Phase 0: Project Planning & Multi-Company Architecture Design

Implementation Guide for Visual Drag-and-Drop ERP

Duration: 4–6 weeks
Version: 1.0
Date: October 2025

1. Executive Summary

Phase 0 establishes the foundation for your open-source, visual, drag-and-drop ERP system with embedded AI and multi-company capabilities. This phase focuses on strategic planning, architecture design, and team alignment to ensure successful implementation.

Key Objectives

- Define project scope, success metrics, and governance structure
- Design multi-company architecture with proper data isolation
- Select technology stack and deployment strategy
- Create master implementation timeline
- Establish risk management and change control processes

2. Project Charter

2.1 Vision Statement

Develop an open-source, visual ERP platform that empowers SMEs to manage multiple companies through an intelligent, no-code interface with embedded AI assistance, requiring no dedicated server infrastructure.

2.2 Success Criteria

- **User Adoption:** 80% of target users actively using the system within 3 months
- **Data Migration:** 95% accuracy in data migration from legacy systems
- **Performance:** System response time under 2 seconds for 100 concurrent users
- **AI Accuracy:** 85% relevance score for AI-generated responses
- **Multi-Company:** Seamless inter-company transactions with zero data leakage

2.3 Project Scope

In Scope

- Core platform with modular architecture
- 15+ business modules (Finance, Inventory, Sales, HR, etc.)
- Multi-company support with consolidation
- Intelligent data migration engine
- No-code form and module builder
- Open-source AI companion (Rasa + local LLM)
- Embedded PostgreSQL database
- Progressive Web App (PWA) capabilities

Out of Scope (Phase 0-1)

- Mobile native applications
- Third-party marketplace integration
- Advanced AI features (beyond basic NLP)
- Real-time video communication

3. Stakeholder Analysis

3.1 Key Stakeholders

Role	Responsibilities	Engagement Level
Project Sponsor	Budget approval, strategic alignment	High
Project Manager	Planning, coordination, risk management	Daily
Technical Lead	Architecture, technology decisions	Daily
Business Analysts	Requirements, process mapping	High
Development Team	Implementation, testing	Daily
End Users	UAT, feedback, adoption	Medium → High
IT Operations	Infrastructure, security	Medium

3.2 Communication Plan

- **Weekly Steering Committee:** Project status, budget, risks
- **Daily Stand-ups:** Development team progress
- **Bi-weekly Demos:** Stakeholder feedback sessions

- **Monthly Town Halls:** Organization-wide updates

4. Multi-Company Architecture Design

4.1 Architecture Strategy Decision Matrix

Approach	Data Isolation	Complexity	Cost	Scalability	**Recommendation**
Shared DB + Company ID	Medium	Low	Low	High	✅ **Recommended**
Separate Schema per Company	High	Medium	Medium	High	Alternative
Separate DB per Company	Very High	High	High	Medium	Enterprise Only

4.2 Recommended Architecture: Shared Database with Company ID

Rationale:

- Simple to implement and maintain
- Cost-effective for SMEs managing 2-50 companies
- Scales well with proper indexing
- Easy to add new companies without infrastructure changes

Implementation Details:

Database Schema Design

```
-- Company Master Table
CREATE TABLE companies (
    id SERIAL PRIMARY KEY,
    code VARCHAR(10) UNIQUE NOT NULL,
    name VARCHAR(255) NOT NULL,
    parent_company_id INTEGER REFERENCES companies(id),
    legal_name VARCHAR(255),
    tax_id VARCHAR(50),
    currency_code VARCHAR(3) DEFAULT 'BDT',
    fiscal_year_start DATE,
    is_active BOOLEAN DEFAULT true,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Example: Multi-company transactional table
CREATE TABLE invoices (
    id SERIAL PRIMARY KEY,
    company_id INTEGER NOT NULL REFERENCES companies(id),
    invoice_number VARCHAR(50) NOT NULL,
    customer_id INTEGER NOT NULL,
    invoice_date DATE NOT NULL,
    total_amount DECIMAL(15,2),
    status VARCHAR(20),
    created_by INTEGER,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    UNIQUE(company_id, invoice_number)
);

CREATE INDEX idx_invoices_company ON invoices(company_id);
```

Application-Level Security

- **Middleware:** Automatic company context injection
- **ORM Filters:** Default company_id filtering on all queries
- **API Gateway:** Company validation on every request
- **Row-Level Security:** PostgreSQL RLS policies as backup

4.3 Multi-Company Feature Requirements

Company Hierarchy

- Parent-subsidiary relationships (unlimited depth)
- Consolidation groups for reporting
- Company-specific configurations (currency, fiscal calendar, tax rules)

Inter-Company Transactions

- Inter-company sales and purchases
- Inventory transfers between companies
- Automatic elimination entries for consolidation
- Inter-company pricing rules

Access Control

- User assignment to one or multiple companies
- Role-based permissions per company
- Cross-company roles (Group Controller, Group Admin)
- Company switcher UI component

Consolidated Reporting

- Group-level financial statements
- Drill-down to company level
- Inter-company elimination tracking
- Multi-currency consolidation

5. Technology Stack Selection

5.1 Backend Stack

Component	Technology	Justification
Web Framework	Django 4.2+ or FastAPI	Your expertise, mature ecosystem, excellent ORM
Database	PostgreSQL 15+ (Embedded)	ACID compliance, JSON support, mature
API Layer	Django REST Framework or FastAPI	RESTful APIs, auto-documentation
Task Queue	Celery + Redis	Background jobs, data migration
Event Bus	Redis Pub/Sub or RabbitMQ	Inter-module communication
Caching	Redis	Performance optimization

