# Project Management User Manual — TECHGROUP. Excel App (Detailed)

*Complete, step-by-step user and administrator manual for the Excel-based Project Management application used by the TechGroup company.*

## Table of Contents

[1. Introduction & Purpose 2](#_Toc207195059)

[2. Quick start checklist (pre-use) 2](#_Toc207195060)

[3. Roles & permissions (User vs Admin) 2](#_Toc207195061)

[4. Workbook layout (sheets & tables) 3](#_Toc207195062)

[5. Admin initial setup (one-time tasks) 3](#_Toc207195063)

[6. Opening the UI and authentication flow 4](#_Toc207195064)

[7. Creating a new Project — step-by-step 4](#_Toc207195065)

[8. Adding, editing and deleting lines (staging) — exhaustive 5](#_Toc207195066)

[9. Loading / Searching / Selecting existing projects 7](#_Toc207195067)

[10. Saving / Committing (what happens on Save) 7](#_Toc207195068)

[11. Clearing staging rows (start fresh – RECOMMENDED) 8](#_Toc207195069)

[12. Deleting a project (safe procedure — Admin only) 8](#_Toc207195070)

[13. Reports — Admin reporting and PDF export 8](#_Toc207195071)

[14. Sheet protection & settings (how lockdown works) 9](#_Toc207195072)

[15. Troubleshooting — exhaustive checks & fixes (Developer Only) 10](#_Toc207195073)

[16. Appendix A — Table schemas (detailed) 12](#_Toc207195074)

[17. Appendix B — Important forms, controls & code references 13](#_Toc207195075)

[18. Appendix C — Useful macros & admin commands 13](#_Toc207195076)

[19. Maintenance, backups & recommended workflow 14](#_Toc207195077)

[20. FAQ 14](#_Toc207195078)

## 1. Introduction & Purpose

This Excel workbook implements a lightweight Project Management system tailored for a small industrial welding company. The system is built around:

* A UserForm front-end (VBA) to let non-technical staff create and manage projects and their associated costs.
* Normalized data saved to 'database' sheets (Excel ListObjects) for year-end BI and reporting.
* Staging tables to allow users to add/edit lines before committing them permanently to the DB.
* Admin features: reporting, PDF export, sheet lockdown and secure settings.

Intended users: secretaries and project coordinators who need a simple UI. Administrators: users who configure lookups, manage passwords, and run reports.

## 2. Quick start checklist (pre-use)

Before you start using the application, complete the following checks:

1. Enable macros in Excel: File → Options → Trust Center → Trust Center Settings → Macro Settings → Enable all macros (or 'Disable all except digitally signed macros') as policy allows.
2. Trust access to the VBA project (only required for advanced maintenance): File → Options → Trust Center → Trust Center Settings → Macro Settings → check 'Trust access to the VBA project object model' if you use automation scripts that modify VBA.
3. Run ListAllTables (macro) to confirm required tables exist. If missing, create the tables with the exact names in Appendix A.
4. Populate tblLookups with required lookup values (see Section 5).
5. Add at least 1 company to DB\_Companies and 1 worker to DB\_Workers so combos populate for the first run.
6. Verify Settings (tblSettings): set CurrencySymbol, AllowedFormUsernames if you want to permit direct UI access for certain users, and set Admin password via SetAdminPassword.
7. If you plan to use sheet lockdown, test ApplySheetLockdown on a copy of the workbook first.

## 3. Roles & permissions (User vs Admin)

Overview of roles:

* User (Secretary): Create projects, add staging lines, edit and save projects. Access depends on AllowedFormUsernames or the FormAccessPassword.
* Admin: Configure lookups and settings, reset passwords, access the admin report form, delete projects, and control sheet lockdown. Admin requires AdminPassword.

How the application checks permissions:

- When opening the UI the code runs ShowFormWithFormPassword. The logic is:

* If AllowedFormUsernames contains the Windows username, the UI opens (no password).
* Otherwise, if a FormAccessPassword is set, the user must enter it to open the UI.
* If neither applies, the code falls back to a dialog that lets someone indicate Admin or User; Admin path requires AdminPassword.

