63LABS

Restauranteur Learn & Build

SECTION 13 OF 16

Integrated Restaurant Projects

Multi-project workflows that combine the standalone restaurant builds into usable review systems.



SEPARATE TOPIC PDF / FULL RESTAURANT EDITION PRESERVED

Read this section

Two page-number systems

The section contents below use PDF page positions in this file. Original lesson footers keep their page numbers from the complete 619-page Restaurant Edition. "Master p/pp" still refers to the original 5,410-page Learn & Build source.

Same lessons, smaller files

Content has been separated, not rewritten. Nine shared-boundary pages were divided between their topics so every lesson keeps its ending. The complete Restaurant Edition and original master remain unchanged.

Start with the shared guide

The original scope guide is included below. Project sections also include the shared build starter and spreadsheet setup. These are learning and build specifications, not finished apps or certified operating instructions.

Moving between sections

Local contents and bookmarks open pages in this PDF. Cross-section links use the companion filenames and may require all files in one folder and a supporting PDF reader. If a link does not open, use the numbered section list and the printed edition-page label. Section 01 retains the original full-edition contents and source index.

The 16-section collection

01 Start Here and Source Index

02 Business Planning and Money

03 Stock Suppliers and Food Records

04 Team Operations and Procedures

05 Facilities Equipment and Resources

06 Catering Guests and Delivery

07 Marketing Publishing and Websites

08 Restaurant Products and Services

09 Careers and Team Learning

10 AI Tools and Task Guides

11 Operations Methods and Data

12 Evidence and Reasoning Cases

13 Integrated Restaurant Projects

14 Guest Clubs and Event Ideas

15 Supplier and Product Ideas

16 Concepts and Quality Checks

Contents for this PDF

Numbers on the right are PDF page positions in this section. Retained lesson footers show their complete-edition page labels.

How to use this edition (shared guide) 5

Shared build starter (shared guide) 6

Spreadsheet and document setup (shared guide) 7

Integrated builds: shared method 8

I01 Restaurant-industry service launch 9

I02 Catering client workspace 10

I03 Catering and event sales review 11

I04 Restaurant monthly review 12

I05 Restaurant digital-product release 13

I06 Restaurant content studio 14

I07 Restaurant newsletter desk 15

I08 Restaurant freelance delivery 16

I09 Restaurant learning product 17

I10 Restaurant career portfolio 18

I11 Restaurant stock control 19

I12 Restaurant booking desk 20

I13 Guest feedback improvement loop 21

I14 Restaurant service renewal review 22

I15 Menu or catering pricing review 23

I16 Restaurant shift handoff training 24

I17 Restaurant partner proposal 25

I18 Restaurant audio release 26

Restaurant workweek system 27

Restaurant improvement goal review 29

Menu, ingredient and recipe record system 31

Restaurant learning and service-language practice 33

Restaurant facilities and housekeeping coordination 35

Restaurant relocation and equipment move 37

Restaurant team travel and off-site logistics 39

Restaurant wedding and private-event workspace 41

Restaurant budget, obligations and reserve records 43

Restaurant creative practice and live-event content 45

Operational handoffs and minimal sharing 47

How to use this edition

This is a restaurant-focused curation of the actual 5,410-page 63Labs Learn & Build combined master. It keeps useful original lessons, projects, worked reasoning and tool guidance, then adds clearly labeled restaurant applications. The original file remains unchanged.

A reference library rather than a course

Start with the problem you are solving. Operators can use the money, stock, supplier, team and guest workflows; restaurant-industry builders can use the service, software, content and commercial project material. You do not need to implement every project or buy every named tool.

Broad restaurant use, not just daily operations

The edition retains all 50 original business projects and 43 adaptable planning, facilities, food-record, event and learning projects. Optional supplier, catering, career, venue-event and creative applications are identified as such. Private health, medication, sleep, fitness, mood and veterinary-care projects are not converted into employee-management tools.

What was changed

Repeated setup instructions and repeated practice layouts are consolidated. Restaurant application paragraphs are new adaptations. Source lessons and project requirements are identified by their original master page numbers. Necessary source corrections are labeled rather than silently presented as original text.

What you receive

A learning and build-specification PDF. It does not include generated apps, connected POS systems, live forms, installed automations, completed spreadsheets, legal policies, or a restaurant-specific food-safety program. The projects explain what to build and how to test it.

Source references

Master p or pp always means the physical page position in 63Labs_Combined_Ready.pdf, not a page in this edition. P001-style labels preserve the original project IDs. The contents and PDF bookmarks navigate this edition; the final index traces its source material.

Keep consequential decisions reviewed

Use fictional data while building. Keep private guest, staff and payment data out of shared examples. Use approved restaurant procedures and current authoritative guidance for food safety, employment, taxes, accessibility, privacy and other regulated matters. This edition does not certify compliance or safe equipment work.

Shared build starter

Use this once per build, followed by either the spreadsheet setup or document setup, then the chosen project specification. These consolidated instructions preserve the common safeguards from the source rather than repeating them in every project.

Copy this starter

Act as a patient build partner. Ask the project intake questions in one message, then wait for my answers. I may answer not sure. Separate my facts, goals, assumptions and research still needed. Never invent business facts, sources, permissions, credentials, customers, results or regulatory requirements. Use fictional sample records. Produce the complete requested artifact, not merely an outline. Explain how to copy, set up, use, test and revise it. Do not claim you created or saved a file unless you actually did. If long, stop at a section boundary and resume when I say continue without omissions. End with a requirement-by-requirement checklist. Do not connect services, publish, send messages, spend money or change live records as a side effect of drafting this build.

Add my restaurant context

My restaurant or industry role: [role]. Intended user: [person or role]. Exact task: [task]. Scope and exclusions: [limits]. Approved sources: [documents and dates]. Tools and access available: [tools]. Currency, locale and units: [values]. Owner and reviewer: [roles]. Deadline or work budget: [limit]. Ask about missing information rather than assuming it.

If a response or formula fails

Continue from the exact point where you stopped; keep the same structure, tab names, column order and references. For a formula issue, inspect the actual formula, cell, expected result, source values, locale, dates, units and copied ranges. Explain the proposed correction and rerun the original test. A polished answer is not proof of correctness.

Spreadsheet and document setup

Spreadsheet setup

For Google Sheets, confirm locale and units. The source examples use English functions, comma separators, real dates displayed YYYY-MM-DD, headers in row 1 and data rows 2:1000. Preserve named tabs and column order. Supply actual tab-separated headers, fictional sample rows and exact paste cells. Leave computed sample cells empty. Give each formula with its exact cell, explanation, fill range and expected result. Keep unused rows blank and distinguish missing inputs from zero. Specify validation, dropdowns, formats, filters, frozen headers and formula protection. Keep raw data tables unmerged. Do not add scripts, paid add-ons or external connections by default. Explain how every related range extends beyond row 1000.

Document setup

Create copy-ready pages with clear headings, useful tables only where appropriate, fill-in fields and a completed fictional example. Keep the reusable blank version separate from the example. Supply the actual wording and templates requested, not instructions to write them later. Name owners, actions, dates and review points. Use approved policy content; for changing rules, cite current primary sources only when verified, otherwise list a precise verification task.

First use

Create a safe copy. Enter the source example and compare every expected value. Test blank inputs, explicit zero, duplicate IDs, refunds/reversals, period boundaries and a changed value. For dates and percentages, inspect the underlying values as well as the display. Save evidence before replacing fictional records with authorized live information.

Limits of the supplied specifications

The arithmetic examples have been reviewed, but this PDF is not an executed application test. Generated formulas, data validation, access controls, links, integrations and regulatory treatment still need testing in the actual implementation. Editorial clarifications identify known source gaps, not a claim that all possible edge cases are covered.

Integrated builds: shared method

Master pp 4937-5116; shared ten-page project scaffold consolidated. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Join the projects, not just the files

These source projects connect earlier P-numbered builds into a single workflow. The business capstones are followed by additional restaurant adaptations of the later planning, food-record, facilities, events and learning capstones. Restaurant applications are editorial adaptations; the component requirements, scope, failure cases and acceptance criteria below come from the source. Use fictional records and no live payments, sending or publishing. A description alone is not a finished prototype.

Architecture and first-use record

For each build, name the intended user, decision, first artifact and explicit exclusion. Assign stable IDs to shared records. State the authoritative source, conflict rule and unknown-value rule. Enter source facts once where practical and distinguish derived views. Include a valid item, an incomplete item and a conflicting or repeated item.

Copy-ready shared build instruction

Act as my project drafting assistant. Use the selected brief, restaurant context, component specifications and fictional scope below. Ask about missing facts. Draft components in dependency order. Specify shared IDs, information passed between components and rules for missing/conflicting records. Trace one fictional record from entry to final output. For every changed quantity, show calculation and units; for a status change, show the event or approval supporting it. Deliver usable components, a first-use guide and known limits, not just an outline.

Failure simulation and repair

Seed the listed defect in a fictional copy. Predict expected behavior, record actual behavior if executed, identify smallest reproduction steps and explain the likely cause with evidence. Fix the rule rather than deleting the difficult record. Rerun that same failing case and one ordinary case. Keep expected results separate from tests actually executed.

Second-user change request

Add a second user with different information needs. Document what each person may see, who can edit source records, conflict resolution and regression tests. Fictional separate views illustrate the design; a document or sheet does not prove access controls are enforced. Real security requires separate implementation and verification.

Handoff and review

Deliver a clean blank artifact, a separate worked sample, setup instructions, a recovery path and a known-limits note. Observe a second person attempting the first task. Score completeness, traceability, failure handling, usability and honesty from 0 missing to2 evidenced. Any misleading output or essential unresolved defect blocks release regardless of total. Label practice work honestly in any portfolio and retain version, test record and next review trigger.

