted.
- create → edit → sign out → sign in → then a different device: edited values.
- create → edit (fails) → view: original intact.
- create → edit → edit → cancel → recreate the original slot: accepted.
- Extend an in-progress reservation (end 10:00 → 10:30, at 09:45). Into free time: per D15. Into an adjacent booking: rejected.
- Shorten an in-progress reservation to end now. Expected: remainder freed, if permitted (D15).

4.11 Room and volume

- Room with capacity 1; capacity 0 if such a room can exist.
- Capacity reduced below existing bookings → I2 (D10).
- Room removed with active reservations → orphan check (D10).
- User with hundreds of reservations → view completeness, ordering, pagination (D12).

5. Unresolved decisions

Decisions the brief does not make. Each affects the expected result of at least one test above.

5.1 Time semantics

ID	Decision
D1	Touching intervals (one ends exactly when the next starts): overlap or not? Closed vs half-open intervals.
D2	Time granularity and rounding.
D3	Time zone governing the date and times — server, user, or the room's site. Daylight-saving handling.
D4	Reservations crossing midnight. The single date field suggests not. Behaviour when end < start: error, or next day?
D5	Zero-length reservations; minimum and maximum duration; how far ahead a reservation may be made.
D6	Past dates and times: bookable, cancellable, editable? Treatment of in-progress reservations.
D7	Operating hours, weekends, holidays.

5.2 Capacity and attendees

ID	Decision
D8	Is capacity enforced at all, or informational? Is the comparison $\leq$ or $<$?
D9	Attendee count range. Is 0 valid? Does the count include the reserver? Required or optional?
D10	Effect of a capacity reduction or room removal on existing reservations.

5.3 View

ID	Decision
D11	Own reservations only, or all bookings per room? If own-only, how does a user find a free slot?
D12	Are past and cancelled reservations shown? Sort order, filtering, pagination.
D13	Does a conflict error disclose who holds the slot?

5.4 Edit

ID	Decision
D14	Editable fields: all five? Room? Owner (transfer)?
D15	Editable states: active only, or also cancelled (reinstatement), past, in-progress? Minimum notice before start?
D16	Concurrency control: last-write-wins vs version check. Full-record vs partial payload — does an omitted field mean unchanged or cleared?
D17	Implementation: in-place update vs cancel + recreate. Is the ID stable? Determines the audit trail, deep links, and whether I9 is achievable.

ID	Decision
D18	Does a failed edit retain the original slot? It should; confirm it is a stated requirement rather than an assumption.
D19	Do all create constraints apply to edits — booking horizon and past-time rules evaluated against the new times?
D20	Edit history and audit; notifications; limits on edit frequency or count.

5.5 Cancel and lifecycle

ID	Decision
D21	Hard vs soft delete; audit trail; confirmation and undo.
D22	Any administrative or override role. None is defined.

5.6 Identity and ownership

ID	Decision
D23	Who counts as an employee; authentication mechanism; session lifetime; concurrent sessions.
D24	Booking on behalf of others; ownership transfer.
D25	May one employee hold several rooms at the same time? Per-user or per-room quotas?

5.7 Persistence

ID	Decision
D26	Server-side storage is implied but not stated. Across devices? Across server restart? Retention period? Reservations belonging to departed employees?

5.8 Concurrency and errors

ID	Decision
D27	Resolution policy under simultaneous submission. First committed wins is assumed, not stated.
D28	Does a rejection give a distinct reason — conflict vs capacity vs validation vs authorization?

5.9 Silent scope

ID	Decision
D29	Confirm out of scope: recurring reservations, notifications, room attributes beyond capacity, room administration, reporting.