## 4. Workbook layout (sheets & tables)

This application uses named Excel tables (ListObjects). DO NOT rename tables unless you update the code. The sheets and table names used by the system are:

* UI — user front sheet (optional).
* README — project documentation sheet.
* Rpt\_Project — temporary report sheet used by the admin report generator.
* DB\_Projects (table: tblProjects) — Project header database.
* DB\_Consumables (table: tblConsumables) — committed consumable lines.
* DB\_Payments (table: tblPayments) — committed payments lines.
* DB\_Logistics (table: tblLogistics) — committed logistics lines.
* DB\_Safety (table: tblSafety) — committed safety lines.
* DB\_Materials (table: tblMaterials) — committed material lines.
* DB\_Companies (table: tblCompanies) — client master data.
* DB\_Workers (table: tblWorkers) — worker master data (optional default rate).
* DB\_Lookups (table: tblLookups) — central lookup table (LookupType, Value).
* Settings (table: tblSettings) — Key / Value settings.
* AuditLog (table: tblAudit) — create/update/delete audit trail.
* Staging\_Consumables (tblStgConsumables), Staging\_Payments (tblStgPayments), Staging\_Logistics (tblStgLogistics), Staging\_Safety (tblStgSafety), Staging\_Materials (tblStgMaterials) — temporary staging tables.

**Why staging exists:**

Staging lets users add lines and preview them before they are made permanent in the DB. When Save is pressed, only staging rows are committed and then cleared. This avoids accidental duplicate insertions and gives the secretary a chance to edit lines before finalizing.

## 5. Admin initial setup (one-time tasks)

Perform these steps once (Admin):

**Open the DB\_Lookups sheet and add required lookup rows. Use table tblLookups with columns:**

LookupType, Value.

Required LookupType values and recommended entries:

* PaymentMethod: Cash, Bank Transfer
* ExpenseCategory: Welding Rods, Gases, Safety, Consumables
* ProjectStatus: Planned, Active, Completed, On Hold
* SafetyCategory: PPE, FireExtinguisher, FirstAid, GasMask, SafetySigns
* MaterialCategory: Pipe, AngleBar, Plate, Bolt, Nut

**Populate DB\_Companies and DB\_Workers:**

* Open DB\_Companies and create at least one company row. Required columns: CompanyID, CompanyName.
* Suggested sample: CompanyID=1, CompanyName='ABC Metals'
* Open DB\_Workers and add at least one worker. Required columns: WorkerID, WorkerName, DefaultRate
* Example: WorkerID=1, WorkerName='John Doe', DefaultRate=1500

**Settings:**

Open Settings sheet (tblSettings). Add or confirm these keys: CurrencySymbol (value e.g. XAF),

AllowedFormUsernames (comma separated Windows usernames), AdminPassword\_Obf (use

SetAdminPassword routine to set), FormAccessPassword\_Obf (optional, use SetFormPassword\_Admin to

set).

Recommendation: Set CurrencySymbol to your company's preferred currency. This populates labels in

small forms.

**Test the environment:**

* Run ListAllTables macro. Confirm no missing tables.
* Open the UI (ShowFormWithFormPassword). Confirm the main form loads and combo boxes (Company, Status) populate.
* Try creating a dummy project, add a consumable, and Save to verify commit works.

## 6. Opening the UI and authentication flow

Detailed flow and behavior when opening the UI:

1. User clicks the **'Add/Load Project**' button on the UI sheet (or the Admin runs ShowFormWithFormPassword macro).
2. The macro reads AllowedFormUsernames and checks the current Windows user (Environ$('USERNAME')). If username present, UI opens without asking password.
3. If username not allowed and FormAccessPassword is set, InputBox prompts for form password (user password). On success the form opens.
4. If form password is not set, the user is asked whether they are Admin. If Admin is chosen an Admin password is requested. Successful admin login grants access.
5. If login fails, the macro shows an error and access is denied.

**Notes on secure storage:**

Passwords are stored obfuscated in Settings (tblSettings). This is not cryptographically secure — it prevents casual reading but is not a replacement for enterprise password management. Admins should treat the AdminPassword as a high-privilege credential and rotate it if compromised.

