Объединённый документ для Голиба и Алекса

Голиб, вот API-ключ для Groq, нужно встроить в мозг:

GROQ\_API\_KEY=gsk\_3bWk5uhQNJqDe21sgFbSWGdyb3FYbIaxxRuEhLl5sB4MHt4LK27Z

✅ Проверка (curl)

curl https://api.groq.com/openai/v1/chat/completions -s \

-H "Content-Type: application/json" \

-H "Authorization: Bearer $GROQ\_API\_KEY" \

-d '{

"model": "meta-llama/llama-4-scout-17b-16e-instruct",

"messages": [{

"role": "user",

"content": "Explain the importance of fast language models"

}]

}'

✅ Задача

Добавить ключ в secrets.env.

Настроить LLM-router так, чтобы:

Короткие задачи (memeability, фильтрация, классификация постов, антиспам) шли через Groq LLaMA-3 70B (latency <0.3s).

Длинные пояснения/отчёты → через основной LLM (Llama3.3/OpenAI).

В конфиге:

llm\_router:

short\_form: groq\_llama3\_70b

long\_form: llama3\_70b\_open

thresholds: { max\_tokens\_short: 512, max\_latency\_ms: 300 }

⚡ Итог: Groq используем как «ускоритель» для коротких задач, чтобы мозг был быстрым и дешёвым.

ЗАДАЧИ

вот чёткий список задач для Голиба по встраиванию Groq и роутингу ИИ.

1) Какие задачи гоним через Groq (быстро/дёшево)

Классификация постов: has\_token / will\_token / noise

Экстракция сущностей: тикер/имя/хэштеги/упоминания

Memeability/score (Z-метрика + короткое объяснение)

Антиспам/антибот: дубликаты, стоп-слова, cooldown-флаги

Ранжирование кандидатов (top-K соц-сигналов)

Нормализация/тональность (sentiment короткой фразы)

Правила CELEB\_OVERRIDE (распознать «из белого списка»)

Короткие резюме для карточки (≤ 2–3 строки)

2) Что оставить на основной LLM (длинные)

Deep-объяснения и отчёты (для DOCX/MD)

Сложные выводы при merge\_insights (соц + ончейн)

Генерация развернутых причин/рекомендаций (BUY/WATCH/NO)

Любые ответы >512 токенов или где нужна высокая точность

3) Роутер ИИ (реализация)

Конфиг:

thresholds:

max\_tokens\_short: 512

max\_latency\_ms: 300

Правила:

если tokens\_input ≤ 512 → Groq

если > 512 или нужен детальный reasoning → long\_form

таймаут Groq >300ms → fallback на long\_form

Параметры Groq-модели: meta-llama/llama-4-scout-17b-16e-instruct (или согласованный эквивалент), temperature низкая (0.2–0.4).

4) Инфраструктура

secrets.env: GROQ\_API\_KEY=...

Health-check эндпоинт: /health/llm (показывает latency Groq/long\_form)

Логи роутера: trace\_id, model\_used, latency\_ms, tokens

Rate-limit и backoff при 429/5xx, не более 2 rps в пике (конфигурируемо)

5) Точки интеграции (в мозге)

Social pipeline:

classify\_post, extract\_entities, score\_meme → Groq

explain\_alert, compose\_report → long\_form

Antibot pipeline:

normalize\_text, spam\_detect, dup\_hash\_reason → Groq

Enrichment:

short\_summary\_for\_card (<=200–300 токенов) → Groq

6) Тесты и критерии приёмки

Юнит-тест: 20 коротких постов → ответ ≤300ms, точность ≥ Голиб-бейзлайна

Fallback: выключить Groq → роутер уводит на long\_form без падений

Нагрузка: 100 req/min (батчем) — без ошибок, ровный latency

Логи: для каждого алерта видно, какая модель выбрана и почему

Конфиг-флаг USE\_GROQ=true/false — горячее переключение

7) Безопасность ключа

Ключ хранить только в secrets.env, не логировать

Проверка на «утечки»: warn если ключ в конфиге/логе

Если всё это сделает — мозг станет быстрым и дешёвым на коротких задачах и точным на длинных.

5 функций, которые Голибу нужно добавить сразу, чтобы Алексу потом было легко стыковать. Коротко, чётко, с I/O и критерием приёмки. (Да — Groq тоже включаем: роутинг LLM через Groq для быстрых задач.)

1) /ingest/social\_alert (v1) — приёмник событий Голиба → Алексу

Зачем: единый формат алертов в систему Алекса.  
Вход (POST JSON):

{

"type":"social\_alert","version":"1.0","alert\_id":"...","source":"twitter|tg|discord|reddit",

"celeb":false,"text":"...","hints":{"ticker":"XYZ","mint":null},

"social\_score":0.0,"labels":["SPAM","CELEB\_OVERRIDE"],

"metrics":{"freq\_1m":0,"freq\_5m":0,"z\_meme":0.0,"tier\_weight":1.0},

"ts":1725

}

Выход: 200 {"ok":true,"id":"<alert\_id>"} (идемпотентность по alert\_id).  
Критерий: Алекс принимает alert → стартует resolve→snapshot→security.

2) /probe/telegram\_reaction — добивка реакции из TG по алерту

Зачем: после зачистки слов оценить, как отреагировали каналы в TG.  
Вход: alert\_id, keywords[]  
Выход: TG-метрики врезкой к SocialAlert:

{"alert\_id":"...", "tg":{"channels\_hit":N,"freq\_5m":M,"sentiment":0.0}}

Критерий: при наличии упоминаний ≥ порога → повышает приоритет токена (добавляется в SocialAlert.enrichment).

3) Whales 30d модуль (локальная агрегация)

Зачем: топ-100 китов за 30 дней + by\_token.  
API:

GET /whales/top?period=30d&limit=100

POST /whales/by\_token {"mint":"...","window":"7d|30d"}  
Логика: Helius/SolanaFM swaps → связка buy→sell (FIFO) → win\_rate, PEL, median\_hold\_days, pnl\_pct.  
Критерий: отдаёт стабильный JSON; по test-mint показывает вошедших китов с суммой входов.

4) Risk Management Core (fail-closed + override policy)

Зачем: единые правила безопасности, чтобы UI Алекса мог строго блокировать.  
Функция: POST /risk/evaluate {"mint":"...","snapshot":{...}}  
Выход:

"risk":"red|yellow|green",

"critical":[...], "warnings":[...],

"policy":{"buy\_allowed":false,"needs\_override":true,"required":{"sl":0.03,"slippage":0.5,"maxBuyPctLP":1.0}}

Критерий: при критике → buy\_allowed=false, override только с обязательными параметрами.

5) LLM Router с Groq (быстрый / дешёвый)

Зачем: маршрутизация ИИ-задач:

быстрые классификации/экстракции → Groq LLaMA-3 70B (низкая задержка),

генеративные пояснения/длинные отчёты → основной LLM (Llama 3.3-70B / OpenAI).  
Конфиг:

Критерий: задачи <512 токенов идут через Groq; latency снижается, качество сохраняется.

Что сказать Голибу про Groq

Да, обязательно скажи. Пусть добавит LLM-router и креды Groq в secrets.env (GROQ\_API\_KEY) + health-check. Это часть «мозга» (быстрые классификаторы и memeability по Groq → дешево и шустро).

Мини-чек-лист без дыр

POST /ingest/social\_alert с идемпотентностью.

TG-reaction probe в обогащение SocialAlert.

/whales/top и /whales/by\_token (win\_rate, PEL, hold).

/risk/evaluate с критическими/предупредительными флагами и policy.

LLM-router с Groq (конфиг + fallback).

Логи и метрики по каждому шагу (trace\_id/alert\_id).

Документация 1 стр. с форматами JSON для Алекса.

ИИ каждые 5 минут вынимает из твитов 200 знаменитостей мем-кандидаты из текста, ALT-текста и OCR с картинок. Берём короткие, звучные, легко-тикеризуемые штуки.

