

Enterprise

How the lifecycle modules connect — hiring to workforce operations in one system.

- [The connected HR lifecycle](#)
- [Branches, departments and job positions: a multi-site company](#)
- [A job requisition becomes a candidate pipeline](#)
- [Interview feedback, an approved offer, and a new hire](#)
- [A draft payroll contract, reviewed and explicitly activated](#)
- [Attendance, a geo-fence, and leave for a distributed project team](#)
- [A first payroll run, reviewed and approved by someone else](#)
- [India payroll compliance: a monthly run and a governed correction](#)
- [From a company objective to an individual one: the OKR cascade](#)
- [Corporate learning: courses, enrolment and completion with Moodle](#)
- [Background verification: consent, review and audit \(mock provider\)](#)
- [Mitra recommends the next step; a person decides](#)
- [Payroll to accounting: a journal today, an ERP integration that is ready but not live](#)
- [Payments: a payout flow that is provider-ready, and a mock today](#)

The connected HR lifecycle

BlueOneHR is not a set of separate tools glued together: a hire, an offer, an employee record, a payroll contract, an objective and a background-verification case all reference the same underlying data, in one system. This page follows one hire through that connected lifecycle — candidate, interview, offer and acceptance, employee record, a payroll contract an HR leader explicitly activates, an individual objective, and a background-verification case on a mock provider — and says, at each step, exactly what is real product behaviour and what is illustrative or mocked for the recording.

Illustrative demonstration prepared using publicly available company context and synthetic data. This video does not represent a production deployment, customer endorsement, or use of real employee data.

<https://www.youtube-nocookie.com/embed/Fgg9vJvo7KE>

[The connected HR lifecycle: hiring to workforce operations](#)

The hire this page follows

Every screen below was recorded against a fresh, fictional company built for this walkthrough — Mumbai head office, a regional office, and a handful of departments spanning project delivery, MEP, EHS, tendering and estimation, HR and finance. Its name, its departments, its one example employee and every figure on screen are synthetic; none of it represents a real customer, a real headcount or a real payroll.

Candidate, interview and offer

A requisition's open position becomes a candidate in [Recruitment](#)'s pipeline board, moving through stages such as Applied, Initial and Hired. An interview can be scheduled against the candidate at any stage. An offer is a separate record: it carries a position, a salary template and an annual CTC, and the platform's own compensation engine computes the monthly Basic/HRA package and **freezes it with the offer** — the breakdown shown on an offer is read-only from that point on. The candidate accepts (or declines) the offer from their own link, without signing in to the rest of the platform.

The protected handoff: offer to employee

Accepting an offer does not, by itself, create an employee. Converting a candidate is a single, guarded action that checks, in order: the offer is **Accepted**, the offer is **effective** (accepted and — where a signature is required — signed), and the candidate's **onboarding hand-off** has been consumed. Any one of those being missing refuses the conversion outright, with a specific reason, rather than creating a half-complete employee. Converting a candidate that does pass every check creates, in one step: the [employee record](#), their work information (department, job position, reporting line), and a **draft** payroll contract built from the offer's frozen package.

A contract a person activates

A draft [payroll contract](#) does not pay anyone. Nothing in this chain auto-activates a contract — not the conversion step, and not Mitra. An HR leader reviews the draft against the offer it was built from and activates it explicitly; only an active contract with a positive wage is ever payable. That one explicit, human action is the point where "a person was hired" becomes "a person is on payroll."

Workforce operations continue from there

Once someone is an employee, the same record is visible to [Performance](#) (an individual objective can be set against them), and to a background-verification case, which in this recording used the platform's **mock** BGV provider — the case, its status and its audit trail are real BlueOneHR behaviour; no outside verification agency was contacted. Corporate learning assignment and ERP/payments integrations exist as real product surfaces but are shown only as capability boundaries in the video that goes with this page, not as completed third-party actions.

Where Mitra fits

[Mitra](#) opens alongside pages such as the Control Center and reads only what the signed-in person's own permissions allow, explaining rather than acting. It does not activate a payroll contract, approve an offer or convert a candidate. Every action described above — approving, sending, converting, activating — was taken by a named person, not by Mitra.

Related reading

- [How recruitment works](#)
- [The employee directory](#)
- [Contracts](#)
- [How performance works](#)
- [What Mitra is](#)

Branches, departments and job positions: a multi-site company

A company that works across more than one physical site — a head office and one or more regional offices — needs its structure to say so explicitly, not just as addresses on a page. This page walks through how BlueOneHR represents that: a company with branches, departments that span those branches, job positions within each department, and an employee list that shows exactly who sits where.

Illustrative demonstration prepared using publicly available company context and synthetic data. This page does not represent a production deployment, customer endorsement, or use of real employee data.

https://www.youtube-nocookie.com/embed/XJWc3_WQCKU

[Setting up a multi-site organisation \(HQ, regional office, project teams\)](#)

The company this page follows

This page continues the same illustrative demo structure as [The connected HR lifecycle](#): one fictional company, built to show BlueOneHR as one connected system — here, organised across a Mumbai head office and a Hyderabad regional office. Its name, its branches, its departments and every person shown are synthetic; none of it represents a real customer, a real headcount or a real site.

Branches

Settings → **Branches & Scope** → **Branches** lists every branch a company has — a code, a name and an Active/Inactive status for each. A company with one physical location may never need more than one row here; a company running a head office and a regional office keeps one row per site,

each independently editable.

Who sits where

The **Employee Assignments** tab on the same screen is the direct answer to "who is at which site": one row per employee, their branch, their company, whether that branch is their primary assignment, and the date it took effect. A project-driven company can read its own distribution straight off this table — who is based at the head office, and who is out at the regional site running delivery.

Departments and job positions

Branches describe *where*; departments and job positions describe *what*. The [Employees](#) screen's **Group By** control can group the same company by **Department** — every department a company uses, each with its own headcount — or by **Job Position**, which narrows further: a department such as Project Execution can hold several distinct positions (a Site Engineer and a Project Manager are both Project Execution, but not the same seat). Neither grouping is scoped to one branch; a department or a job position can have people at the head office and at a regional office at the same time.