## 7. Creating a new Project — step-by-step

This section describes creating a new project from scratch, including staging lines and final Save. Follow each step exactly:

1. Open the UI: click on **'Add/Load Project**' and authenticate if asked.
2. On the Project page (first MultiPage tab), fill the Project header fields:

* Project name (txtProjectName) — type the full project title.
* Project code (txtProjectCode) — unique short code (no spaces recommended).
* Client (cmbCompany) — choose the awarding company from the dropdown. If missing, add company in DB\_Companies first.
* Start date (dtStart) — enter a valid date (required). Use format dd/mm/yyyy or your local Excel date format.
* End date (dtEnd) — optional; if left blank project is open-ended.
* Budget (txtBudget) — enter numeric only (no currency symbol). The form will format it on exit. Example: 1200000
* Project manager (txtManager) — name of the responsible person.
* Status (cmbStatus) — choose from ProjectStatus lookups (Planned, Active...).
* Notes (txtNotes) — free text multi-line.

**Validation behavior:**

- Project code is required; the system checks duplication by ProjectCode. If a duplicate is found the user can choose to load the existing project instead of creating a new one.

- Start date is required and must be a valid date.

- Budget must be numeric. If invalid, the save is blocked. Do not append currency symbols in the Budget field.

## 8. Adding, editing and deleting lines (staging) — exhaustive

The application supports five line types: Consumables, Payments, Logistics, Safety, Materials. The pattern is the same for each type; the small difference is the field set. Below are detailed instructions for each type, including control names, validations, and examples.

### 8.1 Consumables

Table: tblStgConsumables (staging) and tblConsumables (DB). Use the MultiPage > Consumables page.

Add a consumable (step-by-step):

1. Click 'Add' on the Consumables page (btnAddConsumable). The small form frm\_ConsumableLine opens.
2. Fill fields on frm\_ConsumableLine:

* Date (txtLineDate) — required. Cannot be a future date. Example: 2025-03-15.
* Category (cmbLineCategory) — choose from ExpenseCategory lookups (Welding Rods...).
* Item description (txtItemDesc) — required. Example: '7018 Rods 2.5mm'.
* Quantity (txtQty) — numeric. Example: 20.
* Unit cost (txtUnitCost) — numeric (formatted on exit). Example: 1500 (Represents unit cost in CurrencySymbol).
* Supplier (txtSupplier) — optional.

Click 'OK' (btnLineOK). The form validates required fields, then writes a new row in tblStgConsumables. The small form hides and frm\_UI.RefreshStagingLists updates the listbox showing the staged line.

Edit a staged consumable: select the row in lstConsumables and click Edit (btnEditConsumable). The same form opens populated. Update fields and click OK to modify staging row.

Delete a staged consumable: select it and click Remove (btnRemoveConsumable). Confirm deletion. If the project is already committed (CurrentProjectID > 0) and editing DB rows, the delete operation will remove the DB row via DeleteConsumable and log an audit entry.

### 8.2 Payments

Tbls: tblStgPayments (staging) and tblPayments (DB). Payment lines are usually labor disbursements to workers.

1. Click Add on Payments page (btnAddPayment).
2. frm\_PaymentLine opens. Fill the fields:

* Date (txtPayDate) — required.
* Worker (cmbPayWorker) — select from DB\_Workers. If Worker has DefaultRate in DB\_Workers, the Rate (txtPayRate) will auto-populate.
* Hours (txtPayHours) — numeric. When entered, Amount (txtPayAmount) auto-calculates: Hours × Rate. You can override Amount.
* Rate (txtPayRate) — numeric.
* Payment method (cmbPayMethod) — from PaymentMethod lookup.
* Notes (txtPayNotes) — optional.

Click OK to add to tblStgPayments. Example: Worker 'John Doe', Date 2025-03-15, Hours 8, Rate 1500, Amount 12000.

### 8.3 Logistics

Tbls: tblStgLogistics and tblLogistics. Used for site travel, transport, rental etc.

1. Click Add on Logistics page (btnAddLogistic).
2. frm\_LogisticsLine opens. Fields: Date (txtLogDate), Category (cmbLogCategory), Description (txtLogDesc), Amount (txtLogAmount), Vendor (txtLogVendor).
3. Click OK to write to staging.

Validation: Amount must be numeric. Category populated from Lookups where LookupType='LogisticsCategory' if present.

### 8.4 Safety

New feature: Safety lines track PPE, safety equipment and consumables dedicated to safety.

