TECH RULES & TECHNOLOGY STACK DOCUMENT
Project: MedSure — Fake Drugs Detection + QR Verification (Supply Chain)
Version: v1.0
Date: 2026-01-28

1) Purpose
This document defines the technical rules and technology stack to ensure consistency, compatibility, maintainability, and scalability across the MedSure platform.

2) Platforms & Applications
A) Consumer Verification App
- Mobile Web (PWA-ready)
- Android app (optional wrapper)

B) Business Portals
- Pharmacy/Distributor Portal (Web)
- Manufacturer Portal (Web)
- Admin/Regulator Dashboard (Web)

3) Architecture Overview
- Client Apps (Web/Mobile)
- Backend API (REST)
- Auth & RBAC
- Database + Cache
- Blockchain layer (permissioned)
- Observability (logs/metrics/tracing)

4) Recommended Tech Stack

4.1 Frontend (Web)
- Framework: Next.js (React, TypeScript)
- Styling: Tailwind CSS
- UI Components: shadcn/ui
- Forms: React Hook Form + Zod validation
- Data fetching: TanStack Query
- State: React Context + Query cache

Rules:
- TypeScript required for all UI code
- Component-driven architecture
- Reusable UI components for badges, cards, tables

4.2 Mobile
- Primary: Mobile Web (responsive)
- Optional native: React Native (Expo)
Rules:
- Shared design system with web
- Camera + QR scanning must be fast and reliable

4.3 Backend
- Language: Node.js (TypeScript)
- Framework: NestJS (preferred) or Express
- API style: REST (v1), OpenAPI documented
- Validation: Zod/Joi/class-validator
- Background jobs: BullMQ (Redis)

Rules:
- API versioning required (/api/v1)
- DTO validation mandatory
- All endpoints must return consistent JSON error formats

4.4 Database & Storage
- Primary DB: PostgreSQL
- ORM: Prisma
- Cache: Redis
- File storage: S3-compatible storage (AWS S3 / MinIO)

Rules:
- Migrations required (no manual schema drift)
- Use UUIDs for public identifiers
- Encrypt sensitive fields where needed

4.5 Blockchain Layer
- Type: Permissioned blockchain (recommended)
Options:
- Hyperledger Fabric
- Quorum
- Corda (optional)

Approach:
- On-chain: batch_id, manufacturer_id, mfg/expiry, event hashes, recall flags
- Off-chain: documents, scan analytics, attachments

Rules:
- Blockchain is append-only (no edits; corrections via new events)
- Sign transactions with org keys
- Maintain an off-chain indexed mirror for fast reads

4.6 Authentication & Security
- Auth: JWT + refresh tokens
- RBAC: role-based permissions
Roles:
- Consumer (read-only)
- Pharmacy/Distributor (event writer)
- Manufacturer (batch creator)
- Admin/Regulator (recall & audits)

Rules:

- TLS everywhere
- Rate limiting for scan endpoints
- Audit logs for every write action
- QR payload must be signed (anti-tamper)

4.7 QR Verification Standard
QR encodes a secure URL:
- https://medsure.app/v/{batch_id}

Rules:
- Batch ID must be immutable
- Verification must check: existence, expiry, recall, metadata match
- Store scan logs (anonymized where possible)

4.8 Observability & DevOps
- Containerization: Docker
- CI/CD: GitHub Actions
- Hosting: AWS / GCP / Azure (cloud-agnostic)
- Monitoring: Prometheus + Grafana (or Datadog)
- Logging: ELK stack or cloud logging

Rules:
- Infrastructure as Code (Terraform recommended)
- Environments: dev / staging / production
- Automated tests required in CI pipeline

5) Engineering Standards

5.1 Code Quality
- Linting: ESLint
- Formatting: Prettier
- Commit convention: Conventional Commits
- Branching: GitFlow or trunk-based

5.2 Testing
- Unit tests: Jest
- API tests: Supertest
- E2E tests: Playwright

Rules:
- Minimum coverage target 70% for backend services
- Critical flows must have E2E coverage:
  - scan -> verification -> result
  - create batch -> generate QR
  - recall -> invalid scan result

5.3 Documentation
- OpenAPI/Swagger required
- README + runbooks for deployment

- Database schema diagram maintained

6) Scalability Requirements
- Support millions of scans/day
- Response time < 2 seconds for verification
- Horizontal scaling for API and read replicas for DB
- Caching for repeated batch lookups

7) Security Requirements
- Protect against QR replay attacks via scan anomaly detection
- Prevent unauthorized event creation via RBAC
- Daily backups and disaster recovery plan
- Compliance-friendly audit trails

8) Deliverables
- Tech stack confirmation
- System architecture diagram
- API spec (OpenAPI)
- Database schema (Prisma)
- Deployment pipeline + IaC