A separate concept: Geo-Fence

BlueOneHR also has a **Geo-Fence** module, listed on its own in the sidebar next to **Branches & Scope**. A geo-fence is a physical location with coordinates and a radius, used to confirm an employee's attendance clock-in and clock-out are actually happening on site. It is a deliberately separate concept from a branch: a branch is the company's legal and reporting structure (which legal site an employee's record belongs to); a geo-fence is a real-world location constraint used by one specific feature, attendance. A company can have branches without ever configuring a geo-fence, and a geo-fence is not required to exist per branch.

Related reading

- [The connected HR lifecycle](#)
- [The employee directory](#)
- [How geo-fencing works](#)

A job requisition becomes a candidate pipeline

Before a vacancy is open to candidates, most companies want someone other than the hiring manager to sign off on it — a budget check, not just a staffing decision. This page walks through how BlueOneHR represents that: a job requisition, a second person approving it, the job opening that approval creates, and the real candidates that then move through a pipeline.

Illustrative demonstration prepared using publicly available company context and synthetic data. This page does not represent a production deployment, customer endorsement, or use of real employee data.

<https://www.youtube-nocookie.com/embed/h2U69kP0kS4>

[Recruitment: requisition to candidate pipeline](#)

The company this page follows

This page continues the same illustrative demo structure as [The connected HR lifecycle](#) and [Branches, departments and job positions](#): one fictional company, Meridian Interiors, organised across a Mumbai head office and a Hyderabad regional office. Its name, its roles and every person shown are synthetic; none of it represents a real customer, a real headcount or a real hiring decision.

Raising a requisition

Requisitions is its own area, separate from Recruitment itself: a job title, the department and job position it belongs to, a branch, a headcount, a needed-by date, an optional budget band and CTC range, and a justification that the approver reads. Raising one creates it as a **draft** — nothing is visible to candidates yet, and nothing is committed until it is submitted.

A maker cannot also be the checker

Submitting a requisition sends it into the company's approval chain, the same approvals engine used elsewhere in BlueOneHR. The product enforces maker-checker here directly: the person who raises a requisition is never allowed to also be its only possible approver — if nobody else could approve it, the platform refuses the submission outright rather than letting one person both ask for a hire and sign off on it. In practice that means two different people: one raises the requisition, a different one — with approval authority — decides it.

Approving creates the job opening

Once approved, BlueOneHR creates the job opening automatically and links it back to the requisition — nobody has to re-type the title, department or job position a second time. The opening starts **unpublished**: HR reviews it, and only then publishes it as an open vacancy that candidates can be added against.

Candidates and the pipeline

A published opening appears under **Open Jobs** with how many people have applied against its capacity. Candidates can be added — by name, email, mobile and job position — and from there each one sits in exactly one pipeline stage for that opening. Stage names are configurable per company, not fixed: a straightforward pipeline is **Applied** → **Initial** → **Interview** → **Hired**, and a card moves between stages one at a time as a recruiter's own call, not an automatic progression.

Duplicate candidates: an exact match, not a fuzzy one

Adding a candidate checks for another candidate in the same company with the same email and mobile number. This is an **exact match on both fields**, not an approximate or "looks similar" comparison, and it never blocks the save — it is reported as a warning for the recruiter to judge, not a wall that stops data entry.

Related reading

- [The connected HR lifecycle](#)
- [Branches, departments and job positions](#)
- [How recruitment works](#)

Interview feedback, an approved offer, and a new hire

Continuing from [A job requisition becomes a candidate pipeline](#): a candidate who has reached the Interview stage still needs a documented evaluation, a decided offer, and finally a converted employee record before any of this becomes a real hire. This page walks through those three real screens.

Illustrative demonstration prepared using publicly available company context and synthetic data. This page does not represent a production deployment, customer endorsement, or use of real employee data.

<https://www.youtube-nocookie.com/embed/JzB6quZFukA>

[Interview feedback, an approved offer, and a new hire](#)

The company this page follows

Same illustrative demo structure as [The connected HR lifecycle](#), [Branches, departments and job positions](#) and [A job requisition becomes a candidate pipeline](#): one fictional company, Meridian Interiors. Its name, its roles and every person shown are synthetic; none of it represents a real customer, a real headcount or a real hiring decision.

Interview feedback is a real, separate screen

Each interview has its own scorecard: an interviewer records an overall rating, a Hire / No Hire / Hold recommendation and free-text comments, submitted once and then locked — it cannot be

silently edited afterward. Until every assigned interviewer has submitted, each one sees only their own scorecard; a recruiter or HR, scoped to the company, sees the live partial state at any time. Once the panel is complete, everyone sees the full panel summary: how many submitted, the recommendation counts, and the average overall score.

A panel's recommendation does not gate the move to Hired

Feedback is informational, not a lock. Moving a candidate to the Hired stage is a recruiter/HR action — the same authority check used everywhere else candidates are moved between stages — and it does not require a complete panel or a particular recommendation first. A recruiter can move a candidate to Hired with the panel still incomplete, or wait for every scorecard first; the product does not force either choice.

An offer is proposed, approved by someone else, then sent

Raising an offer is a separate action from deciding on it. One person creates the offer against a salary template and an annual CTC — the gross-to-CTC breakup is computed by the compensation engine, never typed in by hand — and submits it, which freezes its terms. A different person, with approval authority, then approves or rejects it; the same maker-checker rule used for requisitions applies here too. Only after approval can the offer be sent: a one-time, token-based link the candidate opens without logging in, to accept, decline, or ask for more time.

Accepting starts, rather than finishes, onboarding

A candidate's acceptance does not convert them into an employee by itself. It requests an onboarding hand-off, which a company-scoped user then explicitly starts (consumes). Only with both an accepted, effective offer **and** a started hand-off does **Convert to employee** become available on that candidate's row — the same protected, single conversion path used for every hire, which refuses outright if either condition is missing.