1. Click Add on Safety page (btnAddSafety).
2. frm\_SafetyLine opens. Fields: txtLineDate, cmbLineCategory (SafetyCategory lookup), txtItemDesc, txtQty, txtUnitCost, txtSupplier, txtNotes.
3. Unit cost will be formatted automatically; CurrencySymbol label is shown on the form (lblUnitCostCur).
4. Click OK to add to tblStgSafety.

When editing, if mIsDB is True the form writes changes directly to tblSafety using UpdateSafety; if False, it writes to staging.

### 8.5 Materials

Materials track structural stock (pipes, angles) used on projects.

1. Click Add on Materials page (btnAddMaterial).
2. frm\_MaterialLine opens. Fields: txtLineDate, cmbLineCategory (MaterialCategory), txtItemDesc, txtQty, txtUnit (e.g., m or kg), txtUnitCost, txtSupplier, txtNotes.
3. Click OK to write to tblStgMaterials.

Quantity multiplied by UnitCost gives Total if your tblMaterials has a TotalCost column; commit routine will compute if present.

## 9. Loading / Searching / Selecting existing projects

When you need to load a project to view or edit:

1. On Project page use the textSearchProject field and Search/Load button (btnSearchLoad).
2. If textSearchProject is empty and you click Search/Load, the UI will populate the projects ListBox with ALL projects from tblProjects. This lets you browse projects and select one.
3. Select a project from the list and click btnLoadSelected (or equivalent). The form will call LoadProjectIntoForm(ProjectID) and set CurrentProjectID to the selected ProjectID.
4. The staging listboxes will be populated from DB if the project is loaded (these show committed DB rows). If you plan to add new lines for this loaded project, use **Clear Staging** first to ensure new lines are added to staging only, then Save to commit.

**Notes on partial loads:**

- Loading a project does not automatically move DB rows into staging. Committing should only apply to staging rows.

## 10. Saving / Committing (what happens on Save)

Save button (btnSaveProject) handles create and update:

**Create new project (CurrentProjectID = 0):**

1. Click Save. The form first validates header fields (ProjectCode, Company, StartDate).
2. It checks for duplicate ProjectCode. If duplicate exists, you can choose to load the existing project instead.
3. A new ProjectID is generated (NextID('tblProjects','ProjectID')) and a new row is added to tblProjects with the header values.
4. If staging rows exist (HasAnyStagingRows), the user is prompted to confirm committing them. On confirmation CommitStagingToDB is called and each staging table is processed into its DB table. Each created DB row gets an audit entry.
5. Staging rows that were committed are deleted.
6. CurrentProjectID is set to the new ProjectID and the UI refreshes showing DB rows.

**Update existing project (CurrentProjectID > 0):**

1. Click Save. The header row in tblProjects is updated with new values.
2. If staging rows exist, they are committed to the DB under the existing ProjectID (CommitStagingToDB). If staging is empty, the user can choose to proceed with header-only update or cancel.
3. Audit entry written for Update action.
4. UI refreshes.

**Commit routine behavior:**

* CommitStagingToDB calls specific committers for each staging table: CommitStgConsumables, CommitStgPayments, CommitStgLogistics, CommitStgSafety, CommitStgMaterials.
* Each committer loops staging rows from last to first (reverse) to safely delete rows while iterating, creates a new DB row copying fields, computes total costs if necessary, writes an audit record, and deletes the staging row.
* Commit routine shows a summary message with counts of committed rows per type.

## 11. Clearing staging rows (start fresh – RECOMMENDED)

Use this when you load a project but want to start adding new lines cleanly:

1. Click Clear Staging (if present) on frm\_UI, or run the macro ClearAllStaging.
2. The macro deletes all rows in tblStgConsumables, tblStgPayments, tblStgLogistics, tblStgSafety, tblStgMaterials.
3. After clearing, RefreshStagingLists to show only DB rows (if project loaded) or empty lists for staging.

