SKEET — Full System Architecture Report

Overview:
SKEET (Statistically Known Entry & Exit Tactics) is a multi-user trading automation and analytics platform. It integrates AI/ML compute, exchange connectivity, and real-time analytics with a scalable architecture built for both crypto and stock markets.

Layer	Responsibility	Technology Stack
Frontend	User dashboards, strategy creation, analytics, subscription	React, Lovable, Tailwind, Supabase JS, Recharts
Backend	Auth, database, event relay, user management	Supabase (Postgres, Realtime, Auth, Storage)
Compute Layer	AI/ML models, CCXT order execution, backtests	Python, FastAPI, Celery/Ray, ccxt, pandas, PyTorch
Queue/Messaging	Dispatch trading and ML tasks	Redis / NATS / Supabase Webhooks
Monitoring	Track bot metrics, logs, PnL	Prometheus, Grafana, Supabase Logs
Payments	Subscription billing and account tiers	Stripe API + Supabase Edge Functions

Key Components:
• Multi-user authentication with role-based access control
• AI/ML trading engine cluster (Python FastAPI)
• Supabase backend for real-time data and event tracking
• Stripe billing for subscription monetization
• Containerized microservices for AI, ML, and CCXT execution
• Secure encrypted API key vault (pgcrypto)
• Realtime WebSocket updates for trades and performance

Folder-Level Blueprint (Developer Reference)

```
skeet-platform/
■■■ frontend/ (React + Lovable)
■ ■■■ components/
■ ■■■ pages/
■ ■■■ utils/
■ ■■■ styles/
■
■■■ backend/ (Supabase Edge Functions)
■ ■■■ functions/
■ ■■■ auth/
■ ■■■ routes/
■
■■■ ai-server/ (Python FastAPI)
■ ■■■ main.py
■ ■■■ trading/
■ ■ ■■■ ccxt_handlers.py
■ ■ ■■■ risk_manager.py
■ ■ ■■■ order_executor.py
■ ■■■ ai/
■ ■ ■■■ model_training.py
■ ■ ■■■ feature_engineering.py
■ ■ ■■■ prediction_service.py
■ ■■■ database/
■ ■ ■■■ supabase_client.py
■ ■■■ workers/
■ ■ ■■■ celery_worker.py
■ ■■■ config.json
■
■■■ infra/
■ ■■■ Dockerfile
■ ■■■ docker-compose.yml
■ ■■■ deploy_scripts/
■
■■■ README.md
```

Next Steps:
1. Convert current Supabase schema to multi-user structure.
2. Set up AI server (FastAPI + Celery) and connect via Supabase webhooks.
3. Implement Redis for job queuing and scaling.
4. Connect Stripe for billing and subscription management.
5. Deploy frontend on Vercel and backend compute cluster on Render/AWS.