What conversion actually creates

Converting a candidate creates the employee record — department, job position and (when the hire came through a job requisition that carries one) branch, carried over automatically — plus a **draft** payroll contract built from the offer's frozen package. Nothing activates itself; who reviews and activates that contract is a separate step, covered on its own page.

Related reading

- [The connected HR lifecycle](#)
- [A job requisition becomes a candidate pipeline](#)
- [How recruitment works](#)

A draft payroll contract, reviewed and explicitly activated

Continuing from [Interview feedback, an approved offer, and a new hire](#): converting an accepted offer into an employee record also creates a **draft** payroll contract. Nothing is payable yet. This page covers the one screen that reviews that contract against the offer it came from, and the one explicit action that turns it into something payroll can use.

Illustrative demonstration prepared using publicly available company context and synthetic data. This page does not represent a production deployment, customer endorsement, or use of real employee data.

https://www.youtube-nocookie.com/embed/RVb_s5T0wyU

[Accepted offer to payroll contract: the protected handoff](#)

The company this page follows

Same illustrative demo structure as the rest of this book: one fictional company, Meridian Interiors. Its name, its roles and every person shown are synthetic; none of it represents a real customer, a real headcount or a real payroll figure.

A frozen copy of the offer, not a new form

The draft contract's review screen does not ask anyone to re-enter the package. It shows a frozen copy of exactly what the candidate accepted — position, effective date, the annual CTC, and the same basic/HRA split the compensation engine computed when the offer was raised — next to the

contract's own current state (Draft, or Active once someone has activated it). Reviewing this screen is how HR confirms the contract matches the offer before doing anything with it.

Activation is a deliberate, explicit, audited action

Nothing in this chain auto-activates. A visible "Activate contract" action is the only way a draft contract starts being payable, and the system records exactly who did it and when, shown back on the same screen afterward. Nothing about *who* may click it is restricted to a different person from the one who converted the candidate — but every illustrative example in this book happens to use someone else anyway, the same maker-checker habit this book uses for requisitions and offers: a different approver, a different authority, a second pair of eyes before anything becomes payable.

What changes the moment it's activated

The contract's own status flips immediately, and so does the surrounding product: the employee's record reflects an active contract right away, and that employee now appears wherever the product lists employees with an active contract. This part of the product answers "is this contract active" with a live, direct check every time it's asked — there's nothing cached or batched about it.

A separate, independent system also *notifies* an employee who has no active contract yet and raises that as something for HR to look at — but that system works by periodically re-checking the company's roster, not by reacting the instant a contract changes. Expect a short delay, not an instant update, from that separate observation — the contract and the employee record themselves are already correct the moment the action completes.

Mitra recommends; it never activates

Mitra can read everything above and tell HR what it means — including flagging an employee with no active contract as something to look at. It does not have, and does not use, any action that activates a payroll contract. The activation shown in this story is a human action, taken on this

screen, by a named person, every time.

Related reading

- [Interview feedback, an approved offer, and a new hire](#)
- [The connected HR lifecycle](#)

Attendance, a geo-fence, and leave for a distributed project team

Continuing from [Branches, departments, and job positions: a multi-site company](#): Meridian Interiors runs project teams out of two offices, Mumbai HQ and a Hyderabad regional office. This page follows one person on that Hyderabad team through three screens that all read from the same record: his attendance, a geo-fence for the site he works from, and a leave request someone else approves.

Illustrative demonstration prepared using publicly available company context and synthetic data. This page does not represent a production deployment, customer endorsement, or use of real employee data.

https://www.youtube-nocookie.com/embed/5i9z51E6k_s

[Attendance, a geo-fence, and leave for a distributed project team](#)

One person, one attendance record

Attendance in BlueOneHR is one record per person per day — clock in, clock out, a shift, a work type, and a validated status — the same record whether the person works from the head office or a regional site. This page does not re-teach the attendance screens themselves: see [Requesting a correction](#) and [Reviewing and validating attendance](#) for the full procedure.

A geo-fence, assigned to the person who works there

The Geo-Fence screen defines a named point with a radius, independent of the company's branch list. A geo-fence is not assigned to a branch directly — it is scoped to a department, a team (job position), an employee group, or an individual employee, and it applies nowhere until one of those assignments exists. For a distributed team, assigning a fence to the specific person working from a regional site is the direct way to represent "this is where they clock in from." See [How geo-fencing works](#) for the screens themselves.

Leave, requested and approved by someone else

Applying for leave and approving it are two different actions, by two different people, every time — the person who raises a request can never also approve it. Casual Leave is a company's own leave type with its own day count; approving a request only succeeds if the balance covers it, and the balance updates the moment it's approved, not before. See [Applying for leave](#) and [Approving and rejecting leave](#) for the full procedure.

Related reading

- [Branches, departments, and job positions: a multi-site company](#)
- [Requesting a correction](#)
- [Reviewing and validating attendance](#)
- [How geo-fencing works](#)
- [Applying for leave](#)
- [Approving and rejecting leave](#)

A first payroll run, reviewed and approved by someone else

Continuing from [A draft payroll contract, reviewed and explicitly activated](#): once people have active contracts and recorded attendance, payroll can be run. This page follows one month's **Payroll Register** run from the first click to an approval given by a different person, and ends at the payout page, which is a mock in this workspace.

Illustrative demonstration prepared using publicly available company context and synthetic data. This page does not represent a production deployment, customer endorsement, or use of real employee data.

<https://www.youtube-nocookie.com/embed/0bbgBBzGrd4>

[A first payroll run: validation, approval by a different person, and a mock payout](#)

The company this page follows

Same illustrative demo structure as the rest of this book: one fictional company, Meridian Interiors. Its name, its people and every figure shown are synthetic; none of it represents a real customer, a real headcount or a real payroll.

