Real-Time Chat Application

Technical Product Plan (NestJS + Next.js)

1. Project Vision

- Build a production-ready real-time chat platform demonstrating scalable architecture and modern backend/frontend patterns.
- Focus on performance, clean architecture, modular design, and secure communication.

2. Tech Stack

- Backend: NestJS, PostgreSQL, Prisma ORM, WebSockets (Gateway)
- Frontend: Next.js (App Router), TypeScript, Tailwind CSS
- Authentication: JWT + Refresh Tokens
- Testing: Jest (Unit + Integration)

3. Core Features

- User authentication and authorization (RBAC ready)
- Private and group conversations
- Real-time messaging via WebSockets
- Replies (threaded messages)
- Reactions (emoji-based)
- Typing indicators
- Online / offline presence
- Delivered & Seen status per user
- Message pagination
- File upload support
- Rate limiting for API protection

4. Database Architecture

- Users
- Conversations

- Participants (many-to-many)
- Messages (with parent_id for replies)
- Reactions
- MessageStatus (delivered / seen tracking)
- Soft delete support
- Indexes for performance optimization

5. Architecture Strategy

- Feature-based modular structure in NestJS
- Gateway layer for real-time events
- Service layer for business logic
- DTO validation using class-validator
- Repository pattern via Prisma
- Global exception filters & interceptors

6. Performance & Scalability

- Database indexing on high-traffic fields
- Efficient message pagination (cursor-based ready)
- Rate limiting middleware
- Optimized WebSocket event emission

7. Testing Strategy

- Unit testing for services
- Integration testing for message flows
- Validation testing for authentication and permissions

8. Deliverables

- Public GitHub repository with clean commit history
- Structured branches per feature
- Comprehensive README documentation
- ERD diagram and database schema
- Deployment-ready configuration