

**Говорят, AI уже везде.
Раз такое дело, может быть,
что-то автоматизируем?**

| HolyJS 2026

Привет, я Илья

Full-stack разработчик

eva.ai – HR SaaS Platform



Военный

в прошлом



Сисадмин

в прошлом



Разработчик

сегодня



Плиточник

План Б 😊



Беларусь 



Мы

Наши боли

3.5 землекопа

Сервисов — много.

**Языков программирования
— много.**

**Тикеты копятя быстрее,
чем фиксиатся**

Сели думать, как жить дальше



Поднимать больше,
тащить дальше

Сели думать, как жить дальше



**Поднимать больше,
тащить дальше**

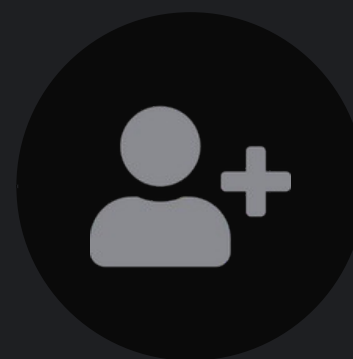


**Нанять новых
специалистов**

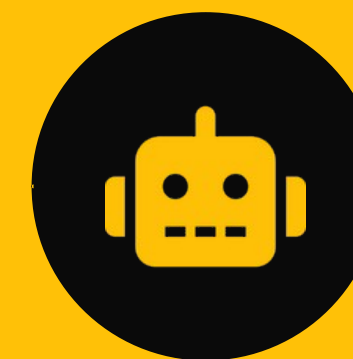
Сели думать, как жить дальше



**Поднимать больше,
тащить дальше**



**Нанять новых
специалистов**



**В любой непонятной
ситуации, добавляй AI**

Ритуал фикса

 Открыть Sentry — читаем полезную нагрузку




 Понять контекст — кто вызвал, почему упало



 Написать фикс



 Прогнать линтеры и чекеры



 Создать PR — описать что и зачем

~1 час
на каждый баг

5-7
тикетов в неделю

А что если создать bug-fix pipeline?



Повторяемость

Баги появляются снова и снова

А что если создать bug-fix pipeline?



Повторяемость

Баги появляются снова и снова



Чёткий I/O

*Вход — баг-репорт.
Ожидаемый выход — PR с фиксом.*

А что если создать bug-fix pipeline?



Повторяемость

Баги появляются снова и снова



Низкий риск

Главное не лить в пятницу, на прод.



Чёткий I/O

*Вход — баг-репорт.
Ожидаемый выход — PR с фиксом.*

А что если создать bug-fix pipeline?



Повторяемость

Баги появляются снова и снова



Чёткий I/O

*Вход — баг-репорт.
Ожидаемый выход — PR с фиксом.*



Низкий риск

Главное не лить в пятницу, на прод.

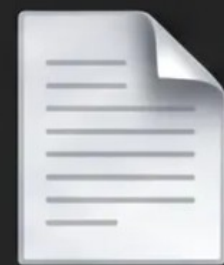


Стоимость

*Желательно не платить.
А еще лучше где-то сэкономить.*

Прежде чем писать pipeline...

нужно проработать два направления



Максимальный контекст

AI должен понимать не только данные из системы мониторинга, но и бизнес-логику.

Прежде чем писать pipeline...

нужно проработать два направления



Максимальный контекст

AI должен понимать не только данные из системы мониторинга, но и бизнес-логику.



Точки контроля

Lint и type-check — недостаточно. Нужен серьёзный верификатор.

Документация — это контекст для агента

Код молчит. Только документация знает.



Помогает решить

Агент находит нужную информацию и принимает решения опираясь на найденный контекст.

*if one Transition Action exception happens
→ other Transition Actions should be proceeded*

Документация — это контекст для агента

Код молчит. Только документация знает.



Помогает решить

Агент находит нужную информацию и принимает решения опираясь на найденный контекст.

*if one Transition Action exception happens
→ other Transition Actions should be proceeded*



Останавливает от создания ошибки

Агент не делает то, что противоречит логике системы и установленным правилам.

Почему не Confluence?

Wiki создавалась для людей, а не для агентов.