Before you begin

A run prices people who have an **Active** contract that covers the month. A contract that is still Draft is not included. Pay for a month also follows each person's **validated attendance** and approved leave for the month's working days: a working day with no validated attendance is treated as unpaid. See [Contracts](#) and [Attendance, a geo-fence, and leave for a distributed project team](#).

Weekly offs are read from the company's own calendar, so a new company needs its weekly offs set up before a run prices a month correctly.

Start a run

1. Go to **Payroll → Payroll Operations → Payroll Register**. A company with no runs shows an empty list and a **Start a new run** button.
2. Choose the month and year, then select **Start run**.
3. The new run opens as a **Draft**. It shows the gross, deductions and net totals for the month, a count of people whose bank details are missing, and one row per person with gross, deduction and net pay.

Nothing is paid by starting a run. The screen says so itself: nothing is paid until the run is approved.

Validate before submitting

Select **Run validation** on the run. The checks list findings by severity. Findings that can stop a run from moving forward, such as a missing PAN, are **Critical**. Open the exceptions list and record a **Resolve** (or **Waive**) decision with a written note. The note is required, the finding is not deleted, and the decision is stored with who made it and when.

A missing bank account is also shown as Critical, but it does not stop the run from being submitted. It stops a payment from being made later, so it stays visible on the run.

Submit, then approve: a different person

Submit for approval moves the run to **Pending Approval**. The run goes to the people in the company's payroll approval group, who see it under **Payroll → Governance → My Approvals**. The person who generated the run and the person who submitted it cannot approve it. If nobody else is eligible, the submission is refused up front rather than left waiting.

When an approver selects **Approve**, the run becomes **Approved** and the payslips in it are confirmed.

Payout is a mock here

The Payout page lists payout batches and says plainly that it uses a mock provider and that no real money moves. A batch can only be prepared for people whose bank details are on record, so until those exist the batch list stays empty. This page makes no claim of a live bank or payment provider connection.

Not shown on this page

- Statutory computation and filing for the month: see the India payroll compliance topic.
- Preparing and approving a payout batch.

Related reading

- [A draft payroll contract, reviewed and explicitly activated](#)
- [Running payroll](#)
- [Payslips](#)

India payroll compliance: a monthly run and a governed correction

Continuing from [A first payroll run, reviewed and approved by someone else](#): once a month's payroll is approved, the **Compliance** area works out the statutory side of that month from the same payslips. This page follows one month through a compliance run, its review and finalization, the statutory working files, and the governed way to correct a finalized run.

Illustrative demonstration prepared using publicly available company context and synthetic data. This page does not represent a production deployment, customer endorsement, or use of real employee data.

https://www.youtube-nocookie.com/embed/4wlJnGq_9Q8

[India payroll compliance: a monthly run and a governed correction](#)

The company this page follows

Same illustrative demo structure as the rest of this book: one fictional company, Meridian Interiors. Its name, its people and every figure shown are synthetic; none of it represents a real customer, a real headcount or a real payroll.

What this page does and does not claim

- Statutory reference values (rates, ceilings, slabs, due dates and file formats) are shipped with a recorded source and a verification status, and they are under controlled review where applicable. Treat every figure as a working figure for a qualified person to check.

Nothing here represents a figure as verified-correct.

- Nothing is filed automatically. There is no portal submission, and nothing on this page is a certification.
- The statutory files are **internal working files**. They are not validated by any government portal.

Before you begin

Go to **Payroll → Compliance**. The Compliance area has six tabs: Compliance Dashboard, Compliance Applicability, Run Compliance, Government Filing, Compliance Calendar and Compliance Settings.

- **Compliance Settings** holds the company's configuration: provident fund, ESI, professional tax (including the state), labour welfare fund, TDS (including the default regime), gratuity and statutory bonus. A run cannot start for a company that has no compliance configuration.
- A run reads the month's payslips. A person with no payslip for the month is not in the run.

See [Running payroll](#) and [Payslips](#).

Run compliance for a month

1. On **Compliance Dashboard**, select **Run Compliance**.
2. Choose the month and year, then select **Run Compliance** again.
3. The run opens as a **Draft**. It shows the statutory totals for the month (provident fund, ESI, professional tax, labour welfare fund, TDS and total employer cost) and one line per employee.

A draft can be recomputed as often as needed; recomputing refreshes the same run rather than creating another one.

Review, then finalize

Review the totals and each employee's line. **Finalize** locks the run. The run's status changes to **Finalized**, and its **Recompute** and **Finalize** buttons are gone: a finalized run cannot be recomputed or edited in place. If the payslips it was computed from change afterwards, the run shows a warning that it no longer matches, and it is not silently updated.

An ordinary run is finalized in one step by a person with permission to change compliance runs.

Statutory working files

From a finalized run, select **Statutory files**. The page offers one generator per file type: the provident fund ECR file, the ESI contribution file, a Form 24Q working file (generated from the last month of a quarter) and a Form 16 Part B working statement (generated from the March run).

- The page states that portal validation (the EPFO and ESIC portals accepting a file) is outside the platform and is not checked. It reports only the platform's own structural validation.
- Form 24Q and Form 16 Part B are internal working files for a tax professional. They are never submitted anywhere.
- A file is generated from a finalized run only, and the platform refuses when there is nothing valid to write, for example when a run has no member rows. A refusal produces no file.
- **Prepare filing tasks** sets up human-assisted tasks for the run. It says plainly that nothing is submitted to a portal.

The run page also offers **Generate XLSX** and **Generate PDF** for an internal summary of the run.

Correcting a finalized run: supersede, with a second person

A finalized run is never changed. A correction is a new run for the same month that **supersedes** the original.