I01 Restaurant-industry service launch

Master pp 4937-4946; brief p 4937, build prompt p 4940, reasoning p 4941. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Restaurant application (adapted)

Offer a narrowly scoped menu-update, catering administration or operations-document service to a defined buyer; do not invent demand or treat an accepted quote as cash.

Original outcome and fictional scope

Create a small service offer, a prospect qualification record, and a billing workflow for a fictional local business. One service; one buyer type; ten fictional prospects; two sample invoices.

Component build requirements

1. Business Planning: Write a one-page plan; mark each claim as evidence or assumption; choose one assumption to test this week. Inputs: Customer, problem, offer, delivery steps, startup costs 2. Prospecting: Define qualified versus unqualified; record evidence of fit; review leads with no next action. Inputs: Lead ID, source, fit criteria, contact status, next action date 3. Billing: Define draft, sent, accepted, and paid statuses; link payments to invoices; show remaining balances. Inputs: Quote ID, invoice ID, line items, due date, payment references

Acceptance criterion

A reader can trace inquiry, scope, invoice, and payment without mixing their states.

Defect to seed and detect

An accepted quote is mistaken for collected money.

Reasoning checkpoints

Can a reader identify who pays, what they receive, and why they would choose it? / Every active lead has an owner and a dated next step. / An accepted quote is neither a payment nor a duplicate invoice.

Complete the build

Use the shared integrated-build method. Supply the actual prototype, a traced sample record, expected/actual test evidence, a repair/regression check and a second-user handoff. Keep the stated scope small enough to inspect completely.

I02 Catering client workspace

Master pp 4947-4956; brief p 4947, build prompt p 4950, reasoning p 4951. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Restaurant application (adapted)

Separate a client contact from each catering event or restaurant-service engagement so closing one event does not close the relationship.

Original outcome and fictional scope

Build a client workspace that separates the person, the engagement, and the onboarding tasks. Three fictional clients; two projects for one client; one missing intake item.

Component build requirements

1. Customer Relationships: Create separate contact and interaction lists; connect them by ID; add a view of overdue follow-ups. Inputs: Contact ID, name, organization, source, interaction date, next action 2. Client Management: Separate clients from individual projects; track delivery and renewal tasks for each engagement. Inputs: Client ID, agreed scope, kickoff date, deliverables, check-in date 3. Client Onboarding: Create a request list; assign owners and dates; confirm prerequisites before kickoff. Inputs: Accepted scope, client contact, assets, access needs, kickoff agenda

Acceptance criterion

The active engagement and missing input remain visible after the other project closes.

Defect to seed and detect

Closing one project accidentally closes the entire client relationship.

Reasoning checkpoints

Renaming a customer must not disconnect their history. / Closing one project must not mark every client engagement complete. / Do not request passwords by email; use delegated access where supported.

Complete the build

Use the shared integrated-build method. Supply the actual prototype, a traced sample record, expected/actual test evidence, a repair/regression check and a second-user handoff. Keep the stated scope small enough to inspect completely.

I03 Catering and event sales review

Master pp 4957-4966; brief p 4957, build prompt p 4960, reasoning p 4961. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Restaurant application (adapted)

Use event inquiries or wholesale accounts as prospects. Keep projected pipeline, won agreements, deposits and actual collected receipts separate.

Original outcome and fictional scope

Create a weekly sales review with qualification evidence, deal stages, and next actions. Eight fictional deals; include one lost and one stale deal.

Component build requirements

1. Prospecting: Define qualified versus unqualified; record evidence of fit; review leads with no next action. Inputs: Lead ID, source, fit criteria, contact status, next action date 2. Sales Management: Write stage entry rules; record stage-change dates; keep won revenue separate from estimated pipeline. Inputs: Deal ID, stage, amount, expected close date, next step 3. Sales Strategy: Map discovery, proposal, decision, and delivery handoff; write questions that reveal fit and limits. Inputs: Buyer role, qualification questions, objections, proof, handoff

Acceptance criterion

Forecast assumptions and collected receipts are displayed as different measures.

Defect to seed and detect

The team calls weighted pipeline available cash.

Reasoning checkpoints

Every active lead has an owner and a dated next step. / Lost deals leave the active forecast without disappearing from history. / A confident conversation must not substitute for agreement on scope and price.

Complete the build

Use the shared integrated-build method. Supply the actual prototype, a traced sample record, expected/actual test evidence, a repair/regression check and a second-user handoff. Keep the stated scope small enough to inspect completely.

I04 Restaurant monthly review

Master pp 4967-4976; brief p 4967, build prompt p 4970, reasoning p 4971. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Restaurant application (adapted)

Reconcile restaurant expense and receipt records, refunds and duplicates. This is a planning/review model, not a statutory accounting system.

Original outcome and fictional scope

Build a monthly overview that reconciles spending, receipts, and selected activity measures. Ten fictional transactions; one refund; one duplicate; one unpaid invoice.

Component build requirements

1. Business Overview: Define each metric; calculate it from source rows; add a link from every total to its underlying records. Inputs: Receipts, expense log, lead log, task log, reporting dates 2. Expense Management: Enter one row per payment or refund; link evidence; reconcile against a statement. Inputs: Payment date, vendor, amount, category, receipt, allocation 3. Income Management: Record cash received; separate fees and refunds; reconcile payouts with sales records. Inputs: Receipt date, customer, amount, fees, refund, order reference

Acceptance criterion

A reviewer can reproduce each total from corrected source records.

Defect to seed and detect

The duplicate inflates activity and the invoice inflates cash.

Reasoning checkpoints

A refund and a duplicate lead must change only the appropriate totals. / Use one sign convention and check that refunds reduce spending. / An unpaid invoice must not increase cash received.

Complete the build

Use the shared integrated-build method. Supply the actual prototype, a traced sample record, expected/actual test evidence, a repair/regression check and a second-user handoff. Keep the stated scope small enough to inspect completely.

I05 Restaurant digital-product release

Master pp 4977-4986; brief p 4977, build prompt p 4980, reasoning p 4981. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Restaurant application (adapted)

Prepare an original recipe-document template, training workbook or restaurant administration download. Rights, exact contents and actual file access must be verified before sale.

Original outcome and fictional scope

Prepare a fictional original digital product for a tested website delivery path. Three sample files; one product page; one download test; no live payments.

Component build requirements

1. Digital Products: Show a sample; collect feedback; define exactly what purchase includes; test the delivery experience. Inputs: Buyer, problem, deliverable, format, support scope, production cost 2. Online Store: Prepare an accurate listing; confirm current platform eligibility and rules; rehearse fulfillment and support. Inputs: Product files, listing details, preview images, fulfillment method 3. Website Planning: Map one visitor journey; build essential pages; test on mobile; verify submissions reach the right place. Inputs: Audience, page purpose, content, domain access, forms, success action

Acceptance criterion

Every advertised deliverable is present, opens, and has a first-use instruction.

Defect to seed and detect

The listing promises an editable file missing from the package.

Reasoning checkpoints

A buyer can open the file, understand the first task, and identify support limits. / Every preview must match what the buyer receives. / A success message is insufficient unless the request is actually received.

Complete the build

Use the shared integrated-build method. Supply the actual prototype, a traced sample record, expected/actual test evidence, a repair/regression check and a second-user handoff. Keep the stated scope small enough to inspect completely.

I06 Restaurant content studio

Master pp 4987-4996; brief p 4987, build prompt p 4990, reasoning p 4991. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Restaurant application (adapted)

Build a small menu/news/social campaign from approved facts and owned or licensed assets. No unsupported dietary or allergen claims, invented reviews or fake photography presented as the actual dish.

Original outcome and fictional scope

Create a small branded content workflow with review and publication gates. Six original draft items; two channels; one reviewer.

Component build requirements

1. Content Planning: Plan a small batch; record asset rights and status; schedule review after publication. Inputs: Topic, audience, format, draft owner, approval, publish date
2. Social Media: Assign a purpose to each channel; choose repeatable formats; measure meaningful responses. Inputs: Audience habits, content themes, channel purpose, capacity
3. Branding: Write a position statement; build three message examples; test whether customers understand the offer. Inputs: Customer problem, offer, alternatives, proof, voice preferences

Acceptance criterion

Publication is blocked when required assets or review evidence are missing.

Defect to seed and detect

An item reaches publish status before its rights and factual checks are complete.

Reasoning checkpoints

A scheduled date must not imply that content was actually published. / Each post should serve the intended audience and a clear next action. / Every promised benefit needs either evidence or wording that makes it an intention.

Complete the build

Use the shared integrated-build method. Supply the actual prototype, a traced sample record, expected/actual test evidence, a repair/regression check and a second-user handoff. Keep the stated scope small enough to inspect completely.

I07 Restaurant newsletter desk

Master pp 4997-5006; brief p 4997, build prompt p 5000, reasoning p 5001. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Restaurant application (adapted)

Prepare a sample restaurant news issue and a fictional subscriber-state workflow. Do not export or send a real list; test opt-out exclusion in the design.

Original outcome and fictional scope

Prepare a sample issue and a paper subscriber-state workflow. One original issue; eight fictional contacts; one opt-out; one duplicate.

Component build requirements

1. Newsletters: Draft one complete issue; check every source; ask three target readers what they found useful. Inputs: Reader promise, topic scope, sources, issue structure, cadence 2. Email Marketing: Map signup to welcome message and ongoing delivery; test unsubscribe and suppression handling. Inputs: Subscriber source, preferences, segments, campaign purpose, stop rules 3. Content Planning: Plan a small batch; record asset rights and status; schedule review after publication. Inputs: Topic, audience, format, draft owner, approval, publish date

Acceptance criterion

The approved send list excludes suppressed records and preserves the reason.

Defect to seed and detect

A stale list includes a person who opted out before sending.

