Time sheets are deceptively simple. A handful of rows, start and stop times, maybe a job code, then a signature or an approval click. Payroll, on the other hand, is where all those small decisions either add up cleanly or turn into painful corrections, missed compliance steps, and employee dissatisfaction.

Over the years, I have learned that the real workflow is not a single handoff from “timesheets” to “payroll.” It is a chain of custody. Each link has a purpose: capture time accurately, validate it consistently, apply the right rules, and produce pay with confidence. When the workflow is clear, payroll feels like an output. When it is not, payroll feels like a rescue operation.

Below is how the process typically works in practical terms, including the edge cases that tend to show up right before payday.

The real starting point is the calendar, not the spreadsheet

Most people assume timekeeping begins when an employee fills out a timesheet. In reality, the workflow begins earlier, with the pay period structure and the rules tied to it. Pay periods vary by organization, but the principle is always the same: define what “on time” means before anyone records hours.

That definition touches several things at once.

For hourly teams, it affects cutoffs for submission and approval. If your payroll provider or internal payroll system expects timesheets by 5:00 p.m. On a specific day, “lateness” has consequences. Those consequences are not only about processing time. They impact overtime calculations, eligibility for differential pay, and compliance reporting that some organizations treat as non negotiable.

For salaried teams, you still need consistency. Salaried pay may not change week to week, but time records often drive exceptions. Think of paid leave conversions, unpaid time deductions when policy allows, or adjustments after a reallocation of work hours.

When the calendar is set incorrectly, everything after it becomes messy. I have seen cases where a company’s pay period ended on a holiday, and managers assumed they could submit adjustments “next business day.” Payroll systems do not share that assumption. They treat the period boundaries like physics.

Step one: capture time in a form your payroll rules can use

Time capture can happen through paper timesheets, spreadsheets, or time and attendance software. Even with software, the core concept remains: you need data that can be interpreted reliably by payroll.

The most common fields are straightforward, but they must be consistent in format.

- employee identifier (name, employee ID, badge ID)
- date and time entries (start and end, or total hours)
- shift or job assignment (department, project code, cost center)
- adjustment reason when time is corrected

Where organizations stumble is not in the presence of fields, but in the meaning behind them. A few examples:

A worker may record “1:00” as a total hours number, while another enters “1:00 p.m.” as a clock time. Both are time related, but they do not behave the same in calculations. Someone may submit a timesheet using local time conventions, while payroll expects a standardized time zone or handles daylight saving changes differently.

Then there are the human details. Employees forget to clock out, managers approve based on memory rather than logs, and exceptions accumulate until someone tries to “clean it up” days later.

The best systems reduce the room for interpretation by making entries structured. The best teams also train people on how to correct mistakes, because corrections are not an afterthought in payroll. They are part of the process.

Step two: approve timesheets with accountability, not just speed

Approval is often treated as a mechanical step: the manager clicks “approve,” payroll gets access, and that is that. In practice, approval is where you prevent most payroll pain.

A strong approval workflow does more than rubber stamp. It enforces responsibility. It also makes sure the corrections are documented in the way payroll auditors and HR teams expect.

Here is the tension I have seen repeatedly: speed versus accuracy. When the deadline is close, managers feel pressure to approve quickly. Employees feel pressure to submit quickly. Both pressures push toward “good enough,” and payroll punishes that behavior.

A good approach is to make approval time part of the weekly rhythm, not a frantic last-minute scramble. That usually means you set internal checkpoints earlier than the formal payroll cutoff. The goal is simple, catch issues when they are still easy to fix.

When a timesheet has missing punches or unusual hours, approval is the moment to verify the exception. Sometimes the correct action is to request clarification from the employee. Sometimes it is to apply an adjustment using an established policy. Either way, the logic should be explicit.

Step three: validate data before payroll rules touch it

Once the data is approved, payroll teams typically run validation checks. This is where the workflow stops being subjective and starts being systematic.

Validation can range from automated checks in time and attendance software to manual review in a payroll staging area. The key is to catch errors that would either cause incorrect pay or create processing failures.

What counts as an error depends on your policy and payroll capabilities, but the categories tend to repeat across industries:

- Missing required fields, like job codes, pay types, or earning categories
- Invalid time formats or impossible ranges
- Entries outside permitted scheduling windows, which may indicate a punch error
- Duplication, such as the same time being imported twice due to a sync or mapping issue

One useful reality check I like to emphasize: treat payroll as an accounting system. If the data cannot balance into the expected structure, it should not be forced through. Forcing bad inputs rarely saves time. It usually delays the work into an even harder correction cycle.

To keep this stage effective, payroll teams often build a “review queue” of flagged exceptions. Those exceptions are then resolved through corrections, re-approvals, or mapped overrides.

Step four: map time entries to earnings, deductions, and rules

After validation, the payroll engine needs a consistent mapping from time entries to payroll components. In other words, your timesheet data must translate into earning types like regular hours, overtime, shift differential, paid breaks, or differentials tied to location.