1. A person who has been explicitly given the permission to request a supersede enters a reason of at least 15 characters on the finalized run and selects **Request supersede**. The reason is kept permanently.
2. The request goes to the company's **Compliance Supersession Approvers**. It appears in their **Governance → My Approvals**. The person who requested it cannot approve it, and the request is refused up front if nobody else is eligible.
3. On approval, the original run becomes **Superseded**. It stays readable, shows who requested and approved the supersede and the reason, and its files stay downloadable, but it is no longer counted. A fresh **Draft** run for the same month replaces it.
4. Finalizing the replacement also needs a second person: whoever requested the supersede cannot finalize its replacement.

A period whose filing task has been submitted or paid cannot be superseded at all.

Not shown on this page

- Marking a filing task as submitted, recording a payment, or reconciling it. Those are human bookkeeping steps for a filing a person made elsewhere; the platform does not file.
- Government notices, the Evidence Vault and the Compliance Calendar.

Related reading

- [A first payroll run, reviewed and approved by someone else](#)
- [Running payroll](#)
- [Payslips](#)

From a company objective to an individual one: the OKR cascade

Continuing the Meridian Interiors story from [Branches, departments, and job positions: a multi-site company](#): this page follows one objective down from a company-wide goal to a single site engineer, through the **OKRs** screen (objectives and key results) and, alongside it, **Review cycles**.

Illustrative demonstration prepared using publicly available company context and synthetic data. This page does not represent a production deployment, customer endorsement, or use of real employee data.

<https://www.youtube-nocookie.com/embed/DVP4NHhfda4>

[From a company objective to an individual one: the OKR cascade](#)

Five levels, one chain

An objective sits at one of five levels, and a child objective must sit strictly below its parent: **Company**, **Business unit**, **Function**, **Team** and **Individual**. The **Cascade** tab under **OKRs** draws the whole chain for the people allowed to see it, with each objective's level, its status and its progress. An individual objective needs an owner; a business unit, function or team objective is attached to an org unit.

Org units: the structure behind the cascade

The **Org units** tab lists the business units, functions and teams the cascade hangs off. Each unit can be nested under the one above it, mapped to a department, and given a head. Creating or

moving an org unit needs HR authority. For Meridian Interiors the demo uses one of each: a *Project Delivery* business unit, a *Site Execution* function, and a *Mumbai HQ Fit-out Team*.

Objective, key result, progress

An objective carries one or more key results (the product also supports KPI and KRA types), each with a baseline, a target and a current value. A key result's progress is how far the current value has moved from the baseline toward the target. An objective's own progress is the weighted average of its key results, and a parent objective rolls up the weighted average of its active children; draft and pending children are left out until they are approved.

Recording progress is a separate, append-only step: each update stores a value, an optional note and an evidence reference, and the objective must already be **Active**. In the demo, one engineer's key result (*fit-out milestones handed over on schedule*, target 8) is recorded at 4, which is 50%, and the same figure then shows on the team, function, business unit and company objectives above it.

Approval is maker-checker

A draft objective is submitted for approval through the company's own approval chain. The person who created an objective, and the person who owns it, cannot approve it: someone else on the chain does, and only then does it become active. In the demo, the engineer's objective was created by Sanjay Oberoi, so Sanjay's screen offers no approval action on it, while Devika Rane, a different person on the chain, approves it.

Review cycles

Review cycles is the neighbouring screen where HR sets up a performance review. A new cycle starts in the **Draft** phase, and from there HR can open self review or assign employees. The recording stops at the new draft cycle; it does not follow a cycle through self review, manager review or any rating.

What to know

- Branch-level visibility for OKRs is not enabled yet, so every objective in a company is listed regardless of branch. The OKRs screens say so directly.

- The older **Performance** objectives and key results screens are a separate module; see [How performance works](#).

Related reading

- [Branches, departments, and job positions: a multi-site company](#)
- [The connected HR lifecycle](#)
- [How performance works](#)

Corporate learning: courses, enrolment and completion with Moodle

Continuing the Meridian Interiors story: this page follows a training course through the **LMS** screens, from a course Moodle already holds, to an employee being enrolled, to the completion that comes back. It also says plainly where BlueOneHR's responsibility ends and Moodle's begins.

Illustrative demonstration prepared using publicly available company context and synthetic data. This page does not represent a production deployment, customer endorsement, or use of real employee data. The Moodle learning system in the recording is a test system, not a customer's live learning platform.

<https://www.youtube-nocookie.com/embed/hT44QjfT3-c>

[Corporate learning: courses, enrolment and completion with Moodle](#)

What BlueOneHR does and what Moodle does

The LMS is a **side-car to Moodle**, a separate learning system. The split is deliberate:

- **Moodle builds and runs the learning.** Courses, their content and the learner's progress live in Moodle. BlueOneHR does not author course content and does not issue certificates.
- **BlueOneHR keeps its own record.** It records which courses it knows about, which employees are enrolled, and which have completed, so HR can see training status next to everything else about an employee.
- **The two stay in step through BlueOneHR's side.** Enrolment is sent to Moodle when HR enrolls someone. Completion comes back by a scheduled check, roughly every 30 minutes. Moodle does not push completion to BlueOneHR, so a completion appears after the next check, not the instant the learner finishes.

This page makes no claim about single sign-on or an external identity provider: how a learner signs in to Moodle is not part of the screens described here.

The three LMS screens

The LMS opens at the `/lms/` address of your workspace and has three tabs under one heading: **Courses**, **Enrol Employees** and **Completion Report**. Your plan must include the learning module, and your role must have been granted access: HR Head and HR Ops are granted the LMS permissions explicitly by an administrator rather than automatically.

Courses

The **Courses** tab lists the courses BlueOneHR knows about for your company, with the columns **Name**, **ID number**, **Moodle course id** and **Last synced**. The heading line says courses are pre-built in Moodle and that this list is BlueOneHR's own record of which ones it knows about. A course is registered once, by an administrator at setup; these screens do not create courses. A course whose Moodle course id shows *not synced* cannot yet be used for enrolment.

