**Inline Inspection Web App – Project Summary**

**1. Business Context & Importance**

* The inline inspection form is the client’s **primary source of quality data**.
* Data is used for reporting, RCA, management reviews, audits, and business growth.
* **Accuracy, traceability, and compliance** (ISO 9001:2015) are critical.
* The current manual process is error-prone, slow, and wastes time.

**2. Form & Data Structure**

* **Form is registered in DCC**; print/export must be “apple to apple” with the official Excel template.
* **Data is entered online** during the shift (no paper notes, no double entry).
* **Each shift**: 3 shifts/day (A, B, C), 3 machines.
* **Each lot**: Contains 13, 21, or more rolls; multiple lots per shift.
* **Each roll**: Has a unique traceability code.

**3. Traceability Code**

* **Format:** YYMMDDMS LL-RR
* YY = Year (last two digits)
* MM = Month
* DD = Date
* M = Machine number
* S = Shift number
* LL = Lot number
* RR = Roll position within the lot
* YYMMDDMS is unique per shift/machine/day; LL-RR repeats per lot.
* **Full code is unique for every roll** and must be shown in UI and export.

**4. Data Entry & UI**

* **Table/grid interface** (Excel-like) for entering lots and rolls.
* **Touch-friendly** for future tablet use.
* **Manual entry** of operator, QC, and supervisor names for each lot/shift.
* **Accept/Reject/Rework/KIV**: Four checkboxes per roll (only one can be checked at a time).
* **Defect Name**: Operator types, with autocomplete from a standardized list (provided by client).
* **Remarks**: Optional free-text field.
* **Blank fields** allowed as per real process.

**5. Validation**

* Numeric fields (e.g., weights) must be within valid ranges.
* If a roll is marked as Reject, Rework, or KIV, Defect Name is required.
* Only one status (Accept/Reject/Rework/KIV) can be selected per roll.

**6. Editing & Audit**

* Data can be edited/corrected before export (role-based controls to be added later).
* Audit trail and locking after export are important for future phases.

**7. Export & Print**

* **Export to Excel** using the official DCC template.
* All data, including calculated totals, must be filled in the correct cells.
* Printout must be “apple to apple” with the registered form.
* Export can be triggered at any time; file is offered for download.
* No need for PDF or in-browser preview; client will print from Excel.

**8. Storage & Compliance**

* Data is stored in Supabase (Postgres).
* 1GB storage is sufficient for thousands of forms.
* Regular backups and ISO 9001:2015 compliance required.
* No need for file storage or offsite backup in phase 1.

**9. User Management**

* Shift-based login for now; operator/QC/supervisor names entered manually.
* Roles (operator, supervisor, admin) and audit trail to be added after MVP.

**10. Device & Language**

* Must work on desktop and be touch-friendly for tablets.
* English (UK) only.

**11. Reporting & Analytics (Future)**

* Centralized data enables instant reports, summaries, and analytics.
* Can add dashboards, graphs, and custom exports in future phases.

**12. Other Notes**

* No notifications, email/SMS, or ERP integration needed in phase 1.
* No multi-language or “nice to have” features for now.
* All requirements are focused on compliance, accuracy, and ease of use.

**Next Steps**

* Step-by-step development: UI wireframe → database schema → data entry grid → validation → Excel export → testing → role controls → audit trail (future).
* Use React (recommended) + Supabase for scalable, maintainable code.
* Use openpyxl (Python) or exceljs (Node.js) for Excel export, depending on backend.

**Inline Inspection Web App – 5-Day Step-by-Step Task Plan**

**Day 1: Project Setup & Database Design**

**1.1. Project Initialization**

* Set up a new project directory (React recommended).
* Initialize with create-react-app (or Vite, or Vue CLI if you prefer Vue).
* Set up version control (Git).

**1.2. Supabase Setup**

* Create a new Supabase project.
* Set up authentication (basic, shift-based login for now).
* Create database tables:
* lots (date, shift, machine, lot number, operator/QC/supervisor names, etc.)
* rolls (lot\_id, roll position, all inspection fields, status, defect name, remarks, etc.)
* defect\_names (for autocomplete)
* Set up Supabase client in your frontend.

**1.3. Excel Template Preparation**

* Obtain the official DCC-registered Excel template.
* Place it in your backend or a secure location for export use.

**Day 2: UI/UX – Data Entry Grid & Form**

**2.1. UI Wireframe/Mockup**

* Design the main data entry page:
* Shift/machine/lot selection at the top.
* Table/grid for roll entry (Excel-like).
* Manual entry fields for operator, QC, supervisor names.

**2.2. Implement Data Entry Grid**

* Use a React table/grid component (e.g., AG Grid, React Table, or plain HTML table).
* Add rows dynamically for each roll in a lot.
* For each roll:
* Input fields for all required data (weights, positions, etc.).
* Four checkboxes for Accept/Reject/Rework/KIV (only one selectable).
* Defect name field with autocomplete (from defect\_names table).
* Remarks field (optional).

**2.3. Touch-Friendly Design**

* Ensure all controls are large and easy to use on tablets.

**Day 3: Validation & Data Handling**

**3.1. Field Validation**

* Numeric fields: enforce valid ranges (e.g., weights > 0).
* If Reject/Rework/KIV is checked, Defect Name is required.
* Only one status checkbox can be checked per roll.

**3.2. Traceability Code Generation**

* Auto-generate the traceability code (YYMMDDMS LL-RR) for each roll and display it in the grid.

**3.3. Save/Edit/Delete Functionality**

* Allow saving lots/rolls to Supabase.
* Allow editing/deleting rolls before export (with future role controls in mind).

**Day 4: Excel Export & Calculation**

**4.1. Backend for Excel Export**

* Set up a backend endpoint (Node.js with exceljs or Python with openpyxl).
* On export, fetch the lot and roll data from Supabase.

**4.2. Fill Excel Template**

* Programmatically fill the Excel template with all data.
* Write calculated totals (accept, reject, rework, KIV, weights) into the correct cells.
* Fill in traceability codes and all required fields.
* Leave blanks where appropriate.

**4.3. Download Functionality**

* Serve the filled Excel file for download to the user.
* Test printout to ensure “apple to apple” match with the DCC template.

**Day 5: Testing, Polish, and Documentation**

**5.1. End-to-End Testing**

* Test the full workflow: data entry → validation → export → print.
* Check for edge cases (blank fields, max/min values, etc.).
* Ensure touch usability on tablet and desktop.

**5.2. Client Review & Feedback**

* Demo the app to the client.
* Make any urgent tweaks based on feedback.

**5.3. Documentation**

* Write a user guide for operators and supervisors.
* Document the codebase and deployment steps.

**5.4. Backup & Compliance Check**

* Ensure Supabase backups are enabled.
* Double-check compliance with ISO 9001:2015 requirements.

**Bonus (If Time Allows)**

* Add a simple dashboard for searching/filtering past lots/rolls.
* Prepare for future features: role-based access, audit trail, analytics.

**Summary Table**

| **Day** | **Task Area** | **Key Deliverables** |
| --- | --- | --- |
| 1 | Setup & DB Design | Project, Supabase, tables, Excel template ready |
| 2 | UI/UX | Data entry grid, touch-friendly, operator fields |
| 3 | Validation & Data Logic | Field checks, trace code, save/edit/delete |
| 4 | Excel Export | Backend, fill template, download, print test |
| 5 | Testing & Docs | Full test, client review, docs, compliance check |