5.2 Frontend Stack

Component	Technology	Justification
Framework	React 18+ or Vue 3+	Component-based, rich ecosystem
UI Library	Ant Design or Vuetify	Enterprise UI components
State Management	Redux Toolkit or Pinia	Centralized state, dev tools
Drag-and-Drop	React DnD or Vue Draggable	Rich interaction
Charts	Chart.js or Apache ECharts	Data visualization
Build Tool	Vite	Fast development experience

5.3 AI Stack (Open-Source)

Component	Technology	Justification
Conversational AI	Rasa Open Source	No limits, self-hosted
LLM	LLaMA 2/3 or Mistral	Local inference, no API costs
Vector DB	Chroma or Qdrant	Document embedding, RAG
NLP	Hugging Face Transformers	Model ecosystem
Semantic Search	Haystack	Document Q&A

5.4 Infrastructure

Component	Technology	Justification
OS	Ubuntu 22.04 LTS	Stability, long-term support
Containerization	Docker	Isolation, portability
Web Server	Nginx	Reverse proxy, static files

Component	Technology	Justification
Process Manager	Supervisord or systemd	Service management
Monitoring	Prometheus + Grafana	System metrics

6. Development Methodology

6.1 Agile Framework

- **Sprint Duration:** 2 weeks
- **Ceremonies:** Daily stand-up, sprint planning, retrospective, demo
- **Tools:** Jira/GitHub Projects, Confluence/Notion

6.2 Development Workflow

1. **Feature Branch:** Develop in feature branches
2. **Code Review:** Peer review required before merge
3. **Automated Testing:** Unit tests (80%+ coverage), integration tests
4. **Continuous Integration:** GitHub Actions or GitLab CI
5. **Deployment:** Staged (Dev → UAT → Production)

7. Risk Management

7.1 Risk Register

Risk	Impact	Probability	Mitigation Strategy
Data migration failures	High	Medium	AI-assisted mapping, phased migration, rollback capability
Multi-company data leakage	Critical	Low	RLS policies, automated testing, security audits
AI inference performance	Medium	Medium	Model optimization, GPU acceleration, caching
User adoption resistance	High	Medium	Change management, training, early user involvement
Scope creep	Medium	High	Change control board, documented requirements
Hardware limitations (100 users)	High	Medium	Performance benchmarking, resource monitoring, upgrade plan

8. Budget and Resource Planning

8.1 Team Structure

Core Team (Phase 0-1):

- 1 Project Manager
- 1 Technical Architect
- 2 Senior Backend Developers (Django/FastAPI)
- 2 Frontend Developers (React/Vue)
- 1 AI/ML Engineer
- 1 Database Administrator
- 1 DevOps Engineer
- 2 QA Engineers
- 1 Business Analyst
- 1 UX/UI Designer

8.2 Hardware Budget (100 Users)

Development Environment:

- 3 Development Workstations: $3,000–$5,000 each

Production Server (Self-Hosted PC):

- CPU: Intel i9-13900K / AMD Ryzen 9 5900X (~$500–$600)
- RAM: 64GB DDR4 (~$200)
- Storage: 2TB NVMe SSD (~$150)
- GPU: NVIDIA RTX 4070 (~$600)
- Case, PSU, Cooling: ~$300
- **Total:** ~$1,750–$2,000

Network & Backup:

- UPS (2000VA): ~$300
- Network Switch: ~$100
- External Backup Storage: ~$200

8.3 Software Licensing

- All components open-source: **$0**
- Optional: Support contracts for enterprise features (future)

9. Master Implementation Timeline

Phase 0: Planning & Architecture (4–6 weeks)

- Week 1–2: Stakeholder alignment, requirements gathering
- Week 3–4: Architecture design, technology stack finalization
- Week 5–6: Team onboarding, environment setup

Phase 1: Platform Foundation (8–10 weeks)

- Week 7–10: Core platform, multi-company data layer
- Week 11–14: Authentication, RBAC, API gateway
- Week 15–16: Initial testing and documentation

Subsequent Phases

- Phase 2–10: See main project timeline (52–78 weeks total)

10. Deliverables Checklist

Phase 0 Deliverables

- ☐ Project Charter (signed)
- ☐ Multi-Company Architecture Document
- ☐ Technology Stack Specification
- ☐ Risk Register
- ☐ Master Project Timeline
- ☐ Budget and Resource Plan
- ☐ Communication Plan
- ☐ Change Control Process
- ☐ Development Environment Setup Guide
- ☐ Database Schema Design (draft)

11. Success Metrics (Phase 0)

Metric	Target	Measurement
Stakeholder Alignment	100% approval	Charter sign-off
Architecture Review	Pass technical review	Architect approval
Team Onboarding	100% team ready	Skills assessment

Metric	Target	Measurement
Environment Setup	All devs operational	Successful local builds
Timeline Approval	Approved by sponsor	Signed timeline document

12. Next Steps

Upon completion of Phase 0:

1. Conduct Phase 0 review meeting with all stakeholders
2. Obtain formal approval to proceed to Phase 1
3. Initialize code repository with project structure
4. Begin Sprint 1 of Phase 1 (Platform Foundation)

Appendices

Appendix A: Glossary

- **RBAC:** Role-Based Access Control
- **RAG:** Retrieval Augmented Generation
- **LLM:** Large Language Model
- **PWA:** Progressive Web App
- **UAT:** User Acceptance Testing

Appendix B: References

- Django Documentation: https://docs.djangoproject.com
- PostgreSQL Multi-tenancy Patterns
- Rasa Open Source Documentation
- ERP Implementation Best Practices (2025)

Document Control:

- **Version:** 1.0
- **Author:** ERP Project Team
- **Approved By:** Project Sponsor
- **Date:** October 2025
- **Next Review:** Upon Phase 0 completion