Enrol Employees

Enrol Employees has two pick lists, **Course** and **Employee**, and an **Enrol** button. Choose both and click **Enrol**: BlueOneHR creates or matches the learner in Moodle, enrolls them in the course, records the enrolment and confirms it with **Enrolment recorded**. Enrolling the same person in the same course again is safe, it does not create a second record. A person who has already completed a course is not reopened by enrolling them again. You only see employees you are allowed to see.

Completion Report

The **Completion Report** lists every enrolment with **Employee**, **Course**, **Status**, **Completed at** and **Grade**. Status is **Assigned**, **Enrolled** or **Completed**. A newly enrolled person shows **Enrolled** until a completion is read back from Moodle; then the row turns **Completed** and shows the completion date. **Grade** shows a dash when Moodle did not return one.

Audit

Each course registration, each enrolment and each completion is written to an append-only audit record on the server, and enrolment and completion are announced to the rest of the platform as

events. The three LMS tabs show courses, enrolments and completions; the audit record itself is the server-side trail behind them.

Related reading

- [The connected HR lifecycle: hiring to workforce operations](#)
- [Workspace apps and inventory sync](#)

Background verification: consent, review and audit (mock provider)

Continuing the Meridian Interiors story: this page follows two background verification cases through the **Background Verification** screens, from a case that cannot be submitted without consent, to a case that cannot be approved by the person who sent it for review. It also says plainly what is real and what is not.

Illustrative demonstration prepared using publicly available company context and synthetic data. This page does not represent a production deployment, customer endorsement, or use of real employee data.

https://www.youtube-nocookie.com/embed/gGax_2-gcG0

[Background verification: consent, review and audit \(mock provider\)](#)

What is real, and what is a mock

In the workspace used for this page the verification provider is a **mock provider**. No external verification agency is contacted and no check is run against a real source: the results you see are simulated and entered by hand. Nothing on this page should be read as a verified identity, address, education or employment record.

What **is** real BlueOneHR behaviour, and the same whichever provider is connected:

- the **case** and the **status** it moves through;
- the **consent gate** that stops anything being submitted before consent is recorded;
- the **review rule** that the person who sent a case for review cannot also approve it;
- the **audit trail**, which records refused attempts as well as successful ones.

Where the screens are

The **Background Verification** list opens at the `/bgv/` address of your workspace, and has two tabs, **Cases** and **Policy**. You only see it if an administrator has granted your role the background-verification manage permission. The list shows each case's **Reference**, **Subject**, **Package**, **Status**, **Consent** and **Due date**, and can be filtered by status and by check type.

A case belongs to either a candidate or an employee and carries a package: **basic** (identity and address), **standard** (identity, address, education and employment) or **comprehensive** (those four plus criminal record, reference and court record). A case moves through **Draft**, **Consent pending**, **In progress**, **Awaiting review**, then **Approved** or **Rejected**, and finally **Closed** (or **Cancelled**).

Consent comes first

A new case opens as a **Draft** with consent not yet recorded. Opening it shows the **Case** details (including **Provider**), a **Consent** section, **Actions**, the **Checks** table, and below them **Documents**, **Tasks**, **Provider events** and the **Audit trail**.

If you choose **Submit to provider** before consent is recorded, BlueOneHR refuses and tells you that consent must be recorded first. The refusal is itself written to the audit trail.

To record consent, fill in the **Purpose**, the **Notice version** and a **Consent evidence reference**, then click **Record consent**. The case keeps the consent details on the page. After that, **Submit to provider** hands each pending check to the provider and the case becomes **In progress**.

Results and review

Each check shows its own status, a risk level and a result summary. With the mock provider, HR records a result by hand against each check. When every check has a final result, the case moves to **Awaiting review** and an open task asks someone to review it.

Send for review opens a review through the approvals engine, using the company's **BGV Review** approval chain. The **Reviewer decision** form then offers **Approve** and **Reject**. A rejection needs a reason and an adverse-action note. Approving despite a discrepancy needs a recorded reason.

The review is maker-checker: the person who sent the case for review cannot also decide it. If they try, BlueOneHR refuses with a message that a different person has to approve it, and records the refused attempt.

Audit trail

The **Audit trail** at the bottom of a case lists **When**, **Who**, **Action**, **Outcome**, **Reason** and **Mode** for each step: case created, consent recorded, each check submitted, each result received, and review decisions. It is append-only, refused attempts are recorded, and identity numbers are never written to it. **Mode** shows **MOCK** for steps handled by the mock provider.

Related reading

- [The connected HR lifecycle: hiring to workforce operations](#)
- [Interview feedback, an approved offer, and a new hire](#)

Mitra recommends the next step; a person decides

Continuing from [A draft payroll contract, reviewed and explicitly activated](#): a draft payroll contract stays a draft until a person activates it. This page shows what **Mitra** does while that is still undone. It reads the company's records, says what it found, and **recommends** a next step. Taking that step stays with a person.

Illustrative demonstration prepared using publicly available company context and synthetic data. This page does not represent a production deployment, customer endorsement, or use of real employee data.

<https://www.youtube-nocookie.com/embed/Ck5VBkR0cIA>

[Mitra recommends the next step; a person decides](#)

The company this page follows

Same illustrative demo structure as the rest of this book: one fictional company, Meridian Interiors. Its name, its people and every figure shown are synthetic; none of it represents a real customer, a real headcount or a real payroll. Some of its people still hold **draft** payroll contracts, which makes them a good example of something Mitra can notice and cannot fix.

Ask Mitra from the Control Center

Mitra opens as a side panel from the top of the Control Center. The panel has two tabs, **Ask** and **Actions & audit**, and a **Voice** control that reads the briefing aloud. Under the question box, the panel's own footer says: "Mitra reads only what your permissions allow, and explains rather than acts."

Asked what needs attention first in payroll, Mitra answers from the company's own records and marks the answer "Computed from your records". The answer carries a **suggested priority** and a link that opens the payroll screen. The suggestion is a sentence for a person to act on. Nothing is changed by asking.