\$

~\$10

в месяц на человека

Платить за всех

QA, BA, разработчики — лицензия каждому, кто хоть раз заглянул в вики.

** Это касается не только Confluence — у любой wiki те же проблемы.*

Почему не Confluence?

Wiki создавалась для людей, а не для агентов.



~\$10

в месяц на человека

Платить за всех

QA, BA, разработчики — лицензия каждому, кто хоть раз заглянул в вики.



Поиск

— это боль

Без прямой ссылки

Найти нужное без подсказки коллеги — сильно сложно.

* Это касается не только Confluence — у любой wiki те же проблемы.

Почему не Confluence?

Wiki создавалась для людей, а не для агентов.



~\$10

в месяц на человека

Платить за всех

QA, BA, разработчики — лицензия каждому, кто хоть раз заглянул в вики.



Поиск

— это боль

Без прямой ссылки

Найти нужное без подсказки коллеги — сильно сложно.



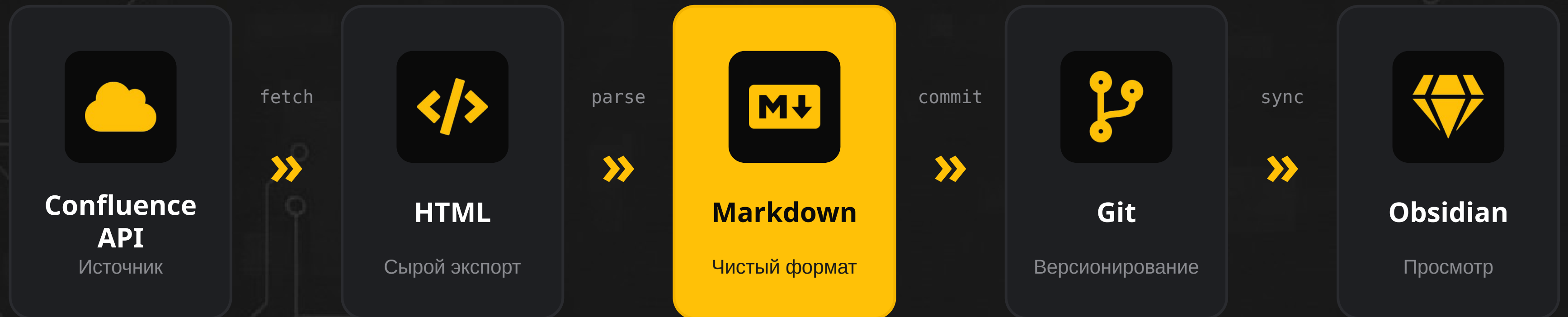
API

медленный






* Это касается не только Confluence — у любой wiki те же проблемы.

Процесс миграции

От Confluence API до читаемого Markdown — пять шагов.



Локальные файлы — игра меняется

 Confluence API	 Локальный Markdown
Поиск: несколько секунд	 Поиск по ~ 600 файлам: ~10 мс
Чтение 10 страниц: ещё пара секунд	 Параллельно: мгновенно
Переход по ссылке: отдельный запрос	 Wikilink: просто открыть файл

 600+ страниц · 1+ ГБ вложений — конвертированы за 1 день


А что делают ВА?

 Ставят Obsidian



 Подключают git plugin




 Клонировать репу с документацией

Public Job Body Template — Config Instructions

References


No Jira tickets referenced.

Motivation

 AI-generated — needs review

Provide administrators with step-by-step instructions for configuring the Public Job Body Template in Unlayer, including HTML structure, param usage, date formatting, and styling guidelines.

