**🚀 Дорожная карта AI-Трейдера**

**✅ Спринт 1. Базовая инфраструктура и запуск (P0)**

**Цель:** система стабильно работает в demo-режиме (Binance Testnet), полный цикл «анализ → сделка → отчёт».

**Что сделать:**

1. Проверка запуска через docker-compose, автоперезапуски (supervisord).
2. Настроить .env, вынести ключи в защищённое хранилище.
3. Исправить баланс/позиции в UI (routers/trading\_exec.py, HTMX-partials).
4. Включить risk-gate (2%/6%) как обязательную проверку в services/trading\_service.py.
5. Реализовать SL/TP при входе (OCO).
6. Добавить e2e-тест на Testnet (tests/test\_e2e\_testnet.py).

📌 Результат: можно торговать на testnet без ручного вмешательства.

**⏳ Спринт 2. Анализ рынка и генерация сигналов (P0–P1)**

**Цель:** система принимает решения на основе индикаторов и базовых правил.

**Что сделать:**

1. Расширить src/indicators.py: Bollinger Bands, Elder Force Index.
2. Добавить src/features/patterns.py: свечные модели (Hammer, Doji, Engulfing).
3. Реализовать мультифреймовый анализ (тройной экран Элдера).
4. Сбор новостей через news/rss\_client.py, добавить простейший sentiment score.
5. Конфиг ансамбля стратегий (configs/strategy.yaml), агрегатор сигналов в src/strategy.py.

📌 Результат: торговые сигналы формируются по набору правил (теханализ + паттерны + sentiment).

**⏳ Спринт 3. Риск-менеджмент и сопровождение сделок (P0–P1)**

**Цель:** торговля строго контролируется, позиции управляются автоматически.

**Что сделать:**

1. Интегрировать дневные лимиты (state/daily\_limits.py) с UI.
2. Расширить risk/trailing.py: частичное закрытие, динамический трейлинг.
3. Добавить логику реверса позиций при противоположном сигнале.
4. В services/reconcile.py — автопроверка открытых позиций и SL/TP.
5. Учёт комиссий в PnL (обновить templates/monitor/\_metrics.html).

📌 Результат: система сама управляет рисками и корректно сопровождает сделки.

**⏳ Спринт 4. Модели ML и интеллектуальные сигналы (P1–P2)**

**Цель:** повысить качество сигналов с помощью ML и мета-лейблинга.

**Что сделать:**

1. Модуль ML (src/models/):
   * LSTM/Transformer для прогноза цены/волатильности.
   * Random Forest/XGBoost для фильтрации сигналов.
   * DRL-агент для обучения стратегии.
2. Подготовка данных: fractional differentiation, triple-barrier labels.
3. Валидация моделей с Purged Walk-Forward CV.
4. Автоматическое переобучение раз в неделю (скрипт + cron).

📌 Результат: система генерирует более точные сигналы, фильтрует шумовые.

**⏳ Спринт 5. Мониторинг и алерты (P1)**

**Цель:** трейдер всегда видит текущее состояние и получает уведомления.

**Что сделать:**

1. Завершить веб-дашборд (templates/monitor/\*): PnL, Sharpe, drawdown.
2. Подключить Telegram-бота (monitoring/alerts.py).
3. Генерация дневных/недельных отчётов (CSV/HTML/PDF).
4. Автоматический backup сделок и логов (backup/).
5. Метрики ресурсов (CPU/RAM) через psutil + Prometheus.

📌 Результат: полная прозрачность, контроль и уведомления.

**⏳ Спринт 6. Автовосстановление и Fail-Safe (P2)**

**Цель:** система устойчива к сбоям и перезапускается автоматически.

**Что сделать:**

1. Автовосстановление позиций после сбоя (services/reconcile.py).
2. Dead-man switch с закрытием всех позиций при критических ошибках.
3. Фолбэк-стратегия «закрыть всё и ждать», если ML-модель не отвечает.
4. Проверка интернет-соединения и автоматический reconnect.

📌 Результат: даже при сбое или потере связи система остаётся управляемой.

**⏳ Спринт 7. Безопасность и аудит (P1–P2)**

**Цель:** ключи и данные защищены, все действия можно восстановить.

**Что сделать:**

1. Шифрование ключей (utils/secrets.py → DPAPI/Keyring).
2. Двухуровневое подтверждение для массовых действий («закрыть всё»).
3. Логи и аудит действий (какой сигнал → какое решение).
4. Маскирование конфиденциальных данных в UI.

📌 Результат: безопасная система с полным журналом для аудита.

**🎯 Итог**

* **Спринт 1–3 (P0):** закрывают полный цикл торговли на testnet → можно реально проверять сделки.
* **Спринт 4–7 (P1–P2):** повышают интеллект (ML), мониторинг, отказоустойчивость и безопасность.