A) Имена/клички/прозвища (паттерны + примеры)

Паттерны: Mr|Mrs|Sir|Lord|Lady|Queen|King|Lil|Big|Baby|Tiny|Grandpa|Grandma|Uncle|Aunt|Dr|Professor

Примеры: Mr Nibbles, Lady Peanut, Sir Whiskers, King Acorn, Lil Paws, Baby Toast, Fluffy, Goober, Bubbles, Pickles, Tofu, Nacho, Biscuit, Noodle, Waffles, Mochi, Nugget, Sprout, Pebble, Poppy, Pudding, Snickers, Jelly, Cookie, Muffin, Cupcake

B) Животные/виды (расширенный список)

dog, puppy, cat, kitten, frog, toad, raccoon, squirrel, duck, goose, owl, penguin, bear, panda, koala, lion, tiger, cheetah, panther, wolf, fox, otter, beaver, mouse, rat, hamster, capybara, hedgehog, porcupine, bat, dolphin, shark, whale, seal, walrus, crocodile, alligator, snake, lizard, gecko, turtle, tortoise, crab, lobster, shrimp, octopus, squid, jellyfish, bee, wasp, ant, fly, mosquito, butterfly, moth, beetle, ladybug, spider, scorpion, snail, slug, camel, donkey, horse, cow, pig, goat, sheep, chicken, rooster, turkey, dragon, unicorn, phoenix (мифология тоже годится)

C) Предметы/еда/бренды (то, что часто «взрывается»)

Фрукты/сладости/еда: banana, apple, orange, lemon, lime, grape, mango, pineapple, coconut, avocado, watermelon, cherry, berry, strawberry, blueberry, raspberry, donut, bagel, croissant, muffin, cupcake, cake, pie, cookie, biscuit, brownie, pizza, burger, hotdog, taco, burrito, nacho, fries, nugget, ramen, sushi, cheese, bacon, peanut, almond, cashew, walnut, pistachio, acorn, coffee, latte, espresso, tea, soda, cola, juice  
Быт/гаджеты/игрушки: toaster, blender, microwave, fridge, vacuum, robot, rocket, drone, laser, lego, slime, joystick, cassette, floppy  
Рандом-вкусности: glitter, confetti, bubble, rainbow, unicorn dust

D) Звуки/ономатопея/вирус-шум

bonk, mew, meow, woof, bark, arf, ribbit, croak, quack, honk, oink, moo, neigh, baa, hiss, roar, rawr, squeak, squeal, buzz, beep, boop, ding, dong, bing, bong, zap, pow, bam, boom, pop, fizz, smack, clack, clonk, yeet, uwu, owo, meep, derp, brrr, zing, zzz

E) Мем-лексика/ярлыки (короткие «ударные»)

based, giga, sigma, chad, goated, mid, bruh, lol, lmao, cope, seethe, npc, skibidi, rizz, gyatt, ohio, grimace, mog, zoomer, cringe, kino, drip, sus, sneed

F) Короткие лозунги/фразы (что часто порождает тикер)

let him cook, IYKYK, this is fine, don’t sleep, don’t miss, send it, make it go brrr, printing, wen, wen lambo, wen moon, gm, gn, we ball, touch grass, it’s happening, no brakes, absolute unit, king of X, queen of X, certified banger, peak

G) Структурные подсказки «это можно в тикер»

ALL-CAPS слова, сдвоенные буквы (FLUFFY → FLUF/FLFY), обрезки гласных (SQUIRREL → SQRL), добавление Y/X (NUT → NUTX / NUTY), рифма/аллитерация (Acorn/Acorny), цифры в конце (DOGE2), эмодзи-слово (ACORN 🥜).

H) Мем-картинки/ALT/текст на изображении (vision+OCR)

Шаблоны: Pepe/Wojak/Chad/NPC/Popcat/Doge/Grimace, Distracted Boyfriend, Galaxy Brain, Slaps Roof of Car, Surprise Pikachu.  
Объекты/животные на фото (совпадение с B): белки, ракуны, лягушки, коты/собаки, утки, совы, капибары и т.д.  
Текст на картинке (OCR-паттерны): крупные слова-лейблы (ACORN KING, NUT LORD, MEW TIME), наклейки типа NO BRAKES, 100X, PEAK, BASED.

I) Эмодзи-кластеры (усиливают «мем-скор»)

🚀 🌕 🔥 ⚡ 💎 🐋 🐸 🐱 🐶 🐿️ 🦝 🦄 🐝 🍌 🍩 🍪 🍕 🍔 ☕ 🧃 🧠 👑 📈 😹 🤯 🤡

J) Нормализация/фильтрация (чтобы не тащить мусор)

Удалить стоп-слова: today, morning, great, happy, check, link, subscribe, meeting, policy, vote, report

Оставить только кандидаты длиной 2–12 символов (под тикер/бренд), без чистых ссылок/упоминаний людей, если это не кличка/мем.

Дедуп по лемме и близости (NUT ~ NUTS ~ NUTTY).

K) Ско́ринг (что даёт шанс «родить» монету)

Memeability (0–1): коротко, звучно, смешно, легко писать в капсе.

Animal/Onoma бонус (+0.3): животные/звуки чаще становятся тикерами.

Novelty (0–1): не встречалось у автора 30 дней.

Celebrity weight (0–1): вес аккаунта из твоего списка.

Engagement z-score: лайки/RT/ответы за 15–30 мин vs медиана автора.

Echo-use: ≥3 уникальных автора повторили слово/образ за 15–30 мин.  
→ Порог эскалации: memeability ≥ 0.6 и (echo ≥ 3 или author ∈ trusted).

Частота: 5 мин → 1 мин (сужаем обзор, следим пристально)

Базово: каждые 5 мин собираем все A–K, считаем скор, отбрасываем стоп-лист.

Как только слово/картинка начинает подхватываться (см. порог), переводим его в watch-лист и сужаем частоту до 1 мин, но только для этих кандидатов.

В 1-мин режиме трекаем: повторения слова/варианты тикеров ($FLUF/$SQRL/$NUT), мем-картинки-эхо, рост z-score.

Если интерес затухает 15–20 мин → возвращаем в 5-мин режим.

Дообучение (что Голиб должен заложить сразу)

Чтобы система училась на лету от твоих загрузок и примеров:

Учебные входы

Тексты/таблицы: словари удачных мем-семян (питомцы/слова), «анти-семена» (шутки, что не стали токеном).

Картинки/мем-паки: примеры с разметкой, где «seed-текст» на картинке (OCR target).

Видео: авто-транскрибирование (ASR) → прогон через те же A–K.

Авто-индексация и обновление словарей

Любой загруженный файл → сразу в память (векторная БД) и в словарь-кандидатов (Animals, Objects/Food, Onomatopoeia, Meme\_words), плюс обновление стоп-листа.

Кнопки «👍 seed» / «👎 not-seed» — твоя метка идёт в active-learning и меняет веса Memeability.

Интерфейсы (для Голиба)

/upload (doc/pdf/img/mp4) → индекс сразу, без рестартов.

/teach/seed (JSON) → добавляет новые семена/синонимы/варианты тикера.

/teach/denylist → пополняет стоп-слова/образы.

/export\_snapshot / /import\_snapshot → перенос знаний между серверами.

Конфиг порогов (memeability, echo, окна 15–30 мин) редактируется без выката кода.

Метрики качества (ежедневно)

Precision/Recall по последним 7 дням на вручную размеченных кейсах.

Среднее время: «твит → seed-детект», «seed → watch-лист 1 мин».

Топ-20 мем-семян дня с причинами (кто произнёс, цитаты, изображения-эхо).

Helius

Да разрешено создавать несколько APIключей через панель (Dashboard) — ты можешь сделать для разработки, теста, продакшена и выдать каждому свою копию.

В Dashboard можно нажать "Create New API Key" и дать ему метку (например, “Голиб test” или “Алекс dev”) .

Chainbase

В бесплатном тарифе указывается, что он поддерживает до 3 проектов, что позволяет иметь отдельные ключи под разные задачи.

Также упоминается JWT-авторизация под каждый ключ + IP-белый список — всё для безопасности и разграничения доступа .

Birdeye

Их Dashboard позволяет управлять ключами, подписками и лимитами, включая создание нескольких API-ключей и отслеживание использования.

В бизнестарифе доступен режим "Unlimited API Keys", но в бесплатной версии скорее всего также несколько ключей доступно, просто лимиты общие .

Dexscreener

Тарифы включают бесплатный план, но детализация по количеству ключей не ясна. Чаще всего один ключ на аккаунт, но можно уточнить в их Dashboard.