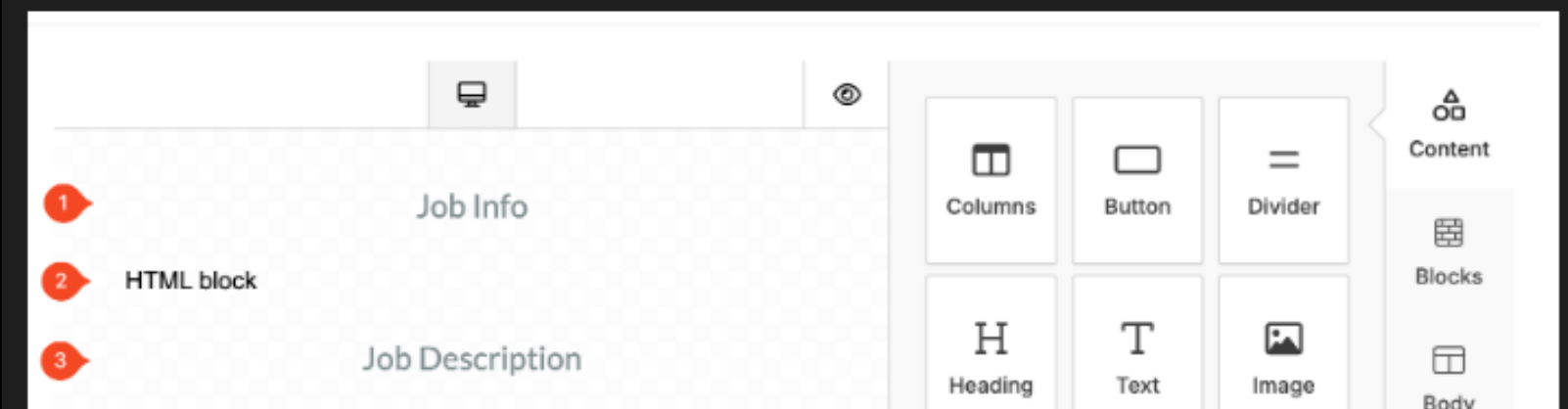
General Idea

 AI-generated — needs review

This page documents the configuration instructions for building Public Job Body Templates using the Unlayer editor. It covers template structure, HTML snippets for job info sections, param logic for conditional display, date formatting, and custom field usage.

Config instructions for Public Job Body Template

Template structure



Obsidian + Markdown

Взвесили плюсы и минусы — для нас плюсы перевесили.



Плюсы






- 🔍 Мгновенный локальный поиск
- 🔄 Git-версионирование из коробки
- 🔗 Wikilinks — связи между документами
- 📖 Агент читает напрямую, без API
- 💰 Бесплатно

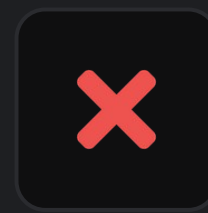
Obsidian + Markdown

Взвесили плюсы и минусы — для нас плюсы перевесили.







Плюсы

-  Мгновенный локальный поиск
-  Git-версионирование из коробки
-  Wikilinks — связи между документами
-  Агент читает напрямую, без API
-  Бесплатно



Минусы

-  Ещё один инструмент в стеке — Obsidian
-  ВА нужно освоить Markdown
-  Нет привычного WYSIWYG-редактора
-  Нет встроенного согласования и ревью



bash — summarize

```
$. ./summarize --section=docs
```

- > Документация: мигрирована ✓
- > Контекст для агента: готов ✓
- > VA workflow: настроен ✓

→ Запускаем pipeline...

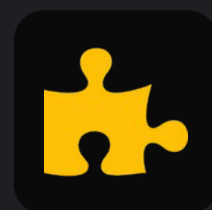


Что мы хотели от решения

Каждое требование напрямую влияет на архитектуру pipeline.

01

Под наш стек



Jira, Sentry, GitHub, Slack.

Что мы хотели от решения

Каждое требование напрямую влияет на архитектуру pipeline.

01

Под наш стек



Jira, Sentry, GitHub, Slack.

02

Микросервисы



Root cause часто в другом сервисе — агент должен видеть всю картину.

Что мы хотели от решения

Каждое требование напрямую влияет на архитектуру pipeline.

01

Под наш стек



Jira, Sentry, GitHub, Slack.

02

Микросервисы



Root cause часто в другом сервисе — агент должен видеть всю картину.

03

Отчётность



Каждый запуск прозрачен: что делал, сколько занял, сколько стоил.

Что мы хотели от решения

Каждое требование напрямую влияет на архитектуру pipeline.

01

Под наш стек



Jira, Sentry, GitHub, Slack.

02

Микросервисы



Root cause часто в другом сервисе — агент должен видеть всю картину.