The **Actions & audit** tab is for asking Mitra to look something up, or to do something small on your behalf. Its own text says: "Anything that writes data is only ever a proposal: nothing changes until you confirm it."

Mitra Operations: what Mitra is allowed to do

Administrators see a separate area, **Settings, then Mitra**, with seven screens: Autonomous Operations, Policies, Activity, Action Review, Simulation, Engine and Audit. The first screen shows a status line and the operational controls. In this company the status reads: "Mitra is watching 10 scopes and recommending. It is not changing anything." The **Autonomous execution** control reads **OFF**.

Autonomous execution is a switch an administrator controls. This page describes the company as recorded, with the switch off. It does not say that every company runs this way, and it does not say Mitra acts on its own anywhere. Where an administrator does turn execution on, the scope cards on the same screen show it, and the list of autonomous actions shows what Mitra may do. Those actions concern Help Desk cases, notes, notifications and re-checks. Activating a payroll contract is not on that list.

The Payroll scope card describes its job in one line: "Identify employees payroll cannot pay: no active contract, a contract about to lapse, or a payslip a run missed." Its detection and recommendation switches are on and its execution switch is off.

A finding, and its recommendation

The **Activity** screen lists every situation Mitra has detected, what it decided and the evidence behind it. One entry reads "No active contract: Meera Iyengar". Opening it shows:

- A one-line summary: the employee is active, has no contract in the Active state, and the contracts on file are draft.
- **Impact:** payroll computes wage, pay frequency and filing status from the contract, so this employee will not be paid by the next payroll run.
- **Recommended action:** "Create or activate this employee's contract before the next payroll run."
- A decision label, "Recommend for human review", and the reason "Autonomous execution is switched off globally."
- **What was done:** "No action has been taken on this situation."

- **Evidence:** each line names the database record it came from. The screen says that nothing in the evidence was written by a language model.

Action Review: a person decides

Action Review is where Mitra's proposals wait for a decision. Its subtitle reads: "Mitra proposes; a human decides; the approved action then runs as that human." When nothing is waiting, it says "Nothing to review." A note at the bottom adds that an approved action runs as the person who approved it, and that Mitra has no account of its own to run it with.

The person's side

The employee's own record shows the same facts from the other side. For the employee in this example, the **Compensation and Compliance** block still reads **Contract: Draft** and **Bank account: Not set**. Mitra pointed at the gap. It is closed by a person, on the screens described in [A draft payroll contract, reviewed and explicitly activated](#).

Detections follow a schedule

Mitra's scan runs on a schedule. In this company the **Engine** screen shows it every 900 seconds, which is every 15 minutes. A finding therefore appears some time after a record changes, and a finding can stay listed for a while after the record behind it has been fixed. The contract and the employee record are correct the moment someone saves them. Mitra's list catches up on its next pass.

What this page does and does not claim

- Mitra reads and recommends. In the screens shown, no action was taken by Mitra.
- Autonomous execution is a switch an administrator controls. It is off in the company shown.
- No claim is made that Mitra acts on its own in any scope, and none is made that Mitra can activate a payroll contract.
- Detections are not instant. Plan for the scan interval.

Related reading

- [A draft payroll contract, reviewed and explicitly activated](#)
- [A first payroll run, reviewed and approved by someone else](#)
- [What Mitra is](#)
- [Mitra administrator settings](#)

Payroll to accounting: a journal today, an ERP integration that is ready but not live

Continuing the Meridian Interiors story: once a month's payroll is approved, the numbers have to reach the company's books. This page says plainly what BlueOneHR does for that today, and what is integration-ready but **not live**: no Tally, TallyPrime or Microsoft Dynamics 365 system is connected, reached or synchronised by anything described here.

Illustrative demonstration prepared using publicly available company context and synthetic data. This page does not represent a production deployment, customer endorsement, or use of real employee data.

<https://www.youtube-nocookie.com/embed/jBLUSv7ZnxM>

[Payroll to accounting: a journal today, a Tally and Dynamics integration that](#)

The status in one line

Integration-ready, not live provider execution. The Finance journal and its CSV export are real and usable today. The ERP integration layer behind them exists and is exercised only against mock endpoints that simulate Tally and Dynamics 365. Every record that layer writes is labelled **MOCK**, and nothing in this workspace can label it otherwise.

What is real today: a balanced journal and a CSV export

Go to **Payroll → Finance. Finance & Accounting** lists the journal entries generated for the company, with the **Type, Period, Status, Debit, Credit** and whether the entry is **Balanced**. Entries are generated from an approved **Payroll Register** run (or from a finalized **Compliance** run for the statutory employer cost); they are not typed in by hand.

Select **View** on an entry. A payroll journal for an approved run has:

- **Salary Expense** on the debit side, for the month's gross pay;
- **Salary Payable** on the credit side, for net pay;
- **Payroll Deductions Suspense** on the credit side, for the month's deductions.

The page shows the **Total** of each side. An entry that does not balance cannot be posted. A draft entry can be **Posted**, after which it is read-only. The account names are the company's own **Account Mapping**, which an administrator can change.

Download CSV exports the entry. The page says in its own words that the entry does not post to any live accounting system: you download the CSV and import it into yours, or use it as a reference for a manual journal entry. That is the supported route to accounting today.

The **Integrations** page covers workspace apps and inventory sync. It has no screen for an ERP connection.

What is integration-ready: the layer behind the journal

BlueOneHR contains an ERP integration layer for two systems, built and tested against mock endpoints only. In order:

1. **Payroll event.** An approved payroll run produces a posted, balanced journal. Only posted, balanced journals can go into a batch. If any journal in a batch is unbalanced, the whole batch is refused.
2. **Batch and approval.** The journals of one period become a draft batch. The batch goes through the company's **ERP Sync Batch** approval chain, and the person who built the batch cannot approve it.
3. **Adapter.** An adapter translates the approved batch for the target system: for **Tally**, a Journal voucher in the XML that TallyPrime imports; for **Dynamics 365 Finance**, a journal post.
4. **Mapping.** Each journal line's account name, taken from the company's **Account Mapping**, goes across as the ledger name in the target system, and each journal carries a stable reference that the target system can match on. The ledger must already exist in the target system.