This is where policy and configuration meet. Two organizations can collect similar time data and still produce different payroll results because the rules differ.

For example, consider overtime. In many settings, overtime depends on how hours are accumulated: weekly, daily, by jurisdiction, or by job classification. In some environments, overtime eligibility can be affected by exemptions or by how certain pay types are treated in the overtime calculation.

Even if your timesheet records only hours, payroll needs to decide which hours count toward overtime and which hours do not. That decision is rarely "obvious" from the timesheet alone. It depends on configuration.

The job code or department often controls those rules too. If a role earns overtime differently depending on work type, then the mapping must be correct.

This stage also covers deductions that may rely on time records, like unpaid leave calculations or certain attendance-based adjustments, depending on local law and company policy.

https://paystub.org/posts/payroll-statistics

When the mapping is wrong, the payroll output is wrong even if every time entry looks plausible. I have seen payroll systems import hours correctly and still miscalculate earnings because the job code mapping changed during a reorganization. The employees "worked the same," but payroll treated them as if they were doing different work.

Step five: handle adjustments like a controlled process

Adjustments are inevitable. Timesheets are human-generated. Clock-in systems fail. Employees change schedules midstream. Managers discover errors after the fact.

The question is not whether adjustments will happen. The question is whether you can make them without destabilizing payroll.

A controlled adjustment process usually includes three elements:

First, an adjustment needs a reason code or category. This turns "we fixed it" into traceable information. Second, adjustments should be restricted by timing. Some systems allow edits up to a cutoff, then require corrections in the next pay period. Third, adjustments should trigger re-validation and sometimes re-approval. If an employee changes hours, the manager approval should reflect that new reality.

A frequent edge case involves corrected time that changes overtime. That can cascade into multiple earnings lines. It can also affect tax withholding if the correction changes gross pay. The more you treat adjustments as casual edits, the more you risk inconsistent documentation.

In mature workflows, adjustments are tracked and audited. You can still move fast, but you do it with guardrails.

Step six: review payroll registers before submission

Once time is mapped and rules are applied, you get a payroll register or preview report. This is the last chance to catch "it will still pay but it is wrong."

The review is often split across HR, payroll, and finance. Some organizations also include department leaders for high-touch groups.

What people review varies, but there are recurring themes: unusual gross pay changes, missing earnings, and employees who did not get expected pay types.

This is also where you confirm that net pay aligns with expected funding and that deductions like benefits and garnishments were applied correctly.

If you are managing an annualized or variable pay scenario, the review must include pay period factors. For example, some organizations calculate certain accruals or benefits based on hours thresholds. Those thresholds can be affected by even small time changes.

A practical review does not require reading every line item for every employee. It requires focusing on patterns. When payroll spikes for one employee, you look at their time entries. When a group's overtime looks off, you check the mapping or policy settings.

The best payroll review habits are consistent, not complicated.

Here is a short checklist that many payroll teams find useful in the final pre-run window:

1. Verify no employees have missing time data for required earning types
2. Spot-check overtime and differential calculations against the policy rules
3. Confirm adjustment reasons and approvals exist for any out-of-cycle changes
4. Reconcile totals for hours and gross pay at the department or cost center level
5. Check for duplicated imports or unexpected pay type assignments

That list is short on purpose. The goal is to keep attention on the highest-leverage checks.

Step seven: finalize payroll and communicate outcomes

After payroll is approved for processing, the workflow moves into finalization. This includes generating pay files, finalizing tax calculations, and issuing payments.

Communication matters here. Employees often do not experience payroll as a complex workflow. They experience it as what hits their bank account and what the payroll statement shows.

If a timesheet correction was made after the cutover, the "where did that money go" question will land soon. Many payroll teams handle this by standardizing communication templates for corrections and by maintaining a list of exceptions for HR responses.

It is also common to address time discrepancies proactively. If an employee forgets to clock out, and policy leads to an estimated correction, they may later notice a mismatch between their recollection and payroll output. If you do not explain the rationale, you spend time rebuilding trust after the fact.

The workflow is not complete until employees understand their outcome or until HR can explain it accurately.

Step eight: reconcile after payroll runs

Reconciliation is where the workflow becomes sustainable. Even with careful validation, payroll runs sometimes surface issues late: a configuration mismatch, an export problem, a missing deduction record, or a time entry imported under the wrong pay period.

A well-run reconciliation process compares what you expected to process against what was actually processed.

It also makes sure corrections are properly captured. If you must reverse an error, the question is whether you fix it through an off-cycle adjustment, a retro pay process, or the next pay period.

Different organizations have different tools for retro processing. Some integrate it into payroll runs. Others manage it via manual adjustments.

Either way, reconciliation closes the loop. It helps you identify where issues originate, not just where they appear.

For example, if a particular manager's approvals repeatedly miss a category that affects overtime, the fix is not to correct payroll each time. The fix is to improve the manager's workflow or the system's prompts and training.

Where the workflow breaks: the edge cases that keep returning

Most payroll problems are not random. They are predictable. They show up in categories, and teams learn to guard against them.