03

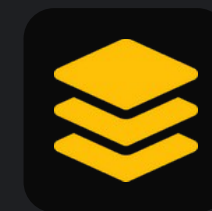
Отчётность



Каждый запуск прозрачен: что делал, сколько занял, сколько стоил.

04

Изоляция

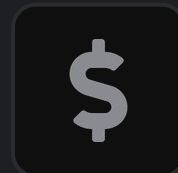


Параллельные фиксы разных проблем в одном репо без конфликтов.

Готовое vs своё

Сравнили подходы — у каждого свои ограничения и сильные стороны.

ГОТОВЫЕ РЕШЕНИЯ



Платные подписки

Devin, Codex, Copilot Agent — функционал, который нам не нужен



Готовое vs своё

Сравнили подходы — у каждого свои ограничения и сильные стороны.

ГОТОВЫЕ РЕШЕНИЯ



Платные подписки

Devin, Codex, Copilot Agent — функционал, который нам не нужен



Чёрный ящик

Не видно, какой контекст, где принимаются решения



Готовое vs своё

Сравнили подходы — у каждого свои ограничения и сильные стороны.

ГОТОВЫЕ РЕШЕНИЯ



Платные подписки

Devin, Codex, Copilot Agent — функционал, который нам не нужен



Чёрный ящик

Не видно, какой контекст, где принимаются решения



Не занимаются майнингом данных

Не идут сами в Jira+Sentry, нет approval-флоу из коробки



Готовое vs своё

Сравнили подходы — у каждого свои ограничения и сильные стороны.

ГОТОВЫЕ РЕШЕНИЯ



Платные подписки

Devin, Codex, Copilot Agent — функционал, который нам не нужен



Чёрный ящик

Не видно, какой контекст, где принимаются решения



Не занимаются майнингом данных

Не идут сами в Jira+Sentry, нет approval-флоу из коробки



✦ СВОЁ РЕШЕНИЕ



Полный контекст

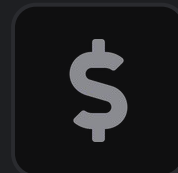
Документация, репо, история багов



Готовое vs своё

Сравнили подходы — у каждого свои ограничения и сильные стороны.

ГОТОВЫЕ РЕШЕНИЯ



Платные подписки

Devin, Codex, Copilot Agent — функционал, который нам не нужен



Чёрный ящик

Не видно, какой контекст, где принимаются решения



Не занимаются майнингом данных

Не идут сами в Jira+Sentry, нет approval-флоу из коробки



✦ СВОЁ РЕШЕНИЕ



Полный контекст

Документация, репо, история багов



Контроль

Знаем что и почему делает агент



Готовое vs своё

Сравнили подходы — у каждого свои ограничения и сильные стороны.

ГОТОВЫЕ РЕШЕНИЯ



Платные подписки

Devin, Codex, Copilot Agent — функционал, который нам не нужен



Чёрный ящик

Не видно, какой контекст, где принимаются решения



Не занимаются майнингом данных

Не идут сами в Jira+Sentry, нет approval-флоу из коробки



✦ СВОЁ РЕШЕНИЕ



Полный контекст

Документация, репо, история багов



Контроль

Знаем что и почему делает агент



Кастомизация процессов

бранчевание, управление этапами



Жизненный цикл тикета

TRIGGER



Slack

Jira URL в канале



Ilya 7:08 PM

@FixBot <https://evadotai.atlassian.net/browse/DEV-33333>

Jira Cloud ▼



UATMH | Agent | MAT user | Contact not found message is ...

Bug DEV-33333 in Jira Cloud

Status

In Progress

Priority

High

Assignee



Ilya (you)