Reasoning checkpoints

Confirm dates and links immediately before sending. / A suppressed address must remain excluded from future campaign exports. / A scheduled date must not imply that content was actually published.

Complete the build

Use the shared integrated-build method. Supply the actual prototype, a traced sample record, expected/actual test evidence, a repair/regression check and a second-user handoff. Keep the stated scope small enough to inspect completely.

I08 Restaurant freelance delivery

Master pp 5007-5016; brief p 5007, build prompt p 5010, reasoning p 5011. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Restaurant application (adapted)

Track a restaurant photo, menu design or website-content engagement with explicit revisions, administration and delivery time.

Original outcome and fictional scope

Plan and review a freelance project using total effort and clear scope. One fictional project; three milestones; delivery and admin time tracked separately.

Component build requirements

1. Freelancing: Package one repeatable service; estimate delivery hours; write a scope and handoff checklist. Inputs: Skill, target client, scope, revision limit, capacity, costs 2. Project Management: Break one outcome into tasks; identify blocked work; define what finished means before starting. Inputs: Project ID, task owner, deadline, dependency, completion evidence 3. Time Management: Log actual sessions; prevent overlaps; obtain review before preparing a billing summary. Inputs: Person, project, start, end, break, billable status, rate

Acceptance criterion

Actual effort, remaining estimate, and scope changes can be explained separately.

Defect to seed and detect

The estimate ignores revisions and administration.

Reasoning checkpoints

Extra requests have a defined change process rather than silently expanding the work. / A task cannot be complete solely because its due date passed. / The same hour cannot be billed twice because it appears in two projects.

Complete the build

Use the shared integrated-build method. Supply the actual prototype, a traced sample record, expected/actual test evidence, a repair/regression check and a second-user handoff. Keep the stated scope small enough to inspect completely.

I09 Restaurant learning product

Master pp 5017-5026; brief p 5017, build prompt p 5020, reasoning p 5021. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Restaurant application (adapted)

Use an approved service-handoff or stock-record procedure, not generated safety instructions. Assess a changed case, not copied wording.

Original outcome and fictional scope

Create a short original lesson with practice, a workbook page, and a transfer assessment. One learning outcome; two examples; one changed-context challenge.

Component build requirements

1. Course Creation: Define a demonstrable outcome; build the assessment first; write lessons that prepare the learner for it. Inputs: Learner baseline, final skill, lesson sequence, exercises, rubric 2. Workbook Creation: Turn one task into explanation, example, independent exercise, and review checklist. Inputs: Reader goal, prompts, examples, writing space, completion criteria 3. Learning Practice: Create one concept per card; cite the source; test clarity; log incorrect answers and corrections. Inputs: Source passage, learning objective, question, answer, explanation

Acceptance criterion

The assessment requires reasoning beyond reproducing the example.

Defect to seed and detect

A learner copies the model but cannot explain a changed input.

Reasoning checkpoints

Each lesson includes practice that supports the final assessment. / The sample answer must teach reasoning instead of encouraging copying. / Every answer must be supported by the supplied material.

Complete the build

Use the shared integrated-build method. Supply the actual prototype, a traced sample record, expected/actual test evidence, a repair/regression check and a second-user handoff. Keep the stated scope small enough to inspect completely.

I10 Restaurant career portfolio

Master pp 5027-5036; brief p 5027, build prompt p 5030, reasoning p 5031. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Restaurant application (adapted)

Create a truthful portfolio for a restaurant operations, hospitality, marketing or supplier-support role; label simulated work.

Original outcome and fictional scope

Assemble a truthful practice portfolio and application workflow. One target role type; one labeled practice artifact; two interview answers.

Component build requirements

1. Career Profile: Match genuine evidence to requirements; write concise achievement statements; tailor the opening letter. Inputs: Actual work history, measurable examples, target job requirements 2. Job Search: Build a shortlist; compare gaps; track submissions and interview outcomes; review the strategy weekly. Inputs: Target roles, location, requirements, application date, follow-up 3. Interview Preparation: Prepare situation, action, and result notes; answer aloud; request feedback on clarity and relevance. Inputs: Job description, true work examples, question list, practice partner

Acceptance criterion

The portfolio labels the work accurately and every application claim has evidence.

Defect to seed and detect

A simulated project is described as paid client work.

Reasoning checkpoints

Do not invent metrics, qualifications, employers, or dates. / Separate saved jobs, submitted applications, interviews, and offers. / A strong answer addresses the question without claiming someone else's work.

Complete the build

Use the shared integrated-build method. Supply the actual prototype, a traced sample record, expected/actual test evidence, a repair/regression check and a second-user handoff. Keep the stated scope small enough to inspect completely.

I11 Restaurant stock control

Master pp 5037-5046; brief p 5037, build prompt p 5040, reasoning p 5041. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Restaurant application (adapted)

Use restaurant supplies or ingredients with explicit units and manual stock records. This does not establish safe food conditions or automatic POS synchronization.

Original outcome and fictional scope

Design a purchasing and inventory record for a small fictional shop. Five products; two suppliers; three orders; one partial receipt.

Component build requirements

1. Inventory: Use a movement log; calculate stock from movements; document physical-count differences. Inputs: SKU, opening quantity, receipts, shipments, returns, adjustments 2. Purchasing: Separate requested, approved, ordered, received, and billed quantities; investigate differences. Inputs: Request ID, supplier, approved quantity, unit cost, receipt, invoice 3. Supplier Management: Keep a single supplier record; log performance evidence; schedule renewal decisions before deadlines. Inputs: Supplier ID, service, terms, contact, review criteria, renewal

Acceptance criterion

Ordered, received, reserved, and shipped quantities remain distinguishable.

Defect to seed and detect

An ordered quantity is treated as physically received stock.

Reasoning checkpoints

Reservations reduce availability without changing physical stock. / Do not approve payment solely because an invoice exists. / A favorable price does not erase missing quality or capacity evidence.

Complete the build

Use the shared integrated-build method. Supply the actual prototype, a traced sample record, expected/actual test evidence, a repair/regression check and a second-user handoff. Keep the stated scope small enough to inspect completely.

I12 Restaurant booking desk

Master pp 5047-5056; brief p 5047, build prompt p 5050, reasoning p 5051. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Restaurant application (adapted)

Schedule a private dining space, catering consultation or shared event resource. Include setup/cleanup occupancy and distinguish requested, confirmed, changed and cancelled states.

Original outcome and fictional scope

Create a booking schedule and a support process for changes and conflicts. One shared resource; six bookings; one cleanup buffer; two change requests.

Component build requirements

1. Scheduling: Model availability before bookings; prevent overlaps; record confirmation and cancellation separately. Inputs: Service duration, staff availability, timezone, buffer, cancellation 2. Customer Support: Define urgency rules; assign responsibility; record resolution evidence and reopened cases. Inputs: Ticket ID, customer reference, category, priority, owner, status 3. Meetings: Circulate a short agenda; record decisions separately from discussion; review incomplete actions next time. Inputs: Meeting purpose, decision needed, owner, due date, prior actions

Acceptance criterion

The schedule checks the full resource occupancy and records each resolution.

Defect to seed and detect

Two bookings fit the appointment time but overlap when cleanup is included.

Reasoning checkpoints

Test simultaneous bookings and a timezone change. / Closed does not automatically mean the underlying issue is fixed. / Every action needs one accountable owner and a verifiable output.

Complete the build

Use the shared integrated-build method. Supply the actual prototype, a traced sample record, expected/actual test evidence, a repair/regression check and a second-user handoff. Keep the stated scope small enough to inspect completely.

I13 Guest feedback improvement loop

Master pp 5057-5066; brief p 5057, build prompt p 5060, reasoning p 5061. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Restaurant application (adapted)

Connect guest responses to support follow-up and a manager-approved procedure change. Preserve question wording and avoid generalizing a small response sample.

Original outcome and fictional scope

Build a feedback-to-improvement process with issue classification and a revised procedure. Six fictional responses; two support tickets; one process change.

Component build requirements

1. Customer Feedback: Ask a small consistent question set; group responses; select one improvement and follow up. Inputs: Question, response date, customer group, theme, action owner 2. Customer Support: Define urgency rules; assign responsibility; record resolution evidence and reopened cases. Inputs: Ticket ID, customer reference, category, priority, owner, status 3. Procedures: Observe the task; write numbered actions and decision points; ask someone else to follow the procedure. Inputs: Process trigger, inputs, owner, steps, exceptions, output

Acceptance criterion

The report preserves question meaning and links the change to specific evidence.

Defect to seed and detect

The team averages incompatible feedback questions.

Reasoning checkpoints

Do not present five responses as representative of every customer. / Closed does not automatically mean the underlying issue is fixed. / A new reader can complete a dry run without asking what each step means.

Complete the build

Use the shared integrated-build method. Supply the actual prototype, a traced sample record, expected/actual test evidence, a repair/regression check and a second-user handoff. Keep the stated scope small enough to inspect completely.

I14 Restaurant service renewal review

Master pp 5067-5076; brief p 5067, build prompt p 5070, reasoning p 5071. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Restaurant application (adapted)

Review software and service renewals, payment timing and operational dependencies. A cancellation request is not confirmation, and a file backup is not a demonstrated restore.

Original outcome and fictional scope

Create a subscription review showing cash timing and continuity considerations. Five fictional services; monthly and annual charges; one cancellation request.

Component build requirements

1. Subscriptions: Separate billing and cancellation deadlines; record actual usage; assign a keep-or-cancel decision. Inputs: Service owner, billing amount, renewal date, notice deadline, use 2. Cash Flow: Forecast by week; separate expected from confirmed items; update balances when timing changes. Inputs: Opening cash, dated receipts, dated payments, timing assumptions 3. Business Continuity: Map essential dependencies; choose a manual fallback; rehearse a small outage on paper. Inputs: Critical service, disruption scenario, recovery target, contacts, backups

Acceptance criterion

Renewal decisions, confirmation status, and actual payment timing remain visible.