**Important:** clearing staging cannot be undone. Always confirm before running on production data. (It does not delete the Item, It Recommended to Clear Staging so as to not have duplicates on Saving)

## 12. Deleting a project (safe procedure — Admin only)

Project deletion is destructive. Only Admins should do this. Steps:

1. Load the project in frm\_UI and verify ProjectCode and ProjectID.
2. Ensure backups are taken (export DB tables or copy workbook).
3. Click Delete Project (btnDeleteProject). A confirmation dialog should show.
4. If confirmed, the delete routine should:

* Delete rows from tblConsumables, tblPayments, tblLogistics, tblSafety, tblMaterials where ProjectID = target.
* Delete the project row from tblProjects.
* Write an AuditLog entry with Action='Delete'.
* Reset CurrentProjectID = 0 and Refresh UI.

**For Future Updates:** If your workbook does not implement a safe delete, run custom SQL-like deletion macros carefully after backup.

## 13. Reports — Admin reporting and PDF export

Admin report flow: frm\_AdminReport provides options to generate a printable report on **Rpt\_Project** sheet and export to PDF. Steps:

1. Open frm\_AdminReport (admin-only).
2. Select Project from cmbReportProject and click Load.
3. Choose which sections to include: Consumables, Payments, Logistics, Safety, Materials via checkboxes.
4. Set date filters from txtFromDate and txtToDate and optional category/worker filters.
5. Click Generate to build the report on Rpt\_Project. The code writes section headers, rows and totals and formats columns.
6. Click Export PDF to produce a PDF file. The code shows a Save As dialog; choose a path and filename.

**Formatting rules applied by Generate:**

* Section headers are bold and centered.
* Each section includes a header row (Date, Category, Description, Qty, Unit, Amount...).
* All data rows for a section are horizontally centered where appropriate (dates and numeric columns).
* Total rows are inserted after each section with bold formatting, showing the numerical totals for Qty/Amount columns.
* At the end a project summary shows subtotals per section and a grand total.

**Troubleshooting PDF export:**

* If ExportAsFixedFormat throws an error or produces a corrupted PDF: ensure Rpt\_Project is visible when exporting — the routine temporarily shows the sheet then re-hides it.
* Check for hidden shapes, unsupported fonts or very large images that may fail in PDF rendering.
* If Excel's native 'Save as PDF' (File→Save As→PDF) also fails, the issue is with the Excel/PDF stack, not the macro.

## 14. Sheet protection & settings (how lockdown works)

The workbook lockdown functions control which sheets are visible and which are protected. Key routines: ApplySheetLockdown and RemoveSheetLockdown. Important details:

* Allowed sheets array (AllowedSheetsArray) lists sheets that should remain Visible and unprotected — typically UI, README, and Rpt\_Project.
* ApplySheetLockdown sets all non-allowed sheets to VeryHidden and protects them with a password using ws.Protect Password:=pwd, UserInterfaceOnly:=True.
* UserInterfaceOnly flag allows VBA macros to edit protected sheets but prevents manual edits by users. NOTE: UserInterfaceOnly is not persistent between Excel sessions; reapply at Workbook\_Open.
* QuickFix\_UnprotectAllowedThenReapplyLockdown is a helper to unprotect allowed sheets if they were accidentally protected.
* If your code tries to add ListRows on a protected sheet where UserInterfaceOnly is not set, you will get runtime error 1004. Ensure ApplySheetLockdown ran at startup.

**How to configure Allowed sheets (step – Admin only):**

1. Open the AllowedSheetsArray function, add sheet names you want available to users.
2. Run ApplySheetLockdown True to enforce. Verify with ShowSheetVisibilityReport.

## 15. Troubleshooting — exhaustive checks & fixes (Developer Only)

When problems occur, follow this checklist and run the exact diagnostics listed:

### 15.1 Event handler not firing (OK button no action)

Symptoms: clicking OK shows nothing (no message, no staging row).

1) Verify control name: In VBE open the form designer, select the button and check (Name) in Properties. If it's not 'btnLineOK' either rename the control or rename the handler to match.

2) Ensure handler is in the form module: the code window title must show 'frm\_SafetyLine' or 'frm\_MaterialLine'.

3) Use Debug: add MsgBox at top of handler to confirm it fires. If message doesn't appear the event isn't wired.

4) Use DumpFormControls\_Safety / \_Materials macros to list control names and visibility.

5) If the event fires but nothing happens, the code likely hit an error; ensure On Error GoTo ErrHandler is present and check the Err message.