As of Thursday at 7:08 PM

Open in Jira

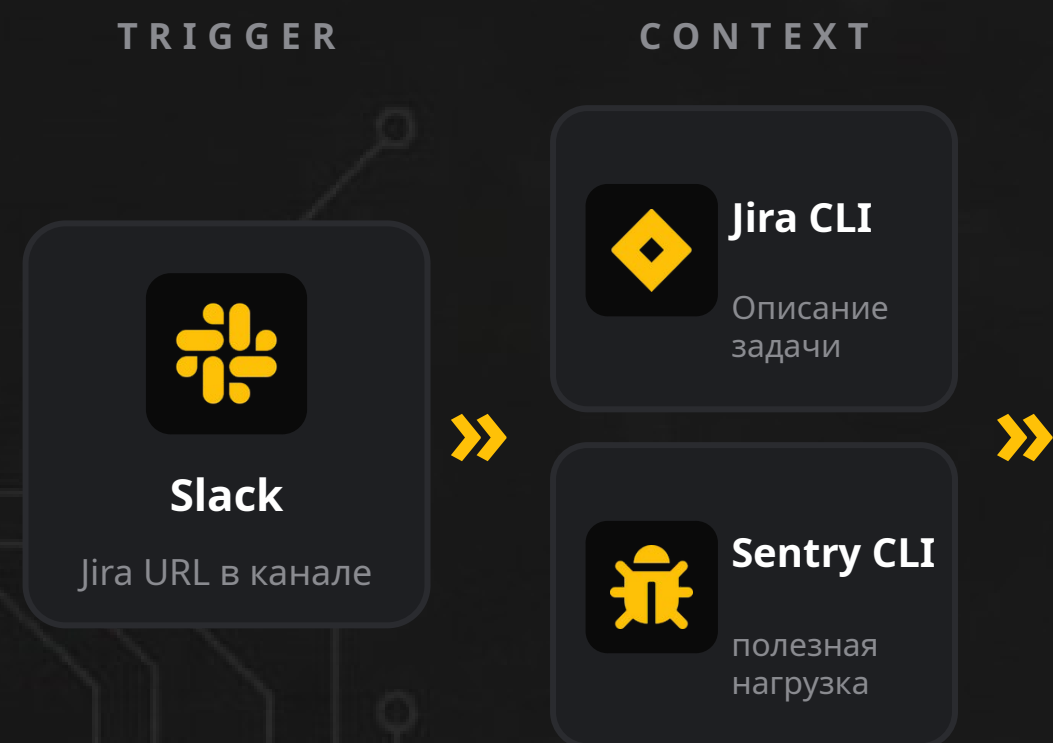
Summarise



1

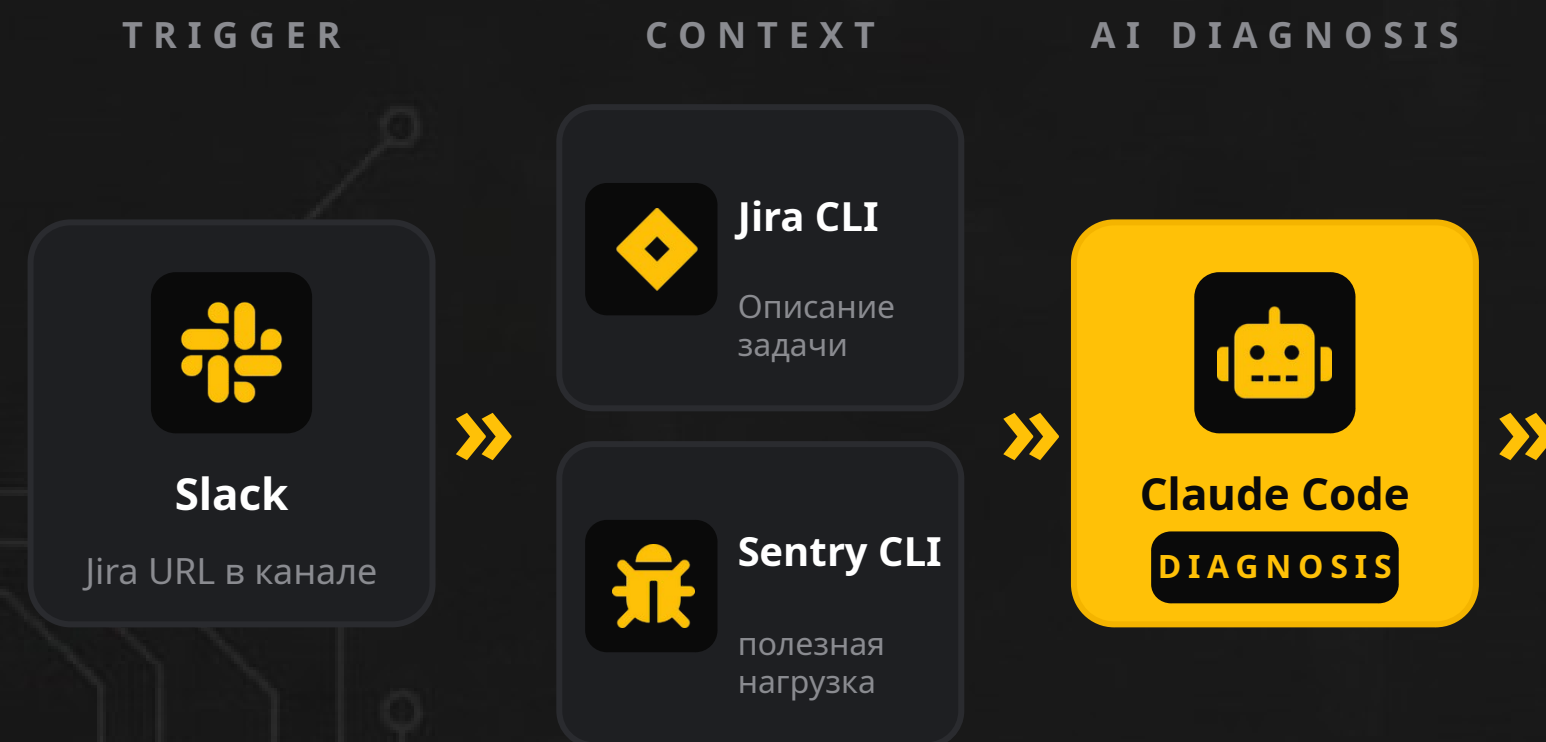


Жизненный цикл тикета



```
19 # JIRA – через CLI
18 ticket = JiraTicket(
17     key="DEV-32554",
16     summary=fields["summary"],
15     description=fields["description"],
14     fix_version=parse_fix_version(fields["fixVersions"]),
13     sentry_info=parse_sentry_info(description),
12     comments=format_comments(fields["comment"]),
11     attachments=download_attachments(fields["attachment"]),
10 )
9
8 # SENTRY – CLI
7 issue = SentryIssue(
6     stacktrace=parse_event_stacktrace(event),
5     breadcrumbs=event["breadcrumbs"],
4     request_data=event["request"]["data"],
3     request_headers=event["request"]["headers"],
2     replay_id=event["contexts"]["replay"]["id"],
1     tag_anomalies=detect_anomalies(tags),
20 )
```

Жизненный цикл тикета



```
function runAgent({ prompt, workingDir, role, model }) {  
  const args = [  
    "-p",  
    "--output-format", "json",  
    "--dangerously-skip-permissions",  
    "--system-prompt", role,  
    "--model", model,  
    prompt,  
  ];  
  
  const { stdout } = spawnSync("claude", args, { cwd: workingDir });  
  const data = JSON.parse(stdout);  
  
  return {  
    text: data.result,  
    usage: data.usage,  
    model: data.model,  
  };  
}
```

Жизненный цикл тикета



FixBot APP Thursday at 7:12 PM