Defect to seed and detect

A requested cancellation is treated as confirmed before the renewal date.

Reasoning checkpoints

A reminder date must fall before the decision deadline. / A sale with payment next month must not fund this week's bills. / Having a backup does not prove it can be restored.

Complete the build

Use the shared integrated-build method. Supply the actual prototype, a traced sample record, expected/actual test evidence, a repair/regression check and a second-user handoff. Keep the stated scope small enough to inspect completely.

I15 Menu or catering pricing review

Master pp 5077-5086; brief p 5077, build prompt p 5080, reasoning p 5081. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Restaurant application (adapted)

Compare two fictional offers using explicit direct, fee, labor and fixed-cost assumptions. Keep markup and contribution margin distinct. Apply the P042/P043 clarifications and avoid calling contribution net profit.

Original outcome and fictional scope

Build a pricing worksheet linked to included costs, time, and a whole-unit break-even check. Two fictional offers; explicit excluded costs; one changed cost input.

Component build requirements

1. Pricing: Define the cost basis; calculate contribution and margin; test sensitivity to one cost change. Inputs: Selling price, direct cost, fees, labor assumptions, target contribution 2. Financial Viability: Calculate contribution per unit; divide fixed costs by contribution; round required units upward. Inputs: Fixed costs, price per unit, variable cost per unit, period 3. Time Management: Log actual sessions; prevent overlaps; obtain review before preparing a billing summary. Inputs: Person, project, start, end, break, billable status, rate

Acceptance criterion

The formulas use the correct bases and whole-unit target with assumptions stated.

Defect to seed and detect

Markup is labeled margin and the break-even target is rounded down.

Reasoning checkpoints

Do not call contribution net profit when fixed costs are excluded. / For positive fixed costs, zero or negative contribution gives no finite nonnegative sales-volume break-even. If fixed costs are zero, zero units already cover fixed costs; handle zero contribution explicitly and never divide by zero. / The same hour cannot be billed twice because it appears in two projects.

Complete the build

Use the shared integrated-build method. Supply the actual prototype, a traced sample record, expected/actual test evidence, a repair/regression check and a second-user handoff. Keep the stated scope small enough to inspect completely.

I16 Restaurant shift handoff training

Master pp 5087-5096; brief p 5087, build prompt p 5090, reasoning p 5091. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Restaurant application (adapted)

Use one approved non-hazardous task and two learner attempts. Reading or attendance is not independent demonstrated competence.

Original outcome and fictional scope

Create a procedure, a short training exercise, and a handoff meeting record. One repeatable task; two learner attempts; three open actions.

Component build requirements

1. Procedures: Observe the task; write numbered actions and decision points; ask someone else to follow the procedure. Inputs: Process trigger, inputs, owner, steps, exceptions, output 2. Team Training: Sequence observation, guided practice, independent demonstration, and review; define evidence for each skill. Inputs: Role duties, prerequisites, approved procedures, mentor, assessment 3. Meetings: Circulate a short agenda; record decisions separately from discussion; review incomplete actions next time. Inputs: Meeting purpose, decision needed, owner, due date, prior actions

Acceptance criterion

The record separates attendance, assisted practice, and checked performance.

Defect to seed and detect

Attendance is reported as independent task competence.

Reasoning checkpoints

A new reader can complete a dry run without asking what each step means. / Reading a manual alone does not prove practical competence. / Every action needs one accountable owner and a verifiable output.

Complete the build

Use the shared integrated-build method. Supply the actual prototype, a traced sample record, expected/actual test evidence, a repair/regression check and a second-user handoff. Keep the stated scope small enough to inspect completely.

I17 Restaurant partner proposal

Master pp 5097-5106; brief p 5097, build prompt p 5100, reasoning p 5101. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Restaurant application (adapted)

Draft a local event, creator, supplier or membership collaboration with measurable deliverables, no guaranteed customer purchases and no unverified audience claims.

Original outcome and fictional scope

Draft a fictional partnership exchange with controllable deliverables and review criteria. Two parties; one audience; three promised placements; no sales guarantees.

Component build requirements

1. Partnerships: Define what each party supplies; substantiate audience claims; agree how delivery will be evidenced. Inputs: Partner audience, proposed deliverables, schedule, measurement, price 2. Memberships: Plan one delivery cycle; calculate workload per member; define what repeats each month. Inputs: Member promise, recurring activities, capacity, onboarding, cancellation 3. Branding: Write a position statement; build three message examples; test whether customers understand the offer. Inputs: Customer problem, offer, alternatives, proof, voice preferences

Acceptance criterion

The proposal defines delivery evidence and avoids unsupported outcome promises.

Defect to seed and detect

Exposure is described as guaranteed customer purchases.

Reasoning checkpoints

Promises in the proposal must match operational capacity and available measurement. / The service remains deliverable if every member uses the included benefits. / Every promised benefit needs either evidence or wording that makes it an intention.

Complete the build

Use the shared integrated-build method. Supply the actual prototype, a traced sample record, expected/actual test evidence, a repair/regression check and a second-user handoff. Keep the stated scope small enough to inspect completely.

I18 Restaurant audio release

Master pp 5107-5116; brief p 5107, build prompt p 5110, reasoning p 5111. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Restaurant application (adapted)

Create a chef interview, restaurant story or training audio sample with consent, rights and consistent transcript/artwork versions. Do not imply publishing occurred.

Original outcome and fictional scope

Plan a sample audio release with assets, permissions, and version control. One original sample recording; transcript; artwork; review log.

Component build requirements

1. Audio Content: Build an episode brief; record a short test; edit for clarity; check attribution and publishing details. Inputs: Episode purpose, guest permission, outline, recording plan, release tasks 2. Content Planning: Plan a small batch; record asset rights and status; schedule review after publication. Inputs: Topic, audience, format, draft owner, approval, publish date 3. Asset Management: Assign IDs; record transfers and service; retain disposal evidence without deleting the history. Inputs: Asset ID, serial number, location, custodian, purchase, service

Acceptance criterion

The final files agree and all blocking reviews are recorded.

Defect to seed and detect

The audio is revised but the transcript retains the removed passage.

Reasoning checkpoints

Listen to the exported file, not only the editing timeline. / A scheduled date must not imply that content was actually published. / A location change must not create a second asset.

Complete the build

Use the shared integrated-build method. Supply the actual prototype, a traced sample record, expected/actual test evidence, a repair/regression check and a second-user handoff. Keep the stated scope small enough to inspect completely.

Restaurant workweek system

Adapted from combined-master pp 5117-5126; all ten pages reviewed. Restaurant examples and tests are editorial adaptations, not completed source demonstrations. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Build a connected week, not three lists

Build one manager workweek prototype connecting P051 Weekly Planning, P052 Daily Planning and P053 Habits. Keep the source scope: one fictional week, three priorities and two recurring routines. The user must decide what fits around fixed restaurant commitments, what receives time today and what to defer. Choose a first artifact such as a linked weekly plan, daily view and routine log. Exclude staff rota approval, payroll and automatic assignment. A new week must reflect actual capacity and an explicit priority choice, not merely copy last week's unfinished work.

Inputs and shared records

P051 inputs: fixed service or meeting commitments, available work blocks, priorities and task estimates. P052 inputs: approved working windows, recurring duties and suitable event anchors; do not collect employees' private sleep schedules. P053 inputs: routine definition, planned opportunities, a useful minimum action and logged status. Give every task a stable task ID and every routine occurrence its own dated ID. Enter estimates and approved availability once; derive daily views from that source. A changed task name must retain its history. Flag conflicting availability for review. Unknown duration is not zero, and an unlogged routine is not automatically missed. Include one valid, one incomplete and one repeated or conflicting record.

Connect the three components

1. Weekly planning: place immovable commitments first, include necessary preparation and travel, then schedule realistic task blocks and interruption space. Show work deliberately excluded from the week.
2. Daily planning: select tasks from those blocks; attach each small routine to an existing restaurant event, such as the end of the administrative handoff. Define a minimum version only for optional administrative routines. A reduced routine must never replace, omit or defer mandatory approved operating steps; record any blocked mandatory duty and escalate through the restaurant's approved process.
3. Routine review: define observable completion, preserve completed and missed occurrences, and distinguish not scheduled from unlogged. Compare completed opportunities with planned opportunities, not all calendar days. Feed the review into next week's capacity decision rather than manufacturing catch-up work.

Copy-ready build brief

Act as my project drafting assistant. Build the restaurant workweek system described here for [role] in [format/tools], using [confirmed operating inputs]. Use one fictional week, three priorities and two administrative routines. Connect weekly capacity, daily actions and dated routine occurrences through stable IDs. Return the working prototype, one traced example, a clean start copy, a first-use guide and a limits note. Show what was deferred and why. Test the carry-forward defect and the reduced-capacity change below. Label predictions separately from tests actually run. If a handoff fails, correct that first failure and explain how to rerun it without expanding the scope.

Worked restaurant demonstration: expected reasoning

Fictional manager M1 has eight hours available after fixed commitments. Three priorities require three, four and two hours: nine hours in total. Choosing the first two uses seven hours and leaves one hour for interruptions; the two-hour third priority stays visible as explicitly deferred. The two brief routines are already included in the seven-hour estimate, so they are not added again. A routine planned for Monday, Wednesday and Friday has Monday completed, Wednesday missed and Friday unlogged. It is not three failures, and Tuesday and Thursday are not missed opportunities. The next weekly review preserves those statuses and re-estimates remaining work. This is an expected worked result, not a claim that a particular spreadsheet was tested.

