Inventory Management System Development Plan

1. System Overview

Problem Statement

Multi-location spare parts inventory management for field service engineers servicing machines across customer sites, with complex ownership and movement tracking requirements.

Key Stakeholders

- **Engineers**: Field service personnel with personal inventory
- **Administrator**: Manages central warehouse and overseas ordering
- **Manager**: Requires territory-wide visibility
- **System**: Tracks parts across multiple locations and ownership states

2. Core Requirements Analysis

2.1 Inventory Types & Locations

- **Customer Sites**: Site-specific and common parts
- **Central Warehouse**: General stock, repaired parts (grey stock), FE consignment stock
- **Engineer Personal Inventory**: Parts at home/in vehicle
- **In-Transit**: Parts being moved between locations

2.2 Part States & Ownership

- **Active**: Ready for use
- **Consumed**: Used in repairs
- **To Spares**: Replenishment stock
- **Grey Stock**: Repaired/refurbished parts
- **FE Consignment**: Customer-owned parts in central warehouse
- **On Order**: Parts being procured

3. System Architecture

3.1 Technology Stack Recommendation

- **Frontend**: React/Next.js with TypeScript (mobile-first responsive design)
- **Backend**: Node.js with Express or Python with FastAPI

- **Database**: PostgreSQL (handles complex relationships well)
- **Authentication**: JWT with role-based access control
- **API**: RESTful API with OpenAPI documentation
- **Reports**: PDF generation (jsPDF/PDFKit), CSV export
- **Deployment**: Cloud platform (AWS/Azure/GCP) with containers

3.2 Database Schema Design

Core Entities

```
Users
- id, email, name, role, territory, active_status

Stores
- id, name, type, location, customer_id, owner_id, description

Parts
- id, part_number, description, manufacturer, category, unit_cost

Inventory
- id, store_id, part_id, quantity, status, ownership_type, work_order_id

Work Orders
- id, customer_id, machine_id, engineer_id, status, created_date

Inventory Movements
- id, from_store_id, to_store_id, part_id, quantity, movement_type, work_order_id, created_by, timestamp

Reports
- id, report_type, parameters, generated_by, created_date, file_path
```

4. User Roles & Permissions

4.1 Role Definitions

- **Super Admin**: Full system access
- **Manager**: Territory-wide read access, reporting
- **Administrator**: Central warehouse management, ordering, all store visibility
- **Engineer**: Personal inventory + assigned customer sites
- **View Only**: Read-only access to specific stores

4.2 Permission Matrix

Action	Engineer	Admin	Manager	Super Admin
View own inventory	✓	✓	✓	✓
View all inventory	-	✓	✓	✓
Create/Edit parts	-	✓	-	✓
Move inventory	✓*	✓	-	✓
Generate reports	✓*	✓	✓	✓
User management	-	-	-	✓

*Limited to assigned stores/territory

5. Development Phases

Phase 1: Foundation (Weeks 1-3)

Deliverables:

- Database schema implementation
- User authentication & authorization
- Basic CRUD operations for Users, Stores, Parts
- Role-based access control
- Mobile-responsive UI framework setup

Technical Tasks:

- Set up development environment
- Database migration scripts
- JWT authentication middleware
- Basic API endpoints for core entities
- React components for login/registration

Phase 2: Core Inventory Management (Weeks 4-6)

Deliverables:

- Inventory tracking system
- Store management interface
- Part allocation to stores

- Basic inventory movements
- Search and filtering capabilities

Technical Tasks:

- Inventory CRUD operations
- Store selection and filtering UI
- Part search with autocomplete
- Inventory movement logging
- Stock level calculations

Phase 3: Advanced Features (Weeks 7-9)

Deliverables:

- Work order integration
- Complex inventory movements (consume, to spares, grey stock)
- FE consignment stock tracking
- Inventory transfer workflows
- Mobile optimization

Technical Tasks:

- Work order entity implementation
- Movement workflow engine
- Status change tracking
- Mobile-first UI refinements
- Offline capability consideration

Phase 4: Reporting & Analytics (Weeks 10-11)

Deliverables:

- Ad-hoc report generation
- CSV/PDF export functionality
- Dashboard with key metrics
- Inventory alerts and notifications