### 15.2 Child form Cancel/X closes main UI

Symptom: child Cancel or clicking X unloads frm\_UI. Cause: child used Unload Me or parent used default-instance. Fix:

1. Make child Cancel use Me.Hide rather than Unload Me.
2. Intercept X via UserForm\_QueryClose and hide the form instead of letting VBA unload it. Example snippet: (provide code)

**Code snippet (paste into child forms):**

*Private Sub UserForm\_QueryClose(Cancel As Integer, CloseMode As Integer)  
 If CloseMode = vbFormControlMenu Then  
 Me.Tag = ""  
 Me.Hide  
 Cancel = True  
 End If  
End Sub*

### 15.3 Runtime error 1004 when adding ListRows

Symptom: error thrown during lr = lo.ListRows.Add. Cause: sheet is protected without UserInterfaceOnly or insertion disabled. Fix:

1. Run ShowSheetVisibilityReport to see protected states.
2. Ensure ApplySheetLockdown is called at Workbook\_Open to set UserInterfaceOnly=True on protected sheets.
3. Temporarily RemoveSheetLockdown, perform the action, and reapply lockdown.

### 15.4 New lines not committed when saving loaded project

This was a common workflow bug. Cause: loaded project's listboxes were populated from DB (for reference) but new lines were added to staging but the form didn't refresh staging listbox or didn't commit because staging appeared empty. Fix:

1. When loading a project and intending to add new lines, run Clear Staging first to remove any leftover staging rows.
2. Add new lines using the Add button so they go to tblStg...
3. Confirm staging tables contain rows (use HasAnyStagingRows or run quick Immediate checks).
4. Click Save to commit staging rows.

### 15.5 Passwords and obfuscation problems

If DeobfuscateString returns gibberish: verify EncodeBase64/DecodeBase64 functions are implemented and symmetric. If you replaced placeholder base64 functions earlier, restore proper implementation or avoid obfuscation until a robust method is in place. Steps to reset Admin password:

1. Use SetAdminPassword 'newpass' to write obfuscated admin password to settings.
2. Verify with: ? DeobfuscateString(GetSetting('AdminPassword\_Obf','')) in Immediate Window.

## 16. Appendix A — Table schemas (detailed)

Exact recommended column names and types. Create each ListObject with these headers in the listed sheet. These exact names are expected by the VBA code unless you adapt it.

### tblProjects (sheet: DB\_Projects)

* ProjectID (Long) — primary key
* ProjectCode (Text) — unique
* ProjectName (Text)
* CompanyID (Long) — FK to DB\_Companies
* StartDate (Date)
* EndDate (Date)
* Budget (Number/Currency)
* ProjectManager (Text)
* Status (Text)
* Notes (Text)
* CreatedBy, CreatedAt, UpdatedBy, UpdatedAt (optional)

### tblConsumables (DB\_Consumables)

* ConsumableID, ProjectID, Date, CategoryID, ItemDescription, Quantity, UnitCost, TotalCost, Supplier, CreatedBy, CreatedAt

### tblPayments (DB\_Payments)

* PaymentID, ProjectID, WorkerID, DatePaid, Hours, Rate, Amount, PaymentMethodID, Notes, CreatedBy, CreatedAt

### tblLogistics (DB\_Logistics)

* LogisticID, ProjectID, Date, CategoryID, Description, Amount, Vendor, CreatedBy, CreatedAt

### tblSafety (DB\_Safety)

* SafetyID, ProjectID, Date, CategoryID, ItemDescription, Quantity, UnitCost, TotalCost, Supplier, Notes, CreatedBy, CreatedAt

### tblMaterials (DB\_Materials)

* MaterialID, ProjectID, Date, CategoryID, ItemDescription, Quantity, Unit, UnitCost, TotalCost, Supplier, Notes, CreatedBy, CreatedAt

### Staging tables (tblStg...)

Each staging table should contain TempID as a unique temporary key and the fields required for that line type. Examples:

* tblStgConsumables: TempID, Date, CategoryID, ItemDescription, Quantity, UnitCost, Supplier, Notes, ProjectID
* tblStgPayments: TempID, WorkerID, DatePaid, Hours, Rate, Amount, PaymentMethodID, Notes, ProjectID
* tblStgLogistics: TempID, Date, CategoryID, Description, Amount, Vendor, ProjectID
* tblStgSafety: TempID, Date, CategoryID, ItemDescription, Quantity, UnitCost, Supplier, Notes, ProjectID
* tblStgMaterials: TempID, Date, CategoryID, ItemDescription, Quantity, Unit, UnitCost, Supplier, Notes, ProjectID

