Lead Tracking System - Step-by-Step Implementation Guide

A concise 5■page step-by-step process to complete your Salesforce Lead Tracking System (Phases 1–10). Use this as a checklist during implementation and handover.

Quick Project Checklist

1. Prepare sandbox and production org details
2. Confirm stakeholders and acceptance criteria
3. Collect sample data for migration and testing
4. Define SLAs, KPIs, and reporting needs

Phase 1 — Org Set-up & Configuration (Step-by-step)

1. Create / refresh a sandbox from Production for development.
2. In Production, set Company Information, Fiscal Year, Business Hours, and Holidays.
3. Define Roles, Profiles, and Permission Sets; create core admin and user accounts.
4. Configure Org-wide defaults and sharing rules for Lead and related objects.
5. Enable features (Lightning Experience, My Domain) and set login security policies.
6. Set up connected apps, SSO (if required), and IP restrictions.

Phase 2 — Data Modeling & Relationships (Step-by-step)

1. Identify entities: Leads, Accounts, Contacts, Opportunities, Campaigns, and any custom objects.
2. Create custom objects and custom fields with clear API names and descriptions.
3. Choose relationship types (Lookup vs Master■Detail) and create lookup/master-detail fields.
4. Create Record Types, Page Layouts, and compact layouts for different user profiles.
5. Add validation rules and picklist value sets; document field-level security and required fields.
6. Create a sandbox test dataset and verify relationships with sample records.

Phase 3 — Process Automation (Step-by-step)

1. Map manual processes to automated flows: lead assignment, routing, auto-response, and status updates.
2. Prefer Salesforce Flow (Record-Triggered Flow) for new automation; use Scheduled Flows for batch jobs.
3. Build flows in a developer sandbox: start with a design (diagrams) and mock data.
4. Add fault paths, decision elements, and bulk-safe design for record-triggered flows.
5. Test flows with different user profiles and bulk loads; capture logs for failed runs.
6. Move to staging and then deploy using change sets or CI/CD after UAT sign-off.

Phase 4 — Apex Programming (Step-by-step)

1. Identify business logic that cannot be handled by Flow and requires Apex (complex validations, callouts).
2. Create Apex classes with single responsibility and triggers that delegate to handler classes.
3. Follow best practices: bulkify, avoid SOQL/DML inside loops, and handle governor limits.
4. Write comprehensive test classes covering positive, negative, and bulk scenarios; aim for >75% coverage.
5. Use Static Resources, Custom Metadata, or Custom Settings for configurable values.
6. Perform code review and run static analysis tools (PMD, ESLint for Lightning) before deployment.

Phase 5 — User Interface Development (Step-by-step)

1. Design Lightning App(s) and navigation based on user personas and tasks.
2. Create Lightning record pages, app pages, and utility bar items using App Builder.
3. Use Lightning Web Components (LWC) for custom UI features; keep components small and reusable.
4. Configure compact layouts and highlight panels to surface key lead info.
5. Implement quick actions, global actions, and pre-filled forms for common tasks.
6. Run usability testing sessions and iterate on layouts; gather screenshots and notes.

Phase 6 — Integration & External Access (Step-by-step)

1. Identify integration points: marketing automation, telephony, ERP, or external lead sources.
2. Choose integration method: REST API / SOAP / Platform Events / Salesforce Connect.
3. Create Connected App and configure OAuth scopes; use Named Credentials for authentication.
4. Implement middleware (Mulesoft, Dell Boomi, or custom) for complex transformations if needed.
5. Monitor integrations with Platform Events, debug logs, and external system dashboards.
6. Secure external access using IP restrictions, connected app policies, and secure headers.

Phase 7 — Data Management & Deployment (Step-by-step)

1. Prepare data mapping templates and conduct test migrations with a subset of records.
2. Create duplicate/ matching rules and cleansing scripts to ensure data quality.
3. Use Data Loader, Data Import Wizard, or ETL tools for bulk import; validate with checksum samples.
4. Configure change management: use Sandboxes, Change Sets, or a Metadata CI/CD pipeline (Git, SFDX).
5. Create rollback plans and backup exports before Production deployment.
6. Perform UAT, gather sign-offs, and schedule Production cutover during low usage windows.

Phase 8 — Reports & Dashboards (Step-by-step)

1. Collect reporting requirements and define KPIs (lead conversion rate, response time, pipeline value).
2. Build Custom Report Types that join Leads with related objects if needed.
3. Create sample reports for different personas and pin necessary list views.
4. Design dashboards (dynamic where possible) and schedule refreshes and subscriptions.
5. Train users on filters, views, and exporting data for advanced analysis.

Phase 9 & 10 — Final Presentation, Demo Day & Closure (Step-by-step)

1. Prepare a demo script aligned to acceptance criteria and select representative sample records.
2. Show end-to-end lead lifecycle: capture → enrichment → assignment → conversion → reporting.
3. Capture stakeholder feedback and create an action log for post-launch items.
4. Deliver admin and end-user training; provide quick-reference guides and recordings.
5. Finalize documentation (architecture, data dictionary, runbooks) and hand over credentials securely.
6. Schedule the post-implementation review and define support SLAs.

Handoff & Post-Launch Checklist

1. Complete deployment checklist and confirm all automated jobs are active.
2. Share system admin contact, escalation matrix, and support procedures.
3. Ensure scheduled backups and health monitoring are set up.
4. Collect UAT sign-offs and archive project artifacts in a central repository.

Prepared by: Project Implementation Team — Lead Tracking System