Failure simulation and changed information

Seed the source defect: copy the deferred two-hour task into the next full eight-hour week without removing or reducing anything. The capacity check must expose the overload, not conceal it by changing dates. Record reproduction steps, actual result, cause, correction and a rerun of the same case. Then reduce the next week's available time from eight to six hours because of a confirmed service commitment. Require a new explicit priority decision: for example, three hours on the first priority, two on a separable portion of the second and one buffer, with remaining work retained in the backlog. A routine missed in the previous week must not erase prior completions or become compulsory double work.

Second-user handoff and acceptance

Give an assistant manager a fictional daily-action view while the manager retains the authority to change weekly capacity and priority. Document which fields each may edit and how contradictory updates are resolved; sample views do not enforce security. Ask the second user to find today's next task and explain the deferred item without a verbal walkthrough. Accept only if fixed commitments and all counted hours reconcile, routines have distinct scheduled/missed/unlogged states, the overload remains visible until an explicit decision resolves it, and every displayed task traces to its source ID. Retain the corrected artifact version, test evidence, observed hesitation, revised first-use instruction and a next-review trigger. Demonstrating a planning prototype does not establish that it is a production scheduling system.

Restaurant improvement goal review

Adapted from combined-master pp 5127-5136; all ten pages reviewed. Restaurant examples and tests are editorial adaptations, not completed source demonstrations. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Build a decision, not a decorative score

Connect P054 Goals, P055 Life Organization and P094 Decision Making into a restaurant improvement review. Replace personal life areas with named restaurant work areas and a specific operational decision. Retain the source scope: two goals, one increasing measure and one decreasing measure. The user must see what improved, what is still unknown and which next action is justified. Choose one bounded decision, such as which administrative workflow to improve next. Do not make a combined score that hides a non-negotiable constraint.

Inputs and shared records

For each goal record its baseline, target, deadline, observable milestone and measurement unit. For the review dashboard record the restaurant work area, selected measure, source log, reporting period and review day. For the decision record alternatives, stated values, tradeoffs, constraints, reason for choosing and evidence that could change the choice. Use goal IDs, observation IDs and a linked decision ID. Keep source measurements authoritative; a chart is a derived display. Never treat missing observations as zero. If two records claim different values for the same observation, show the conflict and its source dates. Include a valid observation, an incomplete one and a repeated one without counting the repeated measurement twice.

Connect the three components

1. Goals: translate each desired result into an observable milestone and a controllable next action.
2. Work-area review: display only a few defined indicators, each with its unit, period, source and freshness. Explain whether the goal increases, decreases or maintains a measure.
3. Decision comparison: compare the same alternatives against the same stated values and constraints. Record the selected action and review trigger. A favorable metric cannot overrule an unmet essential constraint. Keep the dashboard and decision record connected so a changed measurement prompts reconsideration rather than silently rewriting the earlier rationale.

Copy-ready build brief

Act as my project drafting assistant. Build a restaurant improvement dashboard for [role] in [format/tools] from [confirmed goals, source measures and constraints]. Use two fictional goals with opposite desired directions. Connect goal definitions, source-linked review indicators and one documented operational decision. Make units, reporting periods, incomplete data and assumptions visible. Deliver the prototype, one entry-to-decision trace, a first-use guide and known limits. Test a direction error, changed baseline and missing observation. Separate expected calculations from executed test results and repair the first failing handoff before adding features.

Worked restaurant demonstration: expected reasoning

Fictional goal G1 increases the count of completed menu-description checks from a baseline of 20 to a target of 30 per review period. At 25, progress through the intended change is (25 − 20) / (30 − 20) = 50%. Goal G2 decreases median response time for routine event enquiries from eight to four hours; at six hours, (6 − 8) / (4 − 8) = 50%. This shared signed baseline-to-target formula handles both directions when definitions and denominators are valid. The source warning is against ignoring direction, not against all shared formulas. Label the period and source for both measures; the percentages alone do not demonstrate equal business value or justify an average score.

Failure simulation and changed information

Seed an increasing-only formula that reports the decrease from eight to six hours as deterioration, or a current-divided-by-target score that makes worse response time look better. Compare its output with the defined direction and repair the rule, keeping the same input. Then correct the response-time baseline from eight to ten hours based on a revised fictional source log. At six hours toward a four-hour target, progress becomes (6 − 10) / (4 − 10), about 66.7%; retain the baseline revision and earlier review version. A baseline equal to its target needs a separate attainment/maintenance rule, not division by zero. An absent current measure must remain unknown rather than receiving a percentage.

Review, second user and acceptance

Have one fictional user enter observations and another approve goal definitions and decisions. State what each can change and how a disputed source correction is reviewed; sample views alone do not provide access control. Ask the reviewer to explain why the same 50% does not make the two goals interchangeable and why an unresolved venue-capacity constraint can still block an option. Accept only if success direction is correct, every measure has units and a period, baseline changes remain traceable, unknowns stay visible and the decision states its limits. Deliver source-linked goal records, the corrected dashboard, actual test evidence and one observed first-use attempt. Record ready, revise or blocked with reasons, not a score alone.

Menu, ingredient and recipe record system

Adapted from combined-master pp 5137-5146; all ten pages reviewed. Restaurant examples and tests are editorial adaptations, not completed source demonstrations. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Build a bounded restaurant planning demonstration

Connect P056 Meal Planning, P057 Food Organization and P058 Recipes as a restaurant menu-demand, ingredient and recipe-record prototype. Keep five fictional menu uses, shared ingredients and one recorded stock item. Replace household preferences with the supplied restaurant menu brief, approved recipe choices and serving requirements. The user must trace a planned portion count to ingredient demand, verified available stock and remaining requirements. This is a records exercise, not commercial recipe validation, nutrition advice, allergen assurance or a food-safety procedure. The worked reconciliation below covers not-yet-prepared planned menu uses against a matching checked stock snapshot. The original P056 gross weekly plan can include meals already marked Prepared; its gross historical demand must not be relabeled as the current remaining-to-buy quantity after ingredients have been consumed.

Inputs and shared records

Menu planning needs approved recipe references, portion counts, service dates, preparation-time assumptions and required ingredients. Ingredient records need item identity, unit, recorded quantity, verification date, available quantity, planned allocation and storage-location reference. Recipe records need title, source attribution, version, base yield, ingredient quantities, supplied method and change notes. Use distinct recipe-version, menu-use, ingredient and stock-record IDs. An ingredient name alone is not a reliable identity. Keep a recipe's base quantities separate from scaled demand. A missing unit conversion, unverified availability or conflicting count is an exception requiring review, not permission to guess.

Connect the three components

1. Recipe records: retain the supplied original, source and yield. Store an adaptation or tested change separately with its own version and evidence; do not overwrite the base recipe when serving counts change.
2. Menu planning: associate each of the five menu uses with a particular recipe version, portion requirement and schedule. Group preparation tasks only as planning records based on supplied procedures.
3. Ingredient reconciliation: define the planning cutoff, include only not-yet-prepared menu uses in forward demand, aggregate that demand by stable ingredient identity in compatible units, deduct verified available stock from a matching checked snapshot once and show remaining quantity. Keep a separate gross weekly/history view if it includes Prepared uses; do not subtract post-consumption stock from that historical total to create a current purchasing requirement. Keep consumption, allocation and available balance separate so an item cannot be both used and still fully available. Package counts require an explicit conversion before being combined with weight.

Copy-ready build brief

Act as my project drafting assistant. Build a restaurant recipe-version, menu-use and ingredient-requirement prototype for [role] using [supplied recipe records, serving counts and stock facts] in [format/tools]. Use five fictional menu uses, shared ingredients and one verified fictional stock item. Label sample quantities and all unresolved conversions. Trace one ingredient from base recipe through scaled demand to remaining requirement. Deliver linked records, a worked sample, a first-use guide and limits. Test duplicate ingredient demand and a revised recipe version. Do not invent preparation, storage, cleaning, allergen or commercial food-safety instructions. Keep expected arithmetic separate from tests actually executed.

Worked restaurant demonstration: expected reasoning

This fictional demonstration uses only not-yet-prepared menu uses and a stock snapshot checked at the same planning cutoff. Fictional recipe R1 version 1 yields ten portions and records 500 g of ingredient I1. Menu use U1 requires 20 portions, so its planning demand is 1,000 g. Another menu use requires 600 g of the same ingredient ID; the remaining three uses require none of I1. Gross I1 demand is therefore 1,600 g. A single verified stock record shows 400 g available and not already allocated elsewhere. Remaining requirement is 1,200 g. If supplied package data states 250 g per package, five whole packages would total 1,250 g, leaving a 50 g planning excess. Without that verified conversion, keep the requirement in grams and mark package quantity unresolved. No purchase or production decision is executed by this example.

Failure simulation and changed information

Seed the source defect by treating "I1" and an alternate label for I1 as separate ingredient identities, or by re-importing the same menu-use record. The total must not duplicate legitimate demand or subtract the same stock twice. Resolve the identity/import rule with a traceable correction rather than deleting a genuine use. Then introduce R1 version 2 with 600 g per ten portions while U1 is still approved against version 1. U1 must remain at 1,000 g until an authorized record change selects version 2. After that explicit change, U1 demand is 1,200 g, total demand 1,800 g and remaining requirement 1,400 g. Preserve both recipe versions and the changed planning reference. Finally, mark U1 Prepared and record its ingredient consumption in the stock history. Rebuild forward demand from the remaining unprepared uses and obtain a matching current stock snapshot; do not carry U1's consumed demand into the current remaining-to-buy calculation. Preserve U1 in the gross weekly/history view.