## 17. Appendix B — Important forms, controls & code references

Forms and key control names (copy/paste reference when editing code):

* frm\_UI (main form): txtProjectName, txtProjectCode, cmbCompany, dtStart, dtEnd, txtBudget, txtManager, cmbStatus, txtNotes, lstConsumables, lstPayments, lstLogistics, lstSafety, lstMaterials, btnAddConsumable, btnEditConsumable, btnRemoveConsumable, btnAddPayment, btnAddLogistic, btnAddSafety, btnAddMaterial, btnSaveProject, btnNewProject, btnDeleteProject, btnClearStaging
* frm\_ConsumableLine: txtLineDate, cmbLineCategory, txtItemDesc, txtQty, txtUnitCost, txtSupplier, btnLineOK, btnLineCancel
* frm\_PaymentLine: txtPayDate, cmbPayWorker, txtPayHours, txtPayRate, txtPayAmount, cmbPayMethod, txtPayNotes, btnLineOK, btnLineCancel
* frm\_LogisticsLine: txtLogDate, cmbLogCategory, txtLogDesc, txtLogAmount, txtLogVendor, btnLineOK, btnLineCancel
* frm\_SafetyLine: txtLineDate, cmbLineCategory, txtItemDesc, txtQty, txtUnitCost, txtSupplier, txtNotes, btnLineOK, btnLineCancel, lblUnitCostCur
* frm\_MaterialLine: txtLineDate, cmbLineCategory, txtItemDesc, txtQty, txtUnit, txtUnitCost, txtSupplier, txtNotes, btnLineOK, btnLineCancel, lblUnitCostCur
* frm\_AdminReport: cmbReportProject, btnLoadProject, chkIncludeConsumables, chkIncludePayments, chkIncludeLogistics, chkIncludeSafety, chkIncludeMaterials, txtFromDate, txtToDate, cmbCategoryFilter, btnGenerate, btnExportPDF, lblStatus

## 18. Appendix C — Useful macros & admin commands

Paste these into a standard module and run from the Immediate Window or assign to buttons for admin tasks.

**List all tables:**

*Public Sub ListAllTables()  
 Dim ws As Worksheet, lo As ListObject, out As String  
 out = ""  
 For Each ws In ThisWorkbook.Worksheets  
 For Each lo In ws.ListObjects  
 out = out & "Table: [" & lo.Name & "] on sheet: " & ws.Name & vbCrLf  
 Next lo  
 Next ws  
 MsgBox out, vbInformation, "List of tables"  
End Sub*

**Quick check staging presence (Immediate Window):**

? Not GetTable(tblStgSafety) Is Nothing

? Not GetTable(tblStgMaterials) Is Nothing

**Apply sheet lockdown (example):**

*Public Sub ApplySheetLockdown(Optional userVisibleOnly As Boolean = True)  
 ' Call this from Workbook\_Open after verifying settings  
 ApplySheetLockdown True  
End Sub*

## 19. Maintenance, backups & recommended workflow

Best practices:

* Always work on a copy when changing macros or table schemas. Keep a versioned archive (ZIP) of the workbook before changes.
* Export and save VBA modules externally via VBE for source control.
* Create weekly or monthly CSV exports of DB tables for offsite backups.
* When updating code, test in a 'staging' workbook with sample data before applying to production.
* Reapply sheet protections at Workbook\_Open to ensure UserInterfaceOnly flags exist.

## 20. FAQ

Selected frequently asked questions and answers:

* Q: Can I type currency symbol (e.g., XAF) in Amount fields? A: No — type numeric only; the forms format numbers and show a Currency symbol label. Typing symbols will break validation.
* Q: Why does Add return no effect? A: Check control names and code wiring per Troubleshooting 15.1.
* Q: Why can't I add rows when sheets are protected? A: Ensure ApplySheetLockdown ran and UserInterfaceOnly=True is set; otherwise unprotect sheet before adding.
* Q: How to reset admin password? A: Use the SetAdminPassword routine (Admin-only).