Вывод для тебя:

Что это даёт

Алекс и Голиб не будут «съедать» твой лимит параллельно — каждый работает под своим ключом.

Если один ошибётся или превысит лимит — остальные останутся живы и работают.

Возможно мониторить, кто сколько запросов сделал — удобнее для анализа и контроля.

ТЗ для Голиба — Супер ИИ «Радар нарративов + база знаний»

🎯 Цель

Создать ИИ-систему, которая:

В реальном времени отслеживает соцсети (Twitter/Telegram/Discord через DataPulse) и ончейн (Helius, Birdeye).

Выявляет нарративы, мемные слова, картинки и события, которые запускают мемкоины.

Сшивает соцсигналы с ончейном (mint/listing/ликвидность/входы китов).

Считает скоринг (точность ≥ 80%), фильтрует скам (RugCheck/GoPlus/Chainbase).

Запоминает успешные кейсы, дообучается на новых данных (книги, PDF, графики, стратегии).

Выдаёт алерты в JSON и DOCX (для МемкаБота, Телеги и веб-панели).

1. Источники данных (ingest)

Соцсети: DataPulse → X (Twitter), Telegram, Discord (200 Tier A/B аккаунтов + сигнальные группы).

Ончейн (Solana): Helius (TOKEN\_MINT, poolCreated, крупные транзакции), Birdeye (листинги, OHLCV, ликвидность).

Резерв: Dexscreener API.

Формат входа: JSON батчи (author, followers, text, media, ts, link).

2. Фильтры и словари

whitelist: животные, клички, мемные слова, ономатопея, лозунги, эмодзи (готовые seed\_whitelist.txt).

vision anchors: классы объектов и мем-шаблонов (vision\_anchors.txt).

stoplist: спам/общие слова (today, happy, giveaway…).

regex: тикеры, CA.

обновление: через веб-панель или загрузку txt/json.

3. Многоступенчатая фильтрация

5 минут: поиск мем-семян в твитах (слова/картинки).

1 минута: ускоренный мониторинг кандидатов, если ≥3 авторов подхватили.

Финальный отбор: проверка Discord/TG + хайп (эхо, тикеры в ответах).

4. Сшивка «соцсети ↔ ончейн»

Алгоритм match-&-window (−10 / +30 минут).

Кандидат «сильный», если есть ≥2 пласта подтверждения: соц + ончейн или соц + киты.

5. Фичи и скоринг

Формула:  
Score = 0.30SV + 0.20IW + 0.15VMS + 0.15CL + 0.15OC + 0.05WT – Penalties

SV: Social Velocity (скорость сообщений, z-score).

IW: Influencer Weight (Tier A/B, фолловеры, прошлые кейсы).

VMS: Visual Meme Score (CLIP/ViT, мемные картинки).

CL: Crypto Linkage (слова token/coin/contract/$TICKER).

OC: On-chain Confirmation (liq ≥10 SOL, holders ≥20, ≥60% не от деплойера).

WT: Whale Trace (входы китов).

Penalties: Rug/NSFW/плагиат тикера/мусор.

6. Анти-RUG проверки

Подключить GoPlus + RugCheck + Chainbase.

В отчёте: флаг риска (honeypot, freeze, mint-authority).

7. Память и самообучение

Хранилище знаний (ChromaDB):

/strategies/ (книги, стратегии, PDF, TXT)

/formulas/ (таблицы, мат. шаблоны)

/patterns/ (графики + описания)

/logs/ (успешные кейсы в .docx/.md)

DataLoader: авто-скан папки, конвертация pdf/docx/img → векторная память.

Автообновление: при загрузке новых файлов, без перезапуска.

Auto-logger: ИИ сохраняет успешные действия (входные данные, вывод, график, формулы) в /logs/.

8. Вывод (output)

JSON-алерт (score, social, onchain, whales, risk, решение).

DOCX/MD отчёт:

Название токена, график, анализ, паттерны, формулы, уровни, рекомендации.

Статус: BUY / HOLD / RISK.

9. Веб-панель

Редактор Tier A/B аккаунтов.

Настройки порогов (Score, SV σ, OC).

Управление словарями.

Режим risk-off (макро-фразы «war, strike, sanctions»).

Логи алертов + статус интеграций.

10. Архитектура

Backend: Python + FastAPI, асинхронные воркеры.

DB: Postgres + Redis (кэш) + S3 (медиа).

NLP: мультиязычный NER/ранкер (en/es/ru).

CV: CLIP/ViT для VMS.

Deploy: Docker, .env для ключей.

11. Приёмка (DoD)

Поток DataPulse + Helius/Birdeye без падений 24ч.

Событие «инфлюенсер + фото собаки» → mint через 3 мин → алерт Score ≥0.78.

Risk-off режим отключает BUY.

Веб-панель меняет пороги без рестарта.

Логи содержат фичи и причины.

📊 Отчёт по статусу Голиба

1. Что уже сделал ✅

Ingest (оффлайн JSON intake, DataPulse интеграция заготовлена).

Виральная лента + Z-скор (likes, reposts, engagement).

Аналитика токена (.md/.docx + график, RSI/MACD).

OCR+Chroma (база знаний, автоиндексация pdf/docx/img).

Merge Insights (агрегация, JSON-алерты).

2. Что в работе 🔄

Анти-RUG интеграции (GoPlus, RugCheck, Blowfish).

«Китовые» сигналы (PnL, winrate, автообновление из Chainbase).

OpenRouter (мульти-модель, policy-layer).

Бэктест 30 дней (метрики Precision/Recall).

Устойчивость (таймауты, retry).

3. Что ещё не знает (надо добавить в ТЗ) ⚠️

Многоступенчатая фильтрация соцсигналов (5 мин → 1 мин → хайп Discord/TG).

Хранилище знаний /knowledge\_base/ с авто-дообучением (pdf, стратегии, графики).

Auto-logger успешных кейсов в /logs/ (.md, .docx).

Экспорт структурированных отчётов в DOCX/MD.

Режим risk-off (макро-фразы).

Веб-панель управления (Tier A/B, пороги, словари).

………………………………………………………………………….

Что должна делать ИИ (Phase-1, только Twitter)

Шаг 1. Каждые 5 минут — «урожай мем-семян» из твитов знаменитостей

ИЗВЛЕКАЕМ (из текста, ALT-текста, OCR с картинки):

Имена/клички: людей, питомцев, персонажей (Fluffy, Mr. Nibbles, Lady Peanut).

Животных/виды: dog, cat, squirrel, frog, raccoon, duck, shark…

Предметы/еда/бренды: banana phone, toaster, croissant, donut, Tesla coil.

Звуки/ономатопея: bonk, mew, woof, ribbit, quack, boop, bing.

Смешные словечки/каламбуры/рифмы: giga-chad, based, sigma squirrel, turbo raccoon.

Короткие лозунги/фразы: “don’t bonk me”, “touch grass”, “this is fine”.

Хештеги, написания ALL-CAPS, необычные Эмодзи-кластеры (🐿️🐸🐱🚀🔥).

Текст на картинке (OCR): подписи, мем-панчи (например, «ACORN KING» на картинке белки).

ФИЛЬТРУЕМ:

убираем общеобщие слова (today, great, love), ссылки, упоминания людей вне контекста мемов.

оставляем короткие, звучные, легко-токенизируемые кусочки (2–12 символов для будущего тикера).

СЧИТАЕМ СКОРА:

Мемность: коротко, смешно, легко произнести, можно написать в ВЕРХНЕМ РЕГИСТРЕ (Memeability 0–1).

Животное/звуки: + бонус (Animal/Onomatopoeia +0.3).

Новизна: не встречалось у этого автора 30 дней (Novelty 0–1).

Ценность автора: вес знаменитости (Celebrity weight 0–1).

Вирусные признаки за первые 15–30 мин: лайки/мин, ретвиты/мин, уникальные реплаи (Engagement z-score).

→ Если Score ≥ порога или ≥3 разных автора подхватили одно и то же слово/картинку — эскалация на шаг 2.

Шаг 2. Ускорение до 1 минут

Следим только за топ-кандидатами из шага 1:

Появляется повтор этого слова в ответах, мем-репостах, фан-арт у других акков?

Одно и то же слово/имя появляется в разных формулировках/картинках?

Появились самопридуманные тикеры в реплаях ($FLUFFY, $SQUIRREL, $ACORN)?