Handoff and acceptance

Give a fictional kitchen planner control over approved recipe-version references and a stock recorder responsibility for dated quantity observations. Show disagreements for review, not automatic resolution by the most recent sheet. Ask a second user to trace the 1,200 g original remaining requirement without recomputing unrelated items. Accept only when every menu use links to the intended recipe version, stock is counted once, compatible units reconcile, missing conversions remain blocked and no consumed or allocated stock is counted twice. A sample view is not an access-control guarantee. Retain the prototype version, actual defect/regression results, first-use corrections and the limits of the supplied operational data.

Restaurant learning and service-language practice

Adapted from combined-master pp 5147-5156; all ten pages reviewed. Restaurant examples and tests are editorial adaptations, not completed source demonstrations. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Build evidence of independent use

Connect P064 Study Planning, P065 Learning Practice and P066 Language Learning around one non-hazardous restaurant communication topic. Keep six terms and two application tasks. Suitable supplied material might cover describing a venue layout or explaining the restaurant's published booking options. Exclude food-safety certification, medical/allergen decisions and other topics where a practice exercise could be mistaken for authorization. The user must decide what to practice next from independent performance, not time spent reading.

Inputs and shared records

Study planning needs the topic, deadline, baseline gaps, available sessions and practice tasks. Learning cards need a supplied source passage, its version, one learning objective, a question, a source-supported answer and an explanation. Language practice needs a target service conversation, the learner's stated starting ability, available time, resources and access to feedback. Use objective IDs shared by cards, scheduled practice and attempts. Keep attempt ID, date, prompt version, response, evidence of independent use, feedback and correction. A source revision must identify affected cards; a missing or ambiguous source answer must stay unresolved. Use fictional learner IDs rather than collecting private staff profiles.

Connect the three components

1. Study plan: reserve retrieval, application and review sessions, then revise the next session from the previous self-check.
2. Learning cards: put one concept on each card, retain its source reference and explanation, and log incorrect answers and subsequent corrections. Recognition of a displayed answer is not recall.
3. Service-language practice: choose one real communication task from the supplied brief; practice understanding a request and producing a response. Link relevant phrases to the same six objectives. Keep feedback about meaning, clarity and context separate from mere practice duration. Do not invent authoritative translations or venue policies when the supplied source is silent.

Copy-ready build brief

Act as my project drafting assistant. Create a restaurant learning prototype for [role] in [format/tools] from [approved non-hazardous source material and chosen communication situation]. Use six terms, two application tasks and planned review sessions. Build a linked schedule, source-supported cards and conversation practice. Trace an incorrect independent answer through feedback, correction and a later attempt. Deliver the prototype, first-use instructions and a limits note. Test the false mastery caused by reading time and a later source revision. Do not claim certification or demonstrated competence where only predicted results exist.

Worked restaurant demonstration: expected reasoning

A fictional host studies six venue-layout terms for 20 minutes. With the answer cards hidden, the host explains four correctly and cannot explain two. Task A asks the host to guide a guest to a named area using the supplied venue map; Task B asks the host to clarify a changed meeting point in the chosen service language. Reading time is logged as participation, while each independent answer and task attempt is recorded separately. If Task A is clear but Task B confuses the two locations, the next session targets that distinction and a new response attempt. Seeing the correct phrase once is not evidence that the learner can use it when the guest asks differently.

Failure simulation and changed information

Seed the source defect by marking all six objectives mastered once 20 minutes of reading is logged. The artifact should expose the mismatch between participation and independent evidence. Repair the mastery/status rule without erasing the original attempt. Retest with a learner who reads for longer but still cannot perform Task B. Then revise the fictional venue map so the meeting point changes. Mark the affected cards and dialogue prompts for review, preserve prior attempts under the old source version, and test the revised task with the model hidden. A prior correct answer to an obsolete prompt must not certify the new procedure.

Second-user review and acceptance

Give the learner a practice view and a trainer a source/feedback review view. Define which information is necessary for each and avoid exposing individual learning records unnecessarily; separate views are a design demonstration, not security implementation. Ask a second person to complete one practice session without understanding the build process. Accept when every answer is supported by the supplied material, errors and corrections are traceable, the six objectives connect across the three components and both application tasks require independent performance. Retain actual attempts separately from illustrative answers, record the observed first-use difficulty and give one evidence-based next practice task. The artifact demonstrates a learning method, not professional qualification.

Restaurant facilities and housekeeping coordination

Adapted from combined-master pp 5157-5166; all ten pages reviewed. Restaurant examples and tests are editorial adaptations, not completed source demonstrations. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Organize approved work without inventing procedures

Connect P067 Home Cleaning, P068 Home Maintenance and P070 Home Organization as restaurant housekeeping, facilities reminders and small-area organization. Retain three fictional work areas, five tasks and two owners. Replace household routines with supplied restaurant procedures, manufacturer guidance and authorized responsibilities. The user must identify which occurrence is still open, who owns it and what evidence closes it. Exclude technical repair, chemical instructions and invented commercial cleaning or maintenance intervals. A shorter organization session or reduced optional housekeeping scope must never omit, postpone or substitute for mandatory approved cleaning, safety or maintenance steps. Keep an uncompleted required occurrence visible and follow the approved escalation process.

Inputs and shared records

Housekeeping inputs: work areas, approved task descriptions, sourced frequency, available work time and responsibility agreements. Maintenance inputs: asset/system identity, model, manual or qualified-provider reference, last recorded service, stated interval and provider. Organization inputs: one bounded storage area, a session limit, sorting categories and authorized disposal or relocation options. Give each asset, area, task definition and dated task occurrence a distinct ID. Store the recurrence rule with its source separately from each completion event. Unknown last service or unsupported interval stays unresolved. Do not merge two occurrences merely because the task wording is identical.

Connect the three components

1. Housekeeping record: distinguish routine resets from less frequent work; record assigned owner, due occurrence, completion date and evidence required by the supplied procedure.
2. Facilities reminder: calculate a due reminder only from the documented rule for that exact asset, keeping completed service history and overdue occurrences visible. A reminder is not a repair instruction.
3. Organization session: choose a small stockroom drawer or administrative shelf; classify items as keep, relocate, authorized disposal/donation or review, and assign a location to retained items. Do not discard restaurant property, personal belongings or required records without the appropriate owner's agreement.

Copy-ready build brief

Act as my project drafting assistant. Build a restaurant housekeeping, facilities-reminder and small-area organization prototype for [role] using [approved task definitions, asset guidance and ownership decisions]. Use three fictional areas, five tasks and two owners. Connect stable asset/area IDs, recurrence rules and dated task occurrences. Deliver linked records, one traced occurrence, a clean first-use copy and known limits. Test an overdue task hidden by automatic date advancement and a changed interval source. Do not supply technical repair, chemical or food-safety instructions, or infer missing maintenance guidance.

Worked restaurant demonstration: expected reasoning

Fictional task T3 is due on 10 November under a supplied demonstration rule and belongs to owner A. On 11 November, no completion evidence exists. T3's 10 November occurrence remains overdue even if a later occurrence has been generated. A message saying "probably done" is recorded as reported, not verified completion. Once the stipulated evidence is entered on 12 November, that occurrence can be closed with its actual completion date. Whether the next due date follows a fixed calendar or the last completion depends on the supplied recurrence rule; these are different designs and must not be silently substituted. The dates illustrate record behavior, not a recommended maintenance interval.

Failure simulation and changed information

Seed the source defect by overwriting the 10 November due date with the next date at the weekly rollover. The overdue view will falsely show no work outstanding. Repair the data model by preserving occurrences and their status, then rerun the rollover and a normal on-time completion. Next, replace an old fictional guidance reference with a newly approved interval rule effective on a specified date. Flag affected future reminders for review, retain the previous rule/version and do not erase overdue work under the old rule. An asset with no verified interval must show "guidance needed," not a guessed due date.

Handoff and acceptance

Let the two fictional owners record their own completion reports while the facilities coordinator maintains asset guidance and recurrence definitions. Document how a disagreement is resolved and which evidence the receiving person needs. Ask the second user to find the overdue occurrence and the reason it has not closed. Accept only if all five tasks have an owner or a visible assignment gap, open occurrences survive rollover, reminder calculations cite their rule, organization decisions preserve ownership authority and the change test retains history. Deliver the exact tested file, first-use/recovery instructions and an evidence-based ready/revise/blocked decision. A document with separate views does not by itself enforce access permissions.

Restaurant relocation and equipment move

Adapted from combined-master pp 5167-5176; all ten pages reviewed. Restaurant examples and tests are editorial adaptations, not completed source demonstrations. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Build retrieval across a move

Connect P069 Household Assets, P071 Moving and P082 Personal Records as a restaurant asset inventory, relocation plan and operational document index. Retain two fictional areas, twelve items and three document categories. The user must find an item and its relevant record after a box label or destination changes. Use administrative equipment or non-hazardous props for the demonstration. Exclude equipment installation, hazardous lifting, utilities work and food-transport procedures.

Inputs and shared records

Asset inputs: item, current location, serial/reference number when needed, photo, receipt and user-supplied value, with purchase cost distinct from estimated current value. Move inputs: dates, fictional origin/destination, role contacts, box IDs, services, responsibilities and confirmation status. Document inputs: category, secure source location, owner, expiry or renewal date and access method, never passwords or secret credentials. Assign separate stable asset IDs, box IDs, move IDs and document IDs. Photos are evidence attachments to an item, not automatic new assets. Keep the inventory as the item source and the move plan as the transport/destination source; visibly flag conflicting location records.

Connect the three components

1. Asset inventory: record the twelve items by the two starting areas, preserve evidence references and do not turn duplicate photos into duplicate property.
2. Move plan: work backward from the move date; link each item to a stable box/container ID, a destination and a responsible person. Identify arrival essentials and track packing, transport, access and required services as separate confirmations.
3. Document index: point to the permitted source records, assign owners and relevant renewal dates, and keep private source documents out of broadly shared move lists. The receiving team should have the minimum information needed to retrieve the right item or request the right document.