DEV-33333 | chat-back-end → stage | **Low-Medium** Relaxing

`required: true` on `includeCompany` for `getContact` is a localized change. The only consumers of the returned VM that read `company` are already null-tolerant (`Contact.ts:233-234` returns `undefined` when `this.company` is missing). The risk is greater if the default of `includeCompany` itself is changed, because list/search pages may rely on the inner join to filter out company-less contacts (regression to verify in tests).

See the attached report for details.
Reply fix to proceed or skip to pass.

Diagnosis Report ▾

```
1 # DEV-33333 – Diagnosis Report
2 **Generated:** 2026-04-30T19:12:04
3
4 # Root Cause Analysis: Contact Not Found for active MAT Contact
5
6 ## Root Cause
7
8 The 'ContactsOrchestrator.getContact(contactId)' query (used by 'GET /contacts/:id', the endpoint that backs "launch contact profile") performs an **INNER JOIN with 'Company'** because 'ContactQueryBuilder.includeCompany()' defaults to 'required: true' ('src/services/query-builder/ContactQueryBuilder.ts:19'). If the joined 'Company' row is **soft-deleted** (the 'Company' model is 'paranoid: true', 'src/models/Company.ts:171') or otherwise unreachable, Sequelize's paranoid filter excludes it from the JOIN, and 'findByPk(contactId)' returns 'null'. The orchestrator then throws 'NotFoundError('Contact', contactId)' ('ContactsOrohestrator.ts:154-156'), which the UI surfaces as
```



