

The scheduling conflicts checklist

Run it against your own system. 24 questions, written for the person who answers for the schedule rather than for the person who wrote the code.

Answer each one yes or no. A no is not a defect on its own. A no you did not know about is, because every one of these turns into somebody's morning eventually, usually with two people standing in a room that was only ever going to hold one of them.

It comes out of building booking and scheduling engines where people, rooms and equipment are all finite, and where a booking can also move money. The questions apply to any system that hands out time.

Double booking

☐ Can the same person be booked into two places at the same time?

It sounds like the first thing anyone would check, and it usually is, inside one screen. The test is whether the system refuses it, or whether it relies on the person making the booking to notice.

☐ Is that check made across every part of the system that books people, or only within each part?

Training, shifts, appointments and site visits often live in separate modules, and each one checks only its own bookings. Every module can report, correctly by its own lights, that nothing clashed, while one person is booked into three places at once.

☐ When a resource other than the person is required, is it checked too?

A free practitioner in an occupied room is not a booking. Neither is a free room when the equipment the session needs is already in use. If the system only reads the person's diary, the clash moves to the moment somebody opens the door.

☐ Are all of a booking's resources checked for the same slot together?

Checking the person, then the room, then the equipment one after another, with gaps in between, lets another booking slip into the gap. The answer has to be about all of them at the same instant.

☐ Can two people booking at the same moment take the same slot?

Two receptionists, or a receptionist and a customer online, both see the last slot as free and both press book. If the check and the write are separate steps, both succeed. Only the database can hold that line reliably.

Availability

☐ Does availability account for leave, absence and shifts, or only for existing bookings?

A person on leave is not free, however empty their diary looks. A system that only knows about other bookings will happily book somebody who is not at work that day, and the clash surfaces on the morning itself.

☐ Is availability worked out from rules, rather than typed into a grid?

Opening hours, working patterns, buffers and capacity are rules. A grid someone fills in by hand is correct on the day it is filled in and drifts from then on.

☐ Are the gaps between bookings part of the rule, rather than a habit?

Cleaning a room, resetting equipment, travelling between sites. If the gap lives only in the head of whoever books, the first new member of staff will book back to back.

☐ If availability changes after a booking exists, does anything notice?

A shift moves, a room is closed for repairs, somebody calls in sick. Every booking that relied on the old availability has to be looked at again. A check that only runs at booking time answers yesterday's question.

☐ If anyone books across a timezone, whose clock wins?

A booking stored without its timezone means different things to the person who made it and the person who turns up. Decide where time is stored, where it is displayed, and what happens when the clocks change, and write it down.

Refusals and overrides

☐ Is an impossible booking refused before it exists, rather than reported afterwards?

A clash report that runs overnight finds the problem after the confirmations have gone out and people have planned around them. Refusing up front is gentler in practice, because nobody has to unpick a promise.

☐ When a booking is refused, does the system explain why and offer the nearest alternative?

"Not available" sends the person booking back to try slots one at a time, sometimes with a customer waiting on the phone. Naming what clashed, and the nearest slot where everything is free, turns a dead end into a choice.

☐ Who is allowed to override a clash, and is the override recorded?

Sometimes a clash is the right call: an emergency, a deliberate double booking for a handover. Overrides should exist, should be limited to the people who can justify them, and should leave a record that says who and why.

- ☐ **Does the system tell the people affected by a booking, rather than leaving it to whoever made it?**

A booking that nobody was told about is a no-show waiting to happen. The person being booked, and anyone whose resource was taken, should hear it from the system, at the moment it happens.

Changes

- ☐ **When someone cancels or reschedules, what happens to everything that depended on it, including the money?**

A booking rarely stands alone. It may carry an invoice, a deposit, a commission owed to somebody, a room held, or a follow-up scheduled from it. Cancelling the diary entry and leaving the rest in place is the most common half-done change.

- ☐ **Is a cancellation recorded, rather than deleted?**

Deleting a cancelled booking deletes the reason everything attached to it existed. Last month's figures change after they were reported, and nobody can say why. A cancellation is an event, and it belongs in the record.

- ☐ **Is a reschedule checked as strictly as a new booking?**

Moving a booking is booking it again at a new time. If the move skips the checks a new booking would face, rescheduling becomes the quiet way to create the clash the system otherwise refuses.

- ☐ **Does a recurring booking check every occurrence, not only the first?**

A weekly slot that is free this week may collide with a holiday in March or a shift pattern that changes next quarter. Checking only the first occurrence creates a clash that waits months to appear.

- ☐ **Does a no-show have a consequence the system knows about?**

A missed appointment may carry a fee, release a resource, or count against a policy. If the system cannot record a no-show as distinct from a cancellation or an attendance, every one of those becomes somebody's manual follow-up.

- ☐ **Is a booking's state explicit, rather than inferred?**

Held, confirmed, attended, cancelled, no-show. If the state of a booking has to be worked out from which fields happen to be filled in, two reports will work it out differently.

The record

- ☐ **Can you reconstruct who booked what, when, and who changed it afterwards?**

When a customer says they were booked for ten and the system says eleven, the answer has to come from a history, not from memory. Every change to a booking should record who made it and when.

☐ **Does the history survive the booking being changed or cancelled?**

A record that is edited in place keeps only the latest version of the truth. The question that matters is usually what the booking said before somebody changed it.

☐ **Does the money agree with the calendar at month end without anyone reconciling them by hand?**

Where a booking carries a fee, the calendar and the ledger describe the same events. If somebody spends the last day of each month matching one against the other, they disagree the rest of the time.

☐ **If two bookings collided tomorrow, would the system tell you before the people involved did?**

This is the one question that matters. A scheduling problem is rarely discovered by the system that has it. It is discovered by two people at one door, a customer who was not expected, or a resource that was promised twice.

Where this came from

We build booking and scheduling engines, and this list is what we run against an existing one before quoting the work. It is useful on its own, and you are welcome to use it without speaking to us.