5. **Acknowledgement.** Each reply is recorded as **acknowledged**, **partial**, **failed** or **unknown**. Failures are classified (for example authentication, rate limiting, a missing ledger or an unbalanced voucher).
6. **Reconciliation.** BlueOneHR compares what it sent with what the target system acknowledged. A match marks the batch **reconciled**. A mismatch produces a report and the batch is not marked reconciled.

Two rules are worth knowing because they decide how failures behave:

- If a reply is lost after the target system may have accepted a voucher, the batch becomes **unknown**. It is resolved by asking the target system for its status, and the batch is **never** sent again blindly.
- A retry sends only the items that failed, and records a new attempt rather than overwriting the old one.

Every step writes an audit record, and every record carries the **MOCK** label. The integration layer is switched off by default on an instance, so a production instance cannot send anything to a simulator by accident.

Not confirmed

- **No real Tally, TallyPrime or Dynamics 365 system has been reached.** Nothing has been imported into, posted to or reconciled against a real ledger. The vouchers and journals described above are the format the adapters produce, checked against test fixtures.
- Whether a re-sent voucher replaces the earlier one rather than creating a duplicate in a real TallyPrime company is not confirmed.
- Ledger master validation and real authentication to Dynamics 365 (Microsoft Entra) are not confirmed.
- There is no screen to create an ERP connection, list batches, approve a batch or see its reconciliation, and nothing is submitted on a schedule. Those steps are not available from the product's screens today.
- Splitting payroll expense by branch cost centre for an ERP is opt-in per connection and rests on assumed accounting defaults that no accounting owner has confirmed.

Related reading

- [A first payroll run, reviewed and approved by someone else](#)
- [India payroll compliance: a monthly run and a governed correction](#)
- [The connected HR lifecycle: hiring to workforce operations](#)

Payments: a payout flow that is provider-ready, and a mock today

Continuing from [A first payroll run, reviewed and approved by someone else](#): once a month's payroll is approved, the money has to reach people. This page says plainly what BlueOneHR does about that today and what is **payment provider ready** but **not live**. The Payout module here is a **demonstration flow**: it runs against a **mock** provider. No bank or payment provider is contacted, no real money moves, and no real bank account is used.

Illustrative demonstration prepared using publicly available company context and synthetic data. This page does not represent a production deployment, customer endorsement, or use of real employee data.

<https://www.youtube-nocookie.com/embed/npyb-gRKwoY>

[Payments: a provider-ready payout flow, run against a mock provider](#)

The company this page follows

Same illustrative demo structure as the rest of this book: one fictional company, Meridian Interiors. Its name, its people, every amount and every account number shown are synthetic. The two bank accounts used are made-up numbers; none belongs to a real person or a real bank.

The flow, step by step

1. **An approved payroll run.** A payout batch can only be prepared from a payroll run that has already been approved. It cannot be prepared from a draft or pending run.
2. **Payout instructions.** Each person who has an approved, validated beneficiary gets one payout instruction. Its amount is a snapshot of the person's net pay taken when the batch is prepared, written to two decimal places (paise). Someone with no approved beneficiary

is skipped and listed, and does not stop the rest of the batch.

3. **Two people approve.** The person who prepared a batch never approves it. A member of the company's payout approver group does. The same rule applies one level down: a beneficiary (the person's bank details as registered for payout) must be approved by someone other than the person who added it.
4. **The provider adapter.** Approved instructions are handed to a provider adapter, a fixed interface that every provider sits behind. Each instruction carries its own idempotency key, so submitting the same instruction twice does not create a second payment.
5. **The bank or payment provider.** In this workspace the adapter talks only to a **mock** provider. The provider names that a real integration would use are listed in the product's provider registry, but they are not configured, and they refuse to run rather than fake a success. Mock operation is also enforced in the database: every beneficiary, batch and instruction row is stored as mock.
6. **The response.** The provider answers with a state for each instruction. If the answer is lost or ambiguous, the instruction becomes **Unknown**. An unknown instruction is settled only by asking the provider for its status; submitting it again is refused. A failed instruction is retried as a new instruction that needs a fresh approval, never resent as it was.
7. **Reconciliation.** Reconcile compares the batch's own ledger with the provider's statement, instruction by instruction, on amount and status. A batch becomes **Reconciled** only when every instruction has settled. Anything not settled is reported as a mismatch for a person to look at, never corrected automatically.

What the Payout screens show

The **Payout** page has two tabs, **Batches** and **Beneficiaries**. It states "MOCK provider only -- no real money moves." The **Batches** tab lists batches with reference, period, status, instruction count and total. A batch page shows its instructions (beneficiary, masked account, amount, state, and UTR, which stays empty until a provider settles), the reconciliation results, and an audit trail of every step. Account numbers are masked everywhere. Revealing one is permission-gated and is itself recorded in the audit trail.

In the example, one batch for the November run was prepared, approved by someone other than the person who prepared it, and submitted to the mock provider. Both payments then show **Pending**: the mock provider does not settle them, so reconciling reports two matched and two mismatches, and the batch stays **Submitted**.

What this page does and does not claim

- The flow is **payment provider ready**: the adapter contract, the approvals, the status handling and the reconciliation exist and run against a mock.
- Nothing here is **live**. No bank or payment provider has been connected, and no payment has been executed through a real rail.
- No real bank account number appears on this page or in the video; the accounts used are synthetic.
- Preparing a batch is a service action, the same one Mitra's prepare step uses. The Payout page is where a prepared batch is reviewed, approved, submitted and reconciled.

Related reading

- [A first payroll run, reviewed and approved by someone else](#)
- [India payroll compliance: a monthly run and a governed correction](#)
- [Running payroll](#)