Copy-ready build brief

Act as my project drafting assistant. Build a restaurant relocation prototype for [role] in [format/tools] using [confirmed asset, destination and document-index inputs]. Use twelve fictional items in two areas and three document categories. Preserve stable asset, box and document IDs through relabeling and destination changes. Deliver an inventory, linked move plan, minimal document index, one complete retrieval trace and a first-use/recovery guide. Test box renumbering, duplicate evidence and an unconfirmed arrival requirement. Do not perform bookings, share real private records or provide technical moving instructions. State which tests were actually run.

Worked restaurant demonstration: expected reasoning

Fictional asset A07, a menu-display frame, belongs to box B03 and destination area "Front desk." Its receipt is document D02, stored outside the shared move list; two photos both refer to A07. A coordinator changes B03's printed display label from "3" to "Reception 1." The stable box ID remains B03, so A07 still resolves to the same box. A receiving worker can search A07, see the current display label and destination, and request D02 from its owner without receiving a receipt archive. A packing-complete status does not establish that a vehicle, venue access or utilities have been confirmed.

Failure simulation and changed information

Seed the source defect by using printed box numbers as the only join key and then renumbering one box. Demonstrate the broken item lookup. Repair the model with an immutable box ID and a separately editable display label, preserving the label-change history. Rerun lookup before and after relabeling and confirm the twelve assets remain twelve. Then change the receiving area and access time for B03. Update the move record and dependent arrival task, flag the obsolete destination label and leave transport/access pending until evidence confirms each one. Do not mark the entire move ready because all boxes are packed.

Handoff and acceptance

Provide a fictional receiving-team view containing box ID, current label, destination, relevant item list and named operational owner. Keep purchase evidence and restricted documents out of that view. State source-edit authority and conflict handling without claiming that a sample spreadsheet implements security. Ask a second user to retrieve A07 after relabeling and identify the unconfirmed arrival dependency. Accept when stable identifiers preserve all links, duplicate evidence creates no duplicate assets, each confirmation has its own state and document references reveal no unnecessary contents. Retain the final artifact version, actual regression record, corrected first-use instruction and next move-check trigger.

Restaurant team travel and off-site logistics

Adapted from combined-master pp 5177-5186; all ten pages reviewed. Restaurant examples and tests are editorial adaptations, not completed source demonstrations. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Connect time, equipment and shared resources

Connect P072 Travel, P073 Trip Preparation and P074 Family Planning as restaurant-team travel, equipment packing and shared-transport coordination. Replace family relationships with named work roles. Keep three fictional days, two travelers and one pending booking; make no live reservations. A suitable use is a restaurant team attending an industry event or organizing administrative travel for an off-site service. The user must distinguish confirmed plans, proposals, missing checks and conflicts involving a shared driver or vehicle. This organizer supplies no safe food-transport or driving procedure.

Inputs and shared records

Travel inputs: confirmed booking evidence, proposed activities, actual local dates, timezones, transfer estimates and team preferences. Preparation inputs: destination, duration, activities, travelers and supplied bag/equipment limits. Team coordination inputs: people, attendance, events, timezones, shared transport and preparation tasks. Use trip, booking, traveler, event and resource IDs. A confirmed state needs its own supporting record; a proposal or enquiry is not that evidence. Enter each event's date/time and timezone once and derive traveler views from it. Store booking references only where needed. An unknown transfer duration or missing resource owner must remain visible.

Connect the three components

1. Itinerary: separate confirmed arrangements from options and verification tasks; include supplied transfer time, preparation and reasonable explicitly stated buffers.
2. Packing: assign necessary items to a traveler or shared kit and bag, with distinct needed, obtained and packed states. A purchased item is not automatically packed.
3. Team coordination: show who attends, who transports and which driver/vehicle each event requires. Review shared-resource conflicts even when different people attend different events. Link each preparation reminder to the relevant event rather than copying dates into unrelated lists.

Copy-ready build brief

Act as my project drafting assistant. Build a restaurant-team travel and packing prototype for [role] in [format/tools], based on [confirmed arrangements and clearly marked proposals]. Use three fictional days, two travelers, shared transport and one pending booking. Connect itinerary, traveler/shared-kit packing and resource assignments through stable IDs. Deliver one record traced from proposal through the planning views, a first-use guide and known limits. Test a proposed booking displayed as confirmed, a changed local departure time and a shared-driver conflict. Do not reserve, purchase or invent travel confirmations. Separate expected behavior from executed tests.

Worked restaurant demonstration: expected reasoning

Fictional traveler T1 attends an off-site briefing at 09:00 local time; traveler T2 has a supplier visit at 09:15. Both records require driver D1. Different attendees do not eliminate the shared-driver conflict. Pending booking BK2 has only a proposal reference, so every dependent view must show pending rather than confirmed. Shared kit K1 contains a display tablet marked obtained, not packed. The packing view therefore still calls for a physical packing confirmation. Trace the same event IDs through itinerary, preparation and transport views so later changes can reach all three.

Failure simulation and changed information

Seed the source defect by defaulting every populated booking row to confirmed, even when its evidence is a proposal. Repair the state rule and retest with a true confirmation, a pending proposal and a cancelled fictional item. Then move BK2's proposed departure from 08:00 to 07:30 local time. Recalculate dependent preparation/transfer checks from the changed event, preserve the old version and show which resources now conflict. Verify the actual travel date and timezone in the supplied record; never infer them from the display order. If transfer time is unknown, identify the unresolved check instead of claiming the revised plan is feasible.

Handoff and acceptance

Create fictional traveler and coordinator views with only necessary operational details. State who may change the booking source, who confirms packed equipment and how resource conflicts are resolved. Ask the second traveler to find what still needs verification and what must be packed without consulting the builder. Accept only if facts, proposals and unknowns are distinct everywhere; purchased and packed remain separate; shared-resource conflicts are visible; and a date/time change propagates without losing history. Deliver the tested files, first-use and recovery instructions, actual failure/regression evidence and a clear unresolved-items list. A planning prototype does not prove that reservations, access or transport are secured.

Restaurant wedding and private-event workspace

Adapted from combined-master pp 5197-5206; all ten pages reviewed. Restaurant examples and tests are editorial adaptations, not completed source demonstrations. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Build one event with independent commitments

Connect P078 Gift Planning, P079 Wedding Planning and P080 Event Planning for a restaurant-hosted wedding meal or private event that includes a client-approved gift or favor task. Keep twelve fictional invitees, one venue option, three costs and two task owners. The user must see which guests responded, which supplier commitments are confirmed, which payments are recorded and which gifts arrived. Replace personal wedding assumptions with the supplied client brief, venue constraints and restaurant responsibilities. Do not place orders, invite guests or commit the venue through this practice build.

Inputs and shared records

Gift inputs: recipient or guest-group reference, occasion date, approved preferences, spending limit and order deadline. Wedding/event-brief inputs: client priorities, guest count, budget, venue constraints and recorded vendor commitments. Event-plan inputs: purpose, location, supplies, setup/service/cleanup tasks and contingency choices. Use event, guest, supplier, cost, gift/order and task IDs. A guest response belongs to a guest record; a payment belongs to its cost/transaction record; a supplier confirmation belongs to its commitment. Keep those sources separate even when they appear together on an event dashboard. Unknown arrival or payment status stays unknown.

Connect the three components

1. Gift/favor plan: select only from the supplied brief and budget, allow recorded lead time, and distinguish chosen, ordered, shipped and delivered.
2. Wedding brief and cost record: record priorities before choices, link the supplied guest count and venue restrictions, and track deposits and remaining balances against the original cost rather than as additional costs.
3. Event run sheet: assign setup, service handoffs and cleanup to the two owners; link confirmations and contingency choices. Display invited, accepted and attended counts separately. Passing a guest count downstream is not a payment or supplier confirmation.

Copy-ready build brief

Act as my project drafting assistant. Create a restaurant wedding/private-event prototype for [role] in [format/tools] using [client brief, venue requirements and confirmed source records]. Use twelve fictional invitees, one venue option, three costs, two owners and one gift/favor delivery record. Connect the brief, gift plan and event schedule through stable IDs while keeping guest, payment and supplier states independent. Deliver a traced event example, first-use guide and limits. Test a guest acceptance wrongly confirming a supplier and a changed guest count. Do not send invitations, purchase gifts, pay deposits or make binding commitments. Distinguish expected results from tests actually performed.

Worked restaurant demonstration: expected reasoning

For fictional event E1, twelve people are invited, eight accept and attendance is not yet known. Supplier S1 has a stated cost of $600, of which a $200 deposit is recorded as paid: the remaining balance is $400 and the total cost is still $600, not $800. Supplier confirmation remains pending if no separate confirmation exists. Gift order O1 is shipped but not delivered. The dashboard should therefore show eight accepted guests, a $400 remaining recorded balance, a pending supplier commitment and a delivery check. None of those states can be inferred from another. Other costs remain separately linked within the three-cost scope.

Failure simulation and changed information

Seed the source defect by allowing an invitation acceptance to set "paid" or "supplier confirmed" on E1. Repair the model by giving guest responses, payments and commitments their own evidence-backed status transitions. Rerun with accepted-but-unpaid and paid-but-unconfirmed fictional cases. Then increase accepted guests from eight to ten while the venue option and supplier terms have not been rechecked. Update attendance planning and flag affected quantities, costs and constraints for review. Do not silently scale a quoted cost or declare extra capacity approved. Record the revised brief version and preserve the earlier eight-person planning state.

Review and second-user handoff

Give one fictional owner guest/task coordination and the other supplier/cost records, with only the operational information each needs. Define source-edit authority and resolve conflicting changes through the event owner; separate sample views do not enforce permissions. Ask a second user to explain why E1 is not fully ready despite eight acceptances. Accept only if guest, payment, supplier and delivery states remain independent, deposits are not double-counted, changed headcount creates the right review tasks and each setup/cleanup responsibility is assigned or visibly unassigned. Deliver the exact tested version, actual defect/retest evidence, first-use corrections and a documented ready/revise/blocked decision.