Идёт мгновенная «игра слов» (рифмы/сокращения: SQUIRREL → $SQRL, $SCRL)?

→ Если да — помечаем HYPE CANDIDATE и готовим «блок токенизации».

Шаг 3. Финальный отбор (в рамках Phase-1 всё ещё только X)

ИИ генерит кандидат-название и тикер-варианты (2–5 символов, без плохих совпадений).

Проверяет схожесть с существующими известными мемкоинами (чтобы не клон до степени путаницы).

Если упоминания растут, ≥1 инфлюенсер вне исходного автора подхватил — вердикт: “Memecoin Seed = YES” (иначе HOLD/NO).

Уже на Phase-1 можно выводить тебе:  
«Seed: SQUIRREL 👑 (from @elonmusk) — memeability 0.82, variants: $SQRL / $NUT / $ACRN; echo-tweets: 5 authors/20m; recommend: WATCH (1-min)».

Исторические паттерны (что чаще всего «рождает» мемкоин)

Животные/питомцы: dog/doge, cat/mew, frog/pepe, raccoon, squirrel, owl, duck, shark, hamster.

Звуки: bonk, mew, woof, quack, ribbit, honk, beep.

Смешные имена: Cheems, Kabosu, Popcat, Slerf (бессмыслица, но звучно).

Короткие предметы/еда: banana, donut, burger, pizza, peanut, acorn.

Каламбуры/рифмы: SLERF, BONK, MEW, WIF (with), POG, ZOG — короткие, ударные.

Капс-слова/мем-лозунги: BASED, GIGA, CHAD, WOJAK — часто дают префикс/суффикс для тикера.

Конкретные правила/триггеры (для загрузки Голибом)

1) Извлечение кандидатов (текст + OCR + ALT)

Берём Noun Phrases / Proper Nouns / Animal species / Onomatopoeia / All-caps tokens.

Отбрасываем общие и стоп-слова.

Нормализуем: SQUIRREL → squirrel, Mr. Nibbles → MRNIBBLES, banana phone → bananaphone.

2) Счётчики (15–30 минут окна)

unique\_authors ≥ 3 по одной лексеме → +1 к мем-скор.

engagement z-score ≥ +1.5 к посту с этой лексемой.

echo-images: если на разных картинках обнаружен один объект (модель vision: squirrel/dog/cat/frog) → +0.3.

3) Канонизация в тикер

Коротко: 2–5 символов, легко произнести, без токсичных совпадений.

Генерация вариантов:

Сокращение: SQUIRREL → $SQRL

Семантика: ACORN → $ACRN, NUT → $NUT

Звук: BONK → $BONK, MEW → $MEW

Рифма/аллитерация: FLUFFY → $FLUF / $FLFY

4) Пороговые значения (по умолчанию)

Эскалация 1-мин: unique\_authors\_20m ≥ 3 или author ∈ trusted\_celebs и memeability ≥ 0.6.

Seed = YES: memeability ≥ 0.75 и (echo\_mentions\_growth ≥ x2 / 30m или first\_ticker\_variant\_seen == true).

HOLD: 0.6–0.75 (ждём подтверждения).

NO: <0.6 или «одноразовая шутка без подхвата».

Что передать Голибу прямо сейчас (готовые списки)

Категории извлечения (в коде как словари/классификаторы):

ANIMALS = [dog, cat, squirrel, raccoon, frog, duck, owl, shark, hamster, goat, pig, cow, bear, bee]

OBJECTS\_FOOD = [banana, donut, pizza, burger, toaster, croissant, peanut, acorn, cookie, icecream]

ONOMATOPOEIA = [bonk, mew, woof, quack, ribbit, honk, beep, boop, zing, bop]

MEME\_WORDS = [based, giga, chad, sigma, wojak, pepe, doom, cringe, zoomer]

NAME\_HINTS = [mr, mrs, sir, lord, lady, lil, big, baby] ← помогает ловить клички

Визуальные классы (vision-модель на изображениях/превью):

IMG\_CLASSES = [dog, cat, squirrel, raccoon, frog, duck, owl, shark, hamster, banana, donut, pizza, acorn]

Стоп-листы (чтобы не засорять):

Общие слова: today, morning, great, happy, lol, haha, wow, amazing, love, check, link, click, subscribe.

Политика/новости без мем-сигнала: vote, policy, senate, meeting (можно ослабить, если это часть мема).

Порог-конфиг (yaml/ini) — как выше, чтобы сразу включить.

Как выглядит итог для тебя (пример)

[22:14 CEST] Seed candidate: "SQUIRREL" 🐿️

Source: @elonmusk (memeability=0.82, unique\_authors\_20m=5, z=+2.1)

Image: squirrel detected (vision), ALT mentions: acorn

Ticker ideas: $SQRL / $ACRN / $NUT

Verdict: WATCH (1-min), waiting for echo-tickers or wider pickup

Именно это нам нужно: ИИ не ждёт «крипто-слов», она вытаскивает мем-семена у знаменитостей и оценивает шанс рождения токена. Если ок — дам Голибу это в виде трёх файлов (словарь, визуальные классы, конфиг порогов) готовых к загрузке.

Вы сказали:

теперь, что ты написал сверху 1. Шаг 1. Расширь в два раза , а именно допиши туда в двое слов , значений и т.д больше. 2. Важно чтобы ИИ парсила каждые 5 минут, а когда нашла из мусора уже слова которые подхватывают, то сократила и пристально отслеживала такие слова, картинки, выражения. 3. Важно, чтобы Голиб сделал, чтобы в последующим можно было доучивать ИИ \* загружать тексты, видео, графики, чтобы она на этом училась и сразу все подгружала .

ChatGPT сказал:

Думал на протяжении 44s

Брат 🙌 держи расширенный пакет именно для Шага 1 (Twitter-только) — я удвоил объём «мем-семян» (слов/образов/шаблонов), добавил чёткие правила частоты (5 мин → 1 мин) и требования к дообучению у Голиба.

Шаг 1. «Урожай мем-семян» (расширено ×2)

Если хочешь, я соберу всё это в 3 файла для Голиба (готовые к загрузке):

seed\_whitelist.txt (A–F, I)

vision\_anchors.txt (H: шаблоны/классы изображений)

config.yaml (K + пороги, окна, частоты 5 мин/1 мин)

да , давай. только точно, проверь.

Готово, брат 🙌 Я сделал три файла для Голиба — проверил, всё по-английски и заточено под Twitter-только.

Скачать seed\_whitelist.txt — расширенный словарь «мем-семян» (имена/животные/объекты/ономатопея/мем-слова/лозунги/эмодзи).

Скачать vision\_anchors.txt — якоря для картинок/ALT/OCR (классы объектов, мем-темплейты, крупные надписи).

Скачать config.yaml — частоты (5 мин → 1 мин), окна, пороги и правила эскалации + включён режим дообучения (runtime updates, upload/teach/export/import).

Если хочешь, добавлю в config.yaml твой точный список 200 аккаунтов (trusted\_list) — это повысит вес твитов от них и ускорит переход в 1-мин режим.

финальный комплект для Голиба (всё проверено и собрано), чтобы он вшил в ИИ и сдал Phase 1.2–1.3 без дыр. Копируй как есть.

0) Что именно встроить (итог по Phase 1.2–1.3)

Twitter Search + Timeline (приоритет: Tier A → Search → Tier B) с бэкфиллом.

Антибот/фильтры + regex из п.1.

Конструктор Search-запросов (п.2) + минус-слова.

Tier A/B CSV, хот-апдейт, CELEB\_OVERRIDE (п.3).

Политика деградации/ретраев/алертов (п.4).

Контракт /ingest/social\_alert → Алекс (п.5).

Метрики/логгинг (п.6).

LLM-цепочка (Aegis→DeepSeek→MythoMax) (п.7).

Chroma KB: запись, метаданные, снапшоты (п.8).

Docker + Compose + .env + config.json (п.9).

Набор curl-тестов (п.10).