One recurring category is cross-period work. An employee works late on the last day of a pay period and early on the next. If the system allocates time based on clock-out date or on time record date inconsistently, you can end up with partial hours in the wrong pay period.

Another category is midnight and time zone handling. Employees on overnight schedules or remote workers in different time zones expose weaknesses in how timestamps are stored and converted. If your configuration treats timesheets as local time but payroll as a centralized time, entries can shift unintentionally.

Then there is the "estimated time" problem. Policies sometimes allow estimation for missing punches. That estimation might be based on last known schedule, shift duration, or policy rules. Estimation helps operations, but it creates reconciliation needs. Employees will compare estimated time to their internal schedule expectations.

Finally, the workflow breaks when mappings change without downstream verification. Organizational changes, code renames, new job assignments, or HRIS updates can break the assumptions in payroll. You may not notice until payroll preview shows something odd.

The most mature teams treat configuration changes like software releases: test, verify, and document.

A look at the data pipeline: time sheet formats and integrations

Even though the question is about "from time sheets to payroll," the workflow is also about how data moves. If you use software for timekeeping and a separate system for payroll, integrations become part of the workflow.

Common integration approaches include scheduled imports, API syncs, and file-based transfers. Each has failure modes.

- Scheduled imports can miss a window if something delays completion.
- API syncs can partially fail if a specific record breaks the format.
- File-based transfers can map fields incorrectly if column order changes or headers are inconsistent.

The most practical way to protect against integration failures is to add reconciliation at the data level, not only at the payroll totals level. That means you confirm the number of records received, check for missing employee IDs, and verify that pay period tags align.

I prefer early detection. Catching a broken integration two days before payroll is painful but manageable. Catching it two hours before pay runs forces emergency changes that are hard to audit.

Trade-offs: accuracy versus speed, manual review versus automation

No workflow achieves perfection. There are always trade-offs.

If you aim for speed, you might allow late submissions and handle adjustments after payroll preview. That reduces operational friction but increases retro corrections and employee questions.

If you aim for accuracy, you might enforce strict cutoffs and require re-approval for any changes. That prevents surprises but can increase the number of employees who miss deadlines and then need support.

Automation can reduce manual burden, but it can also hide problems. When mapping rules are automated, a configuration mistake gets applied consistently and quickly. That means wrong payroll can be wrong at scale.

Manual review helps catch anomalies, but it is expensive and inconsistent across reviewers. People get tired, and the most subtle errors can slip through.

The best setups balance automation and human oversight based on risk. High risk areas, like overtime calculations or special earning types, get more review attention. Low risk areas, like standard regular hours with predictable schedules, can be processed with lighter review.

Practical tips that actually help

After enough pay periods, you start noticing which practices reduce recurring issues.

One is to keep timesheet training job-specific. Hourly workers need guidance on clock in and out, punch corrections, and how job codes work. Managers need guidance on approval standards, what to do when time looks off, and how to request corrections. If you train everyone the same way, people remember what they care about, not what payroll needs.

Another is to design your internal cutoffs. Many organizations set a payroll cutoff but also have an earlier internal “manager approval” deadline. That gives you buffer for validation and corrections.

A third is to maintain a living exceptions log. When employees ask why their pay changed, you can reference the exact reason and policy logic. When payroll configuration changes, you can tie it to the exceptions it affected.

And finally, keep your documentation tight and searchable. Payroll is stressful. In the moment, people reach for whatever they can find. If the workflow documentation is scattered across multiple places, you lose time when you need clarity most.

How to measure whether the workflow is working

You can feel whether a workflow is working because payroll errors tend to create chaos. But it is still worth measuring.

Look at the number of after-cutoff corrections, the volume of retro pay, the time it takes to resolve payroll exceptions, and the repeat rate of specific error types. A workflow that improves gradually will show fewer recurring issues over multiple pay periods, not just fewer errors in one cycle.

Employee feedback also matters. If employees repeatedly report that their hours look correct but pay looks wrong, your workflow may be failing in mapping, overtime rules, or communication.

Meanwhile, a finance partner might care about reconciliation stability. If payroll totals do not reconcile cleanly to what finance expects, you will spend time chasing differences every month.

When the workflow is healthy, time sheets become inputs, payroll becomes a predictable output, and corrections become rare events rather than a routine.

The bottom line: payroll accuracy is a workflow design problem

From time sheets to payroll is often described as a task. In practice, it is a workflow design problem, with policy decisions embedded in software configuration and operational discipline embedded in human behavior.

When you treat the process as a chain of custody, each step has a purpose. Capture time in a structured way, approve with accountability, validate before payroll rules calculate pay, map correctly to earnings and deductions, manage adjustments under control, and reconcile after runs to make the next cycle smoother.

Pay payroll reliably is not about finding one “right” system. It is about building a workflow where time data becomes trustworthy, rules are applied consistently, and surprises are caught early. That is what turns payroll from a recurring scramble into a dependable business process.