Restaurant budget, obligations and reserve records

Adapted from combined-master pp 5207-5216; all ten pages reviewed. Restaurant examples and tests are editorial adaptations, not completed source demonstrations. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Keep the existing record model narrow

Connect P083 Budgeting, P084 Debt Management and P085 Saving as a restaurant owner's supplied budget, existing-obligation and earmarked-reserve records. Keep one fictional month, six categories and opening balances. This adapts the source's accounting distinctions only: cash movement, category allocation, stated obligations and savings earmarks. It is not a financial recommendation, bookkeeping certification, borrowing strategy, tax treatment or instruction to move money. Use hypothetical inputs or verified records the user supplies; never invent account access or lender terms.

Inputs and shared records

Budget inputs: period, income actually received, six chosen spending categories, commitments and actual transactions. Existing-obligation inputs: statement date, supplied balance, stated rate, minimum payment, due date and any user-specified hypothetical additional-payment amount. Reserve inputs: goal, target, deadline, current earmarked amount and recorded contribution. Use pseudonymous account IDs, transaction IDs, obligation IDs and goal IDs; omit account numbers and credentials from practice files. Keep statements as balance evidence and transactions as movement evidence. Do not mix opening balances with new income or use a later statement value without its date. Missing terms and unmatched transfers stay unresolved.

Connect the three components

1. Budget record: allocate the supplied available amount to the selected period, compare actuals and show the remaining allocation. Separate cash movement from a spending classification.
2. Existing-obligation record: transcribe supplied statement figures. If a user supplies hypothetical schedules for comparison, label assumptions and display differences without ranking, recommending or promising lender results. Actual fees, accrual, rate changes and payment timing may differ from a simplified model.
3. Reserve record: allocate saved cash to named purposes and reconcile total earmarks with the cash actually recorded. A planned contribution is not automatically a completed transfer; the same money cannot be assigned in full to multiple goals.

Copy-ready build brief

Act as my project drafting assistant. Build a restaurant budget/obligation/reserve record prototype for [role] in [format/tools] from [user-supplied hypothetical or verified figures]. Use one month, six categories and opening balances. Connect source transactions, existing-obligation statements and reserve earmarks through stable IDs. Explain account movement, spending classification, allocations and total balances separately. Deliver a complete fictional reconciliation, a first-use guide and known limits. Test the duplicated savings-transfer defect and a changed statement value. Do not recommend financial actions, infer missing lender terms, access accounts or make payments. Keep predicted arithmetic separate from tests actually run.

Worked restaurant demonstration: expected reasoning

In a fictional record, operating account A opens at $1,000 and reserve account B at $200, so combined recorded cash is $1,200. A $200 internal transfer leaves A at $800 and B at $400; combined cash is still $1,200. The transfer is neither new income nor an external expense. Within B's $400, the user records $250 earmarked for equipment and $150 for a seasonal buffer. Those earmarks allocate the same $400; they do not create another $400 of cash. A separate $100 external expense from A leaves A at $700, B at $400 and combined cash at $1,100. Existing debt balances remain separate from this cash reconciliation.

Failure simulation and changed information

Seed the source defect by recording the $200 transfer as spending and also deducting it from combined cash, or by counting B's receipt as new income. Repair paired transfer classification and account reconciliation without deleting the transaction history. Retest the transfer and the normal $100 expense. Then introduce a corrected fictional obligation statement that changes its balance or includes a fee absent from an earlier hypothetical schedule. Preserve both source dates, flag the old model as unreconciled and update only the supplied figures. Do not hide the difference by changing an interest assumption to force a match. Also test earmarks totaling $450 against $400 of recorded reserve cash; show the $50 over-allocation rather than inventing funds.

Handoff and acceptance

Provide a fictional owner view for balances and a minimal operational view for approved category amounts if needed. Do not distribute real financial identifiers or statements in a shared practice copy. State who may change source records and how disputes are checked; separate sheets alone are not security. Ask a second user to explain why an internal transfer reduces A but not combined cash and why earmarks are not additional assets. Accept only if account totals reconcile, internal transfers are neither new income nor external spending, earmarks do not exceed recorded saved cash, obligations retain their source dates and no result implies guaranteed lender behavior. Retain actual test evidence, first-use corrections and unresolved reconciliation items.

Restaurant creative practice and live-event content

Adapted from combined-master pp 5227-5236; all ten pages reviewed. Restaurant examples and tests are editorial adaptations, not completed source demonstrations. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Use when all three outputs have a restaurant purpose

Connect P081 Hobbies, P099 Music and P100 Creative Writing for a restaurant project that genuinely includes original visual/decor work, music practice and written storytelling, such as an original small live-music evening. Retain three short original outputs, limited weekly time and one revision of each. If live music is not part of the restaurant activity, use the relevant individual modules instead of inventing a musical need. The user must identify what improved in each revision, not merely count finished items.

Inputs and shared records

Visual/decor inputs: intended finished object, materials, available sessions and skill gaps. Music inputs: instrument, a small original target passage, tempo, session length and feedback. Writing inputs: genre, intended reader, premise, length target and draft milestones. Use a shared restaurant-event/project ID with separate output, version and practice-attempt IDs. Keep target skill, success evidence, time allocation, source/rights notes and reviewer feedback attached to each output. One file saved twice is not two original outputs. Missing permission or unknown source status stays unresolved; a practice plan does not establish permission for public performance or publication.

Connect the three components

1. Visual/decor prototype: define a small finish line, prepare the actual materials and make a first version before polishing. Buying supplies is preparation, not completion.
2. Music practice: select one short passage and one specific skill, practice deliberately, record an attempt and compare it with a later attempt under comparable conditions. Minutes practiced are activity evidence, not proof of improved accuracy.
3. Writing: outline a brief original restaurant-event story or introduction, draft it, then revise coherence and structure before sentence polish. A word-count target does not prove that the beginning, purpose and invitation connect. Coordinate all three around the same event brief and time budget rather than assembling unrelated hobbies.

Copy-ready build brief

Act as my project drafting assistant. Build a creative-practice prototype for [restaurant event or program] in [format/tools] using [original brief, available time, materials and rights information]. Connect one small visual/decor output, one original music-practice passage and one short written piece. Give each a target skill, versioned first attempt, feedback and one revision. Deliver the linked practice plan, review evidence, a first-use guide and limits. Test output counts mistaken for improvement and a changed event brief. Do not publish, book performers, claim rights clearance or treat planned comparisons as tests already completed.

Worked restaurant demonstration: expected reasoning

Fictional project C1 is a restaurant's original acoustic evening. Output V1 is a small tabletop event-card prototype with a legibility goal; M1 is an original eight-bar passage with a stated accuracy goal at the same chosen tempo; W1 is a short introduction that must connect the event's theme to the guest invitation. A reviewer identifies unclear visual hierarchy, an inconsistent musical transition and a missing link between the introduction and invitation. Revision 2 addresses those specific issues. Evidence consists of the two card versions, comparable recorded attempts and a before/after structural explanation. "Three outputs made" cannot substitute for those observations, and none of the examples claims a public release.

Failure simulation and changed information

Seed the source defect by declaring success when three files exist, regardless of feedback or change. The review must flag each output without a named skill and observable revision evidence. Repair the completion rule, then compare a genuinely revised output with an unchanged file renamed "final." Next, change the fictional event theme or approved running time. Identify affected visual wording, music length and written structure; issue linked revision tasks while preserving earlier versions. Do not call an increase in practice minutes or word count an improvement unless the relevant quality evidence changes. Keep missing rights/source checks visible rather than treating originality of one component as clearance for all.

Review, handoff and acceptance

Give a fictional creator edit access in the design plan and a restaurant reviewer a feedback/approval view with the necessary brief. Describe how conflicting feedback is resolved without claiming the prototype enforces permissions. Ask the reviewer to identify the changed skill and evidence for all three outputs from the delivered files alone. Accept when the three outputs serve the same restaurant brief, time and materials are realistic, every revision has a specific improvement claim supported by before/after evidence, and rights or source uncertainties remain explicit. Retain the tested artifact version, actual observations, revised first-use instruction and one evidence-based next iteration. Completion demonstrates practice and review, not commercial readiness or guaranteed audience response.

Operational handoffs and minimal sharing

Editorial consolidation of non-care lessons only from combined-master pp 5187-5196 and 5217-5226; full mixed-care capstones are not retained. Restaurant applications and changed examples are editorial adaptations unless identified otherwise.

Editorial extension to P075 duties and P077 vehicle records

A reported completion is not a confirmed completion. For a fictional restaurant duty or vehicle-service record, keep occurrence ID, assigned owner, what was reported, reporter/date, required evidence, confirmation and unresolved question separate. If the owner or evidence is missing, retain "reported; confirmation pending" rather than "done." Test the same report before and after evidence arrives. Discuss task effort and availability rather than assuming equal task counts are fair. A maintenance reminder does not diagnose a vehicle problem. This consolidates the operational handoff lesson from master pp 5187-5196; no animal-care component or full mixed-care project is included.

Editorial extension to P082 restaurant document index

Use the least information needed for the recipient's operational task. In a fictional move or event handoff, share document ID, category, responsible owner and the permitted retrieval route rather than copying private source documents into the practice sheet. Seed an unnecessary private-detail placeholder in a sample export and check that the export rule omits it, while the original source remains controlled. Review both the exact export and actual sharing permissions before any real use; hiding a column is not access control. This consolidates the minimal-sharing lesson from master pp 5217-5226. It does not retain a health-record or medication-management workflow.