Technical Tasks:

- Report builder interface
- PDF/CSV generation services
- Dashboard charts and widgets
- Email notification system
- Automated reporting schedules

Phase 5: API & Integration (Weeks 12-13)

Deliverables:

- Complete REST API documentation
- API rate limiting and security
- Integration testing suite
- Mobile app preparation

Technical Tasks:

- OpenAPI specification
- API versioning strategy
- Rate limiting implementation
- Comprehensive API testing
- Mobile app API endpoints

Phase 6: Testing & Deployment (Weeks 14-15)

Deliverables:

- Production deployment
- User acceptance testing
- Training documentation
- System monitoring setup

Technical Tasks:

- Production environment setup
- End-to-end testing
- Performance optimization
- Backup and recovery procedures

- Monitoring and logging

6. Key Features Specification

6.1 Store Management

- Create stores with types: Customer Site, Engineer Personal, Central Warehouse
- Assign ownership and access permissions
- Geographic location tracking
- Store capacity and utilization metrics

6.2 Inventory Operations

- **Stock In**: Receive parts into any store
- **Stock Out**: Remove parts for consumption or transfer
- **Transfer**: Move parts between stores
- **Consume**: Mark parts as used against work orders
- **To Spares**: Convert ordered parts to site inventory
- **Grey Stock**: Mark repaired parts as available

6.3 Reporting Capabilities

- **Stock Levels**: Current inventory by store/part/territory
- **Movement History**: Audit trail of all inventory changes
- **Consumption Analysis**: Usage patterns by site/engineer
- **Order Tracking**: Status of parts on order
- **Custom Reports**: Ad-hoc reporting with filters

6.4 Mobile Features

- Barcode scanning for part identification
- Quick stock updates via mobile interface
- Offline capability for remote locations
- GPS integration for location verification

7. Technical Considerations

7.1 Data Integrity

- Inventory transaction logging for audit trail

- Concurrent update handling
- Data validation at multiple layers
- Backup and recovery procedures

7.2 Performance

- Database indexing strategy
- Caching for frequently accessed data
- Pagination for large datasets
- API response optimization

7.3 Security

- Role-based access control
- API authentication and authorization
- Data encryption in transit and at rest
- Input validation and sanitization

7.4 Scalability

- Microservices architecture consideration
- Database horizontal scaling options
- CDN for static assets
- Container orchestration

8. Risk Assessment & Mitigation

8.1 Technical Risks

- **Complex inventory state management**: Implement comprehensive testing
- **Mobile responsiveness**: Progressive web app approach
- **Data synchronization**: Implement optimistic locking

8.2 Business Risks

- **User adoption**: Involve stakeholders in UI/UX design
- **Data migration**: Plan for gradual rollout with parallel systems
- **Training requirements**: Create comprehensive documentation

9. Success Metrics

9.1 Technical KPIs

- API response time < 200ms
- 99.9% uptime
- Mobile page load time < 3 seconds
- Zero data loss incidents

9.2 Business KPIs

- Inventory accuracy > 95%
- Time to locate parts reduced by 50%
- Stock-out incidents reduced by 30%
- User adoption rate > 80%

10. Post-Launch Roadmap

10.1 Phase 2 Enhancements

- Native mobile application
- Advanced analytics and ML predictions
- Integration with ERP systems
- IoT sensor integration for automatic updates

10.2 Long-term Vision

- Predictive maintenance part ordering
- Supplier integration and automatic reordering
- Advanced reporting with business intelligence
- Multi-tenant architecture for franchise expansion

11. Budget & Resource Estimation

11.1 Development Team

- 1 Full-stack Developer (15 weeks)
- 1 UI/UX Designer (4 weeks, part-time)
- 1 DevOps Engineer (3 weeks, part-time)

11.2 Infrastructure Costs

- Cloud hosting: $200-500/month
- Database: $100-300/month
- Third-party services: $100/month
- SSL certificates and domains: $50/year

11.3 Total Estimated Cost

- Development: $75,000 - $120,000
- Infrastructure (first year): $4,500 - $9,500
- Maintenance (annual): $15,000 - $25,000

This plan provides a solid foundation for building your inventory management system. Would you like me to elaborate on any specific phase or create detailed wireframes for the user interface?