1) Regex-пакет (PCRE/Python OK)

Позитивные триггеры

(?i)\b(just\s+launched|new\s+(token|coin)|created\s+(on|at)\s+(raydium|jupiter)|new\s+pool|pool\s+created|listing\s+(soon|today)|live\s+now|go\s+live)\b

(?i)\b(pump(ing)?|moon(ing)?|spike|break(out)?|parabolic|send\s+it|ath|0\sto\s+100)\b

(?i)\b(raydium|jupiter|orca|pump\.fun|amm|dex|lp|liquidity|mint(ed)?|jito\s+mev|sniper|bot)\b

(?i)\b(og\s+(token|coin)|original\s+(token|coin)|first\s+(token|coin)|the\s+only\s+official)\b

Исключения (шум)

(?i)\b(airdrop|pre[-\s]?sale|whitelist|wl\s+spot|give\s?away|free\s+mint|bounty|invite\s+link|casino|forex|copytrade|signals?\s+group)\b

Solana mint/base58 (32–44)

\b[1-9A-HJ-NP-Za-km-z]{32,44}\b

DEX-баннер/реклама

(?i)\b(promoted\s+by|sponsored|ad|paid\s+banner|promote)\b.\*\b(dex|raydium|jupiter|pump\.fun)\b

Смена имени (эвристика)

(?i)\b(previous(ly)?\s+known\s+as|aka|renamed\s+from)\b

Логика: name\_change\_count ≤ 2, иначе → spam\_flag=true.

2) Twitter Search — готовые шаблоны

Поддержать синтаксис: OR, AND, (), кавычки "...", #tag, @handle, from:, url:, has:links, -слово, lang:  
Лимит строки: ≤ 480 символов  
Минус-слова (единый список):  
airdrop,presale,whitelist,wl,giveaway,free mint,bounty,casino,forex,signals group,copytrade

2.1 Broad (recall):

(pepe OR doge OR "cat ceo" OR "elon meme" OR "trump coin" OR #memecoin)

(pump OR moon OR "new pool" OR "just launched" OR "created on raydium" OR "go live" OR pump.fun)

(#solana OR raydium OR jupiter OR orca)

-airdrop -presale -whitelist -giveaway -bounty -copytrade -"signals group"

lang:en

2.2 Tight (precision):

("just launched" OR "new pool" OR "created on raydium" OR "listing today" OR "live now")

(#solana OR raydium OR jupiter)

(pepe OR doge OR cat OR frog)

(from:tierA\_trader OR from:cryptoOG OR @elonmusk)

-airdrop -presale -whitelist -giveaway

2.3 Locales (по необходимости):

("только что запустили" OR "новый пул" OR "листинг сегодня") (#солана OR raydium OR jupiter) -airdrop -раздача lang:ru

("listado hoy" OR "nuevo pool" OR "acaba de lanzar") (#solana OR raydium OR jupiter) -sorteo -airdrop lang:es

3) Tier A/B — CSV + кэш

Файлы: data/tierA.csv, data/tierB.csv  
Заголовок:

user\_id,handle,tier,weight,celeb,notes,last\_seen

Пример:

44196397,@elonmusk,A,1.0,1,celeb override,2025-09-05T10:00:00Z

12345678,@tierA\_trader,A,0.9,0,alpha caller,2025-09-04T21:15:00Z

87654321,@memelord99,B,0.7,0,meme curator,2025-09-04T21:15:00Z

user\_id резолвим при загрузке → кэш data/tw\_id\_cache.json.

Хот-апдейт: каждые 6 часов + ручной /admin/reload\_tiers.

CELEB\_OVERRIDE=true для Tier A с celeb=1.

4) Деградация / ошибки / алерты

Backoff: base 2s, max 120s, retries 5.

10 минут даунтайма: деградированный режим — Search OFF, Timeline Tier A с интервалом ×4; складываем задания в очередь и реплеим.

Допустимая потеря цикла: ≤ 5%.

Алерты: Telegram ops\_social\_alerts + email (JSON-краткий), уровни WARN/ERROR, rate ≤ 1/мин.

5) Контракт к Алексу /ingest/social\_alert

Stage: https://alex.stage/api/ingest/social\_alert

Prod: https://alex.prod/api/ingest/social\_alert

Auth: Authorization: Bearer INTERNAL\_TOKEN

Идемпотентность: alert\_id в теле + заголовок X-Idempotency-Key: <alert\_id>

Лимиты: body ≤ 256 KB, ≤ 5 req/sec, SLA ≤ 1.5s

Required:  
alert\_id, created\_at, profile(twitter|pump|find|listings), focus(keyword|token|event), confidence(0..1), zscore, sources[]({url,handle,tier,ts}), summary, decision\_hint(BUY|WATCH|NO)

Optional:  
token\_guess{name,mint?}, og, clones, dex\_banner, name\_changes, sentiment(-1..1), media[], trace\_id, debug

Пример тела:

"alert\_id": "sa\_2025-09-06\_12-31-07\_001",

"created\_at": "2025-09-06T12:31:07Z",

"profile": "twitter",

"focus": "keyword",

"confidence": 0.82,

"zscore": 3.1,

"summary": "Tier A упомянул DOGE2025; всплеск упоминаний; TG подтверждает.",

"sources": [

{"url":"https://twitter.com/...","handle":"@tierA\_trader","tier":"A","ts":"2025-09-06T12:29:40Z"}

],

"token\_guess": {"name":"DOGE2025"},

"og": false,

"clones": 3,

"dex\_banner": true,

"name\_changes": 1,

"sentiment": 0.44,

"decision\_hint": "WATCH",

"trace\_id": "trc\_f92a"

6) Метрики / логи

GET /api/golib/health

Счётчики: pulled, accepted, alerts\_sent, forwarded\_ok, forwarded\_fail

Тайминги: fetch\_ms, filter\_ms, ai\_ms, forward\_ms

GET /metrics (Prometheus)

Логи: JSONL, ключи: ts, trace\_id, alert\_id, request\_id, source\_kind(search|timeline), tier, stage, \*\_ms, status

Пример лог-строки:

{"ts":"2025-09-06T12:31:08Z","trace\_id":"trc\_f92a","stage":"forward","alert\_id":"sa\_2025-09-06\_12-31-07\_001","forward\_ms":143,"status":"accepted"}

7) LLM-цепочка

Порядок: Aegis → DeepSeek → MythoMax (авто-fallback).

Таймауты: 6s / 8s / 10s.

Пороги: has\_token=0.70, will\_token=0.60.

Параметры: max\_tokens=1024, temperature=0.2, max\_calls\_per\_event=3, daily\_budget\_usd=3.

8) Chroma KB

Запись: все события c флагом status: accepted|rejected.

Метаданные (vector metadatas):  
id,hash,author,created\_at,tickers[],zscore,memeability,decision\_hint,confidence,tier,og,dex\_banner,name\_changes,source,urls[]

Ретенция: 90 дней или ≤ 5M векторов (TTL + LRU).

Dev-поиск (фильтры): created\_at range, tier∈{A,B}, zscore≥, memeability≥, tickers[], og, dex\_banner.

Экспорт снапшота: ежедневно 00:00 UTC → data/kb\_snapshots/YYYY-MM-DD/ (Parquet + JSONL).

9) Конфиг / .env / Docker

.env

PORT=8080

INTERNAL\_TOKEN=replace\_me

DATAIMPULSE\_TOKEN=replace\_me

GROQ\_API\_KEY=replace\_me

CHROMA\_URL=http://chroma:8000

CHROMA\_COLLECTION=social\_kb

ALEX\_INGEST\_URL=https://alex.stage/api/ingest/social\_alert

TWITTER\_MODE=both

SEARCH\_BACKFILL\_HOURS=24

TIMELINE\_BACKFILL\_HOURS=48

TIMELINE\_MAX\_TWEETS=200

LANG\_FILTER=none

RETRY\_BASE\_SEC=2

RETRY\_MAX\_SEC=120

RETRY\_COUNT=5

DEGRADED\_AFTER\_MIN=10

LOSS\_TOLERANCE\_PCT=5

config.json

"tiers": {

"files": ["data/tierA.csv", "data/tierB.csv"],

"hot\_reload\_hours": 6,

"celeb\_override": true

},

"search": {

"max\_query\_len": 480,

"minus\_words": ["airdrop","presale","whitelist","wl","giveaway","free mint","bounty","casino","forex","signals group","copytrade"]

"llm": {

"order": ["aegis","deepseek","mythomax"],

"timeouts\_sec": {"aegis": 6, "deepseek": 8, "mythomax": 10},

"thresholds": {"has\_token": 0.7, "will\_token": 0.6},

"max\_tokens": 1024,

"temperature": 0.2,

"max\_calls\_per\_event": 3,

"daily\_budget\_usd": 3

"kb": {

"write\_policy": "all\_with\_flag",

"retention\_days": 90,

"max\_vectors": 5000000,

"metadata\_fields": ["id","hash","author","created\_at","tickers","zscore","memeability","decision\_hint","confidence","tier","og","dex\_banner","name\_changes","source","urls"]

"forward": {

"alex\_url\_stage": "https://alex.stage/api/ingest/social\_alert",

"alex\_url\_prod": "https://alex.prod/api/ingest/social\_alert",

"auth\_header": "Authorization",

"token": "INTERNAL\_TOKEN",

"idempotency\_header": "X-Idempotency-Key",

"rate\_limit\_rps": 5,

"max\_body\_kb": 256,

"sla\_sec": 1.5

requirements.txt

fastapi==0.112.0

uvicorn[standard]==0.30.0

httpx==0.27.0

pydantic==2.8.2

regex==2024.5.15

orjson==3.10.7

pandas==2.2.2

numpy==1.26.4

chromadb==0.5.0

tenacity==8.3.0

prometheus-client==0.20.0

Dockerfile

FROM python:3.11-slim

WORKDIR /app

COPY requirements.txt .

RUN pip install -r requirements.txt

COPY . .

ENV PYTHONUNBUFFERED=1

EXPOSE 8080

CMD ["python","-m","golib\_api.server"]

docker-compose.yml

version: "3.9"

services:

golib:

build: .

env\_file: .env

ports: ["8080:8080"]

volumes:

- ./data:/app/data

depends\_on:

- chroma

chroma:

image: chromadb/chroma:latest

ports: ["8000:8000"]

10) Быстрые тесты (обязательно приложить)

Health

curl -s localhost:8080/api/golib/health

Reload tiers

curl -s -X POST -H "Authorization: Bearer INTERNAL\_TOKEN" localhost:8080/admin/reload\_tiers

Social check (demo)

curl -s -X POST -H "Authorization: Bearer INTERNAL\_TOKEN" -H "Content-Type: application/json" \

-d '{ "request\_id":"run\_1","profile":"pump","token":{"name":"DOGE2025","mint":"D0gexxxx"}, "options":{"check\_twitter":true,"check\_telegram":true,"window\_min":30}}' \

localhost:8080/api/golib/social\_check

Forward to Alex (stage)

curl -s -X POST -H "Authorization: Bearer INTERNAL\_TOKEN" -H "X-Idempotency-Key: sa\_test\_001" \

-d '{ "alert\_id":"sa\_test\_001","created\_at":"2025-09-06T12:31:07Z","profile":"twitter","focus":"keyword","confidence":0.82,"zscore":3.1,"summary":"test","sources":[{"url":"https://twitter.com/...","handle":"@tierA\_trader","tier":"A","ts":"2025-09-06T12:29:40Z"}],"decision\_hint":"WATCH"}' \

https://alex.stage/api/ingest/social\_alert

Финальный контроль (что принять у Голиба)

Архив/репозиторий с кодом + Docker + compose + .env.example + config.json.

Папка data/ (Tier A/B CSV + tw\_id\_cache.json).

Док README.md с запуском и списком эндпоинтов.

Скрины/видео E2E: DI batch → фильтр/LLM → social\_alert → приём у Алекса (202/OK).

metrics и health доступны.

Демка social\_check по токену от Алекса (ручной вызов) работает.

Если он всё это вшивает — Алекс просто подставит токены/URL и подключит к интерфейсу.

ответ по Phase 1.2–1.3 — строго по пунктам:

Режимы выборки Twitter (стратегия)  
☑️ Search по ключам/регекспам (основа)  
☑️ Accounts timeline по Tier-спискам (основа)  
☑️ Оба: приоритет Timeline Tier A → Search → Timeline Tier B  
Бэкфилл: Search — последние 24 часа; Timeline — до 200 твитов или 48 часов на аккаунт (что раньше).

Конструкция запросов для Search (точно как запускать)  
Поддерживаемый синтаксис (OR/AND/(), хэштеги, упоминания, язык): поддержать OR, AND, (), кавычки для точной фразы, #tag, @handle, from:, url:, has:links, -слово (исключение), lang:.  
Лимит длины запроса (символов): ≤ 480.  
Фильтр языка (lang): ☑️ нет (многоязычный поток; при необходимости можем сужать до en, es, ru).  
Исключающие слова (минус-слова): airdrop, presale, whitelist, giveaway, “free mint”, “send seed”, bounty.

Дай 2 эталонных запроса для продакшена:  
2.1 Broad (высокий recall):

(pump OR moon OR "about to launch" OR "new token")

-airdrop -presale -whitelist -giveaway -bounty lang:en

2.2 Tight (высокий precision):

("just launched" OR "new pool" OR "created on raydium" OR "jito mev")

(pepe OR doge OR cat)

-airdrop -presale -whitelist

Tier A/B списки (использование и обновление)  
Источник/файл: /data/tierA.csv, /data/tierB.csv (CSV: user\_id,handle,tier,weight).  
Идентификатор: ☑️ оба (@handle и user\_id). Где брать id: одноразовая резолвация через API при загрузке списков; кэш в /data/tw\_id\_cache.json.  
Обновление списков: ☑️ хот-апдейт раз в 6 часов (и вручную через /admin/reload\_tiers).  
Использование: ☑️ также собирать их таймлайны (плюс повышать вес при скоринге).  
CELEB\_OVERRIDE (обход части фильтров для Tier A): ☑️ нужен (всегда показывать, но помечать «опасно», если security-сигналы плохие).

Политика деградации и ошибок (прокси/скрейп)  
Бэкофф: base=2с, max=120с, retries=5.  
При длительной недоступности (> 10 мин): ☑️ деградированный режим — отключаем Search, оставляем Timeline Tier A с интервалом ×4; кэшируем запросы в очередь, делаем реплей при восстановлении.  
Допустимая потеря данных за цикл: ≤ 5%.  
Нужны ли алерты (канал/формат): Telegram ops\_social\_alerts + email (JSON краткая сводка), уровень WARN/ERROR, rate-limit 1/мин.

Контракт Alex /ingest/social\_alert (строгое ТЗ)  
URL (prod/stage): prod https://alex.prod/api/ingest/social\_alert, stage https://alex.stage/api/ingest/social\_alert.  
Auth: Authorization: Bearer \*\*INTERNAL\_TOKEN\*\*.  
Требуемые поля vs optional (список):

required: alert\_id, created\_at, profile (twitter|pump|find|listings), focus (keyword|token|event), confidence (0–1), zscore, sources[] (url, handle, tier, ts), summary, decision\_hint (BUY|WATCH|NO).

optional: token\_guess (name, mint?), og (true/false), clones (int), dex\_banner (true/false), name\_changes (int), sentiment (-1..1), media[], trace\_id, debug.  
Идемпотентность: ☑️ alert\_id в теле + ☑️ заголовок X-Idempotency-Key (тот же alert\_id).  
Лимиты: макс. размер 256 KB, скорость ≤ 5 req/sec, SLA ответа ≤ 1.5 с.  
Коды/ошибки:

202 Accepted { "ok": true, "id": "<alert\_id>" }

400 Bad Request { "ok": false, "error": "validation\_failed", "field": "..." }

401 Unauthorized { "ok": false, "error": "unauthorized" }

409 Conflict { "ok": false, "error": "duplicate" }

429 Rate Limit { "ok": false, "error": "rate\_limited", "retry\_after": 30 }

5xx Server Error { "ok": false, "error": "server\_error", "trace\_id": "..." }

Наблюдаемость/метрики (что обязательно видеть наружу)  
☑️ /api/golib/health  
☑️ Счётчики: pulled, accepted, alerts\_sent, forwarded\_ok, forwarded\_fail  
☑️ Тайминги этапов: fetch\_ms, filter\_ms, ai\_ms, forward\_ms  
☑️ Prometheus /metrics  
Логи: ☑️ jsonl; кореляционные id/ключи: trace\_id, alert\_id, request\_id, source\_kind (search|timeline), tier.

LLM-цепочка (Aegis / DeepSeek / MythoMax)  
Порядок и fallback: Aegis → DeepSeek → MythoMax (авто-переключение по таймауту/ошибке).  
Таймауты: Aegis 6с / DeepSeek 8с / MythoMax 10с.  
Пороги has\_token / will\_token: 0.70 / 0.60.  
Макс. токены / температура / лимиты стоимости: до 1024 токенов/запрос, temp=0.2; лимит цепочки ≤ 3 вызова/событие; дневной бюджет ≤ $3 экв.

Интеграция KB (Chroma) в соц-путь  
Момент записи: ☑️ все (с флагом) — status: accepted|rejected, чтобы учиться и на негативе.  
Метаданные в векторе: id/hash, author, created\_at, tickers[], zscore, memeability, decision\_hint, confidence, tier, og, dex\_banner, name\_changes, source, urls[] (добавить tier, og, name\_changes).  
Ретенция/объём: 90 дней или до 5 млн векторов; eviction по времени (TTL) + LRU.  
Dev-поиск: какие поля фильтровать/показывать в выдаче: created\_at (диапазон), tier ∈ {A,B}, zscore≥, memeability≥, tickers[], og=true/false, dex\_banner=true/false; в выдаче: фраза/сниппет, топ-аккаунты, ссылки, оценки, OG/клоны.  
Экспорт снапшота: ☑️ нужен — ежедневно 00:00 UTC, формат Parquet + JSONL, выгрузка в /data/kb\_snapshots/YYYY-MM-DD/.

Приложения (короткие примеры):

Пример тела social\_alert:

"summary": "Tier A аккаунт упомянул DOGE2025, быстрый рост упоминаний, TG подтверждает.",

"trace\_id": "trc\_f92a",

"debug": false

Пример лог-строки jsonl:

{"ts":"2025-09-06T12:31:08Z","trace\_id":"trc\_f92a","stage":"forward","alert\_id":"sa\_2025-09-06\_12-3

Это список того, что ИИ должна ловить: слова, выражения, шаблоны текста, эмодзи и типовые картинки. Сделано так, чтобы не упустить ни один триггер мем-пампа.

1) СЕМЕНА СЮЖЕТА (что ловим каждые 5 минут только в Твиттере)

A. Лексика старта/создания монеты

launching now, going live, fair launch, no presale, no insiders, mint live, deploying, contract soon, CA soon, contract dropped, just deployed, bonding curve, v2 soon, migration, renounced, LP burned/locked, 0% tax, buy/sell tax 0/0, CMC soon, CG soon, listing soon, listing today, raydium pool created, meteora pool, orca pool, pump.fun trending.

B. Солана/площадки/тех.термины (прямые маркеры)

pump.fun, Raydium, Meteora, Orca, Jupiter, Birdeye, DexScreener, Helius, sniper, anti-MEV, Jito, bonding curve, pump portal.

Шаблоны ссылок/текста: pump.fun/coin/...pump, dexscreener.com/solana/, birdeye.so/token/, raydium.io/liquidity.

C. Символика мем-пампа (сильные вирус-триггеры)

flip top 5/top 10, send it, send, we go parabolic, to the moon, 100x/1000x, next gem, chosen one, new meta, degen season, npc/pepe/wojak, cat/doge/shiba/trump/tate/kanye/elon.

D. Контракт/тикер (сильный фильтр достоверности)

Наличие тикера с $: $TOKEN, $HORNY, $SONIC и т.п.

Маркеры контракта: CA:, Contract:, Mint:, вставка длинной base58-строки (без «…pump» на конце как псевдо-адрес).

Фразы: official CA, real CA, fake CA alert, don’t buy fake.

E. Ранние рыночные признаки

whale buy, $10k buy, $50k buy, fresh wallet, new wallet ape, top holder, smart money in, alpha call.

Рус: заход кита, крупная покупка, умные деньги зашли. улетим, ракета, сотка иксов, пушка, пошла жара, мема-сезон, деген-сезон.

F. Эмодзи/символы, которые часто сопровождают старт

🚀 🌕 📈 💎 🧻 🐋 🐸 🐱 🐶 🤖 👑 🔥 ⚡️ 🧠 🎯

2) ЭТАП УСКОРЕНИЯ (переключаемся на 1-минутные проверки)

Что должно «щёлкнуть», чтобы ИИ ускорилась:

За 15–30 мин по теме: ≥ 3 уникальных твита ИЛИ 1 твит от надёжного аккаунта из твоего списка.

Упоминания растут x2+/15–30 мин; появляются повторы слов/мем-картинок из A–C.

Появились тикер $... или явные намёки на CA (CA soon, contract dropped).

Пошли слова из E (киты/суммы).

3) ТРЕТЬЯ ПРОВЕРКА (подключаем Discord/TG, делаем финальный отбор слов/шуток)

Что дополнительно отслеживаем (повышаем точность):

G. Соц-вспышки/обсуждения

spaces now, voice chat live, community raid, pin & raid, meme contest, poster template, banner pack, new website live.

Рус: спейс сейчас, рейд комьюнити, закреп пост, мем-конкурс, сайт залили. Рус/сленг: стартуем, запуск сейчас, фэйр запуск, без пресейла, адрес контракта скоро, контракт выложили, ликва сожжена/залочена, налог 0/0, листинг скоро/сегодня.

H. Образы/картинки (на что ИИ должна обращать внимание в медиа)

Скриншоты pump.fun с карточкой монеты, скрин DexScreener с зелёными свечами/трендами, коллажи с контрактом (CA/Mint) прямо на картинке.

Мем-темплейты: Pepe/Wojak/Chad, ракета/луна, экраны терминала, счётчики «x100/x1000», коты/собаки/полит-лица (Trump/Tate/Elon/Kanye) в связке с $TICKER.

Изображения «LP burned/locked», «0/0 tax», «renounced» как наклейки/бейджи.

I. Прямые сигналы «массового интереса»

В чате: pin message, everyone join VC, raid this tweet, post your buys, leaderboard, top 1 trending, we’re trending DexScreener, CMC/CG application sent.

4) ЧТО СЧИТАЕМ «МУСОРОМ» (фильтруем на 1-м этапе)

airdrop, giveaway, whitelist, referral, presale, bounty, free mint, claim now — почти всегда шитпост/скам.

Посты без $TICKER и без явных слов из разделов A–E.

Любые «контракты» с подозрительными окончаниями (…pump как часть адреса) или короткими псевдо-строками.

Картинки без текста/контракта и без привязки к крипто-лексике (просто мем без упоминания монеты).

5) Быстрые правила эскалации (коротко)

Шаг 1 (5 мин): ловим A–F; метрики: ≥3 твита/15–30 мин или надёжный автор → ускоряемся.

Шаг 2 (1 мин): ждём появления $TICKER/CA/линков Dex/Pump ИЛИ кит-лексики → готовим финтриггер.

Шаг 3 (с Discord/TG): подтверждение через G–I (рейды, VC, пины, мем-пакеты, сайт) → метка HYPE.

6) Мини-шпаргалка «на чём сходу делаем ставку»

Есть $TICKER + pump.fun/coin + no presale/fair launch → внимание.

Появился CA/Mint + LP burned/locked, 0/0 tax → сильный кандидат.

Видим whale buy/$10k+, fresh wallet ape + DexScreener тренд → готовимся к входу.

В Discord/TG: raid/pin/VC now + рост упоминаний → HYPE подтверждён.

ТЗ для фрилансера

📋 Техническое задание

Проект для архитектурного ассистента

1. Введение и цель проекта

Мы поручаем разработку концептуального архитектурного проекта в рамках учебно-практического задания.  
Основная цель:

показать навыки работы с пространством, планировкой и визуализацией,

оформить результат в виде презентации, которая будет выглядеть профессионально и логично.

Важно: это не рабочая документация для строительства, а именно концептуальный проект — идея, которую можно показать клиенту, инвестору или в портфолио.

2. Тема и объект

Предлагается разработать концепцию небольшого жилого дома площадью 80–120 м².  
Дом должен включать:

гостиную,

кухню-столовую,

2 спальни,

санузел,

небольшую террасу или балкон.

Задача — продумать удобную и компактную планировку, которая могла бы реально использоваться в современном жилье.

Альтернативный вариант (если удобнее):

разработать концепцию павильона для парка (кафе, точка отдыха или выставочный павильон).

3. Задачи исполнителя

3.1. Эскизы и планы (2D)

Нарисовать 2–3 варианта планировочного решения.

Показать, как расположены помещения, мебель и основные элементы.

Важно: планы должны быть простыми, но аккуратными, с понятными подписями.

3.2. Визуализации (3D-эскизы)

Создать минимум 2–3 вида: фасад, интерьер, общий вид здания.

Можно использовать SketchUp, 3ds Max или любой инструмент, которым владеет исполнитель.

Главное — передать атмосферу и объём.

3.3. Презентация проекта (PDF, 6–8 страниц)

Структура:

Титульный лист (название проекта, автор).

Введение (коротко о концепции, 1–2 абзаца).

План-схема (один из вариантов 2D).  
4–6. Визуализации (фасад, интерьер, общий вид).

Краткая характеристика: площадь, материалы (примерно), назначение.

Заключение (одно предложение: «Проект отражает современный минимализм» или др.).

4. Требования к качеству

Аккуратность, читаемость схем.

Чистая и логичная подача в презентации (удобная последовательность, отсутствие перегрузки).

Минимум текста, акцент на визуальный материал.

Размер презентации до 30 МБ.

5. Формат сдачи

Эскизы — JPG/PNG (2D-планы).

Визуализации — JPG/PNG (3D).

Презентация — PDF (6–8 страниц).

6. Сроки и бюджет

Срок выполнения: 7–10 дней.

Бюджет: 50 €.

Сделал под Solana (и добавил опциональные regex-переключатели под EVM на будущее). Всё структурировано: keywords (слова/эмодзи) и regex (тикеры/минты/ссылки).

Для Голиба — входные фильтры соц-аналитики (Solana)

1) Watchlist keywords (слова и фразы для ловли постов)

Группы + RU/EN варианты. Регистр игнорировать. Эмодзи учитывать как отдельные триггеры.

A. Старт/листинг/пулы

mint, new mint, mint live, minting now

new token, token just launched, launching in, goes live

pool created, liquidity added, LP added, Raydium pool

fair launch, stealth launch, prelisting, pre listing, TGE, IDO

RU: минт, листинг, пул создан, добавили ликвидность, честный запуск, тихий запуск

B. Киты/«умные» деньги/владельцы

whale buy, whale in, smart money, insider, dev buy, team buy

top holder, fresh wallet, new wallet, accumulation, wallet flagged

RU: кит зашёл, умные деньги, покупка от кита, новый кошелек, накопление

C. Хайп/мем-триггеры и жаргон

degen, based, alpha, send it, cook, print, ape in

100x, x100, moon, mooning, go parabolic, ATH, breakout

viral, trending, raid, ct (crypto twitter), gmgn, ngmi, wagmi

Тематики/персоны: elon, trump, ai, dog, cat, pepe, frog

Эмодзи-триггеры: 🚀🔥📈💎🤲🐳🐸🐶🟢🧠🪙

RU: памп, полетели, вирусится, тренд, параболик, пробой, ракета

D. Риски/скам-маркеры (сигналы безопасности)

rug, rugpull, honeypot, trading disabled, blacklist, anti-whale

mint authority revoked/not revoked, freeze authority revoked/not revoked

LP unlocked, LP lock, renounced, upgradeable

tax, buy/sell tax, >10% tax

RU: раг, медовый, торговля выключена, чёрный список, анти-кит, налог, минт/фриз отключён/включён, ликвидность не залочена/залочена

E. Соц-механики/рефка/аирдропы

airdrop, giveaway, retweet to win, tag 3 friends, wl (whitelist)

RU: розыгрыш, раздача, отметь друзей, белый список

F. Инфраструктура/сервисы (Solana-контекст)

pump.fun, Raydium, Jupiter, Birdeye, Dexscreener, Solscan, Helius

Примечание: делаем белый список эмодзи и жаргона (выше) → это снижает шум и ловит хайповые посты до пампа.

2) Regex (строгие фильтры по тикерам/адресам/ссылкам)

Базовые (включить)

$TICKER (в верхнем регистре 2–10 символов)

(?<![A-Za-z0-9])\$(?:[A-Z]{2,10})(?![A-Za-z0-9])

Solana mint/public key/program id (base58, 32–44 символа)

Solana tx signature (base58 длинной подписи; держим диапазон шире)

\b[1-9A-HJ-NP-Za-km-z]{64,100}\b

Рекомендуется усиливать контекстом: слово рядом tx|signature|sig|txn|транзакц.

Pump.fun (вытягиваем mint, если указан)

https?://(?:www\.)?pump\.fun/(?:coin/)?([1-9A-HJ-NP-Za-km-z]{32,44})

Birdeye (Solana)

https?://birdeye\.so/token/([1-9A-HJ-NP-Za-km-z]{32,44})(?:\?chain=solana)?

Solscan (token)

https?://solscan\.io/token/([1-9A-HJ-NP-Za-km-z]{32,44})

Dexscreener (Solana pair/token)

https?://dexscreener\.com/solana/([1-9A-HJ-NP-Za-km-z]{32,44})

Jupiter swap (маршрут между mint-ами)

https?://jup\.ag/swap/([1-9A-HJ-NP-Za-km-z]{32,44})-([1-9A-HJ-NP-Za-km-z]{32,44})

Денежные метрики (USD) в тексте — полезно для извлечения ликвидности/объёма

(?:\$|USD\s?)\s?\d{1,3}(?:,\d{3})\*(?:\.\d+)?|\d+(?:\.\d+)?\s?(?:USD|\$)

Опциональные (выключены по умолчанию; на будущее под EVM)

EVM адрес (0x + 40 hex)

\b0x[a-fA-F0-9]{40}\b

Tx hash EVM (0x + 64 hex)

\b0x[a-fA-F0-9]{64}\b

Рекомендация реализации: держать эти regex в конфиге с флагом enabled: true|false, чтобы быстро включать/выключать сети.

3) Правила снижения шума (recommendations)

Игнорировать $USD, $FREE, $LOVE и другие частые не-тикеры через стоп-лист (ticker\_denylist).

Для Solana-адресов отсеивать явные мусорные по контексту: если рядом нет слов mint|token|pool|raydium|pump|birdeye|solscan, снижать вес сигнала.

Сообщения с ≥2 триггерами (напр. whale buy + $TICKER + liquidity added) — повышать приоритет.

Эмодзи 🚀🔥📈 + слова launch|listing|mint — усиливать вес (соц-памп паттерн).

4) Мини-скоринг (как агрегировать сигнал поста)

base\_score =  
0.35 \* social\_intent (хайп/эмодзи/слова) +  
0.40 \* onchain\_context (mint/links/addresses) +  
0.25 \* credibility (автор Tier A/B, метрики поста)

Увеличить на +10..+20%, если упомянуты Raydium/Jupiter/Birdeye/Dexscreener ссылками.

Понизить на −30%, если в том же посте встречаются rug/honeypot/blacklist и т.п.

5) Что нужно от Голиба сейчас

Поднять временный публичный endpoint для POST /internal/merge\_insights (Replit/Render/Vercel/Ngrok — любой).

Вернуть нам URL, чтобы я подставил его в:

Postman-коллекцию ({{GOLIB\_BASE\_URL}}/internal/merge\_insights),

OpenAPI (servers: [ { url: "<URL>" } ]),

.env (GOLIB\_BASE\_URL=<URL>).

Принять наш payload (у тебя уже есть пример merge\_insights\_example.json), прислать 200 OK.

Дополнение

Я добавлю отдельный файл ticker\_denylist.txt в конфиг, и Голиб сможет просто исключать эти совпадения после regex-парсинга.

📛 Стоп-лист тикеров (первый набор)

Часто встречающиеся «ложные» тикеры

$WIN

$HOLD

$LOVE

$FREE

$ALL

$NEW

$ONE

$NOW

$FUN

$GOOD

$YES

$NO

$UP

$MOON (как слово хайпа, не тикер)

$PUMP (аналогично)

Опционально — добавить русские

$ДА

$НЕТ

$ВСЕ

📂 Формат файла (ticker\_denylist.txt)

# tickers to ignore (false positives / spam)

$MOON

$PUMP

⚙️ Логика применения

Сначала regex находит все $[A-Z]{2,10}.

Потом система проверяет, нет ли тикера в ticker\_denylist.txt.

Если совпадает — этот тикер отбрасывается (не идёт в анализ).