Ilya Thursday at 7:14 PM

fix

Жизненный цикл тикета



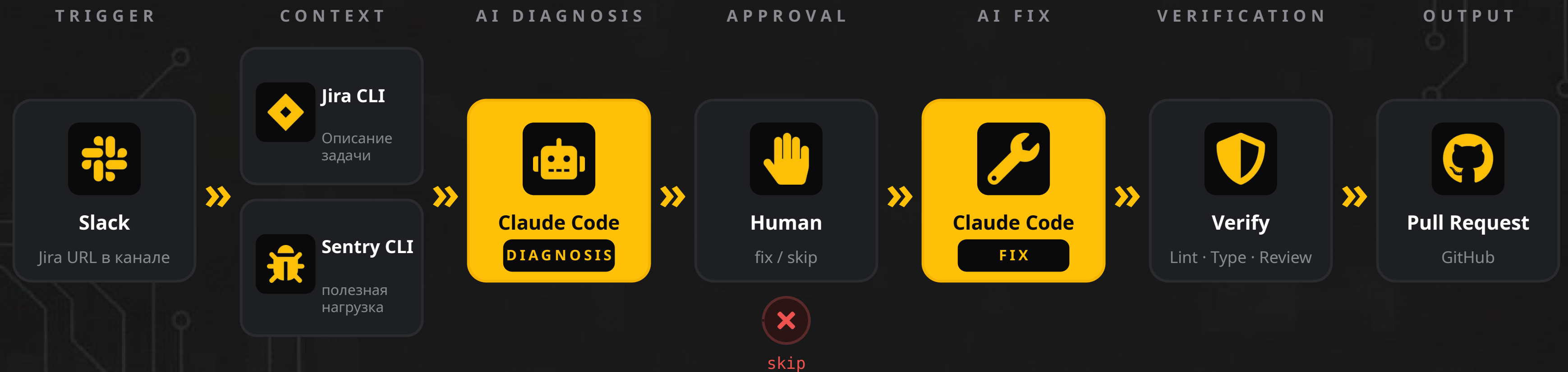
```
FixBot APP Thursday at 7:23 PM
DEV-33333 -- Self-Review ▶

✅ Fix applied and PR created
PR: https://github.com/LuckyLinkTeam/chat-back-end/pull/5598
Branch: DEV-33333-uatmh-agent-mat-user-contact-not-found-messag
displayed-while-laun → stage

📊 Metrics (model: claude-opus-4-7[1m])
🕒 Total: 14 minutes and 50.42 seconds (resolve_repo: 11.28 second
resolve_branch: 0.35 seconds, sentry_fetch: 0.36 seconds, diagnosis:
and 25.72 seconds, approval: 2 minutes and 6.50 seconds, fix: 1 minu
29.35 seconds, verification: 4 minutes and 32 seconds, self_review: 4
seconds, git_pr: 5.03 seconds)
📄 Input: 35 | Cache read: 1,345,263 | Cache write: 109,334 | Output
💰 Estimated cost: $1.72

DEV-33333 -- Fix Summary ▼
1 # DEV-33333 -- Fix Summary
2 **Generated:** 2026-04-30T19:23:14
3
4 ## Diagnosis
5 # Root Cause Analysis: Contact Not Found for active I
Contact
```

Жизненный цикл тикета



Что сохраняется по тикету

Любой тикет можно поднять — отдельная команда открывает все отчёты и стартует дискуссию.

01



Метаданные тикета

Jira ID, fix version, автор, описание

02



Контекст из Sentry

Stack trace, request body, breadcrumbs

03



Диагностика

Root cause, data flow, proposed fix

04



Результат фикса

Изменённые файлы + верификация

05



Review

Замечания AI-ревью с severity

06



Итоговый статус

merged · skipped · failed

07



Метрики

Время, токены, стоимость

DEV-33333

01-ticket.md

01b-sentry.md

02-diagnosis.md

03-fix-summary.md

03b-review.md

04-result.md

05-metrics.md

Почему CLI и почему Claude Code

Используем то, за что уже платим — и то, что уже знает наш репозиторий.



Уже есть подписка

Pipeline запускает тот же CLI как subprocess.
Никаких новых счетов.



bash

```
claude -p "List all API endpoints" --output-format json
```

Почему CLI и почему Claude Code

Используем то, за что уже платим — и то, что уже знает наш репозиторий.



Уже есть подписка

Pipeline запускает тот же CLI как subprocess.
Никаких новых счетов.



Не привязаны к Claude

Headless есть у всех — Codex, Gemini.
Переход на другой агент — пара часов работы.



bash

```
claude -p "List all API endpoints" --output-format json
```

Почему CLI и почему Claude Code

Используем то, за что уже платим — и то, что уже знает наш репозиторий.



Уже есть подписка

Pipeline запускает тот же CLI как subprocess.
Никаких новых счетов.



Не привязаны к Claude

Headless есть у всех — Codex, Gemini.
Переход на другой агент — пара часов работы.



Знает наш репозиторий

В каждом репо живёт CLAUDE.md — конвеншены, стек.
Агент сразу в контексте.













bash

```
claude -p "List all API endpoints" --output-format json
```

Разные баги — один pipeline

Разные стеки, разные слои, разная сложность — агент справляется.

ТИП БАГА	ЧТО СДЕЛАЛ АГЕНТ	ВРЕМЯ	СТОИМОСТЬ
 Рассогласование сервисов	Sender не клал поле в событие (edge case) — ловилось только в receiver через RabbitMQ	12 мин	~\$1
 ТОСТОУ и устаревшие данные	Форма ушла со старым списком кандидатов; ослабил валидацию до проверки вхождения	10 мин	~\$0.5
 Правильное решение — не фиксить	По headers и доке определил, что это E2E-тест шлёт несуществующее поле — отдал диагноз	—	~\$0.3
 Deadlock в PostgreSQL	Нашёл инверсию lock ordering между двумя потоками	13 мин	~\$1
 Stack overflow: Redis + spread	20K ключей через spread переполняют call stack — добавил батчинг	11 мин	\$0.17
 Error wrapping скрывает statusCode	Ошибка теряет statusCode при оборачивании — фикс в 1 строку	5 мин	~\$0.5
 Array вместо string от frontend	Frontend шлёт массив — добавил нормализацию	5 мин	\$0.80
 Email > 200 символов роняет БД	Ни один слой не проверяет длину — добавил валидацию	10 мин	~\$0.5
 Scheduled task + бизнес-логика	Прочитал доку — выбрал graceful handling вместо throw	14 мин	~\$0.5
 Невалидная комбинация от n8n	По headers — это n8n; по доке подтвердил, что комбинация невалидна	10 мин	\$1.00

Четыре роли — четыре промпта

СИСТЕМНЫЙ ПРОМПТ CLAUDE CODE

"You are Claude Code, Anthropic's official CLI for Claude. You are an interactive agent that helps users with software engineering tasks."



Диагностика

You are a senior backend engineer performing root cause analysis. Read code carefully before reasoning. Never speculate about code you have not opened.

Четыре роли — четыре промпта

СИСТЕМНЫЙ ПРОМПТ CLAUDE CODE

"You are Claude Code, Anthropic's official CLI for Claude. You are an interactive agent that helps users with software engineering tasks."



Диагностика

You are a senior backend engineer performing root cause analysis. Read code carefully before reasoning. Never speculate about code you have not opened.



Фикс

You are a senior backend engineer fixing a production bug. Make the smallest possible change. Match the existing code style exactly.

Четыре роли — четыре промпта

СИСТЕМНЫЙ ПРОМПТ CLAUDE CODE

"You are Claude Code, Anthropic's official CLI for Claude. You are an interactive agent that helps users with software engineering tasks."



Диагностика

You are a senior backend engineer performing root cause analysis. Read code carefully before reasoning. Never speculate about code you have not opened.



ФИКС

You are a senior backend engineer fixing a production bug. Make the smallest possible change. Match the existing code style exactly.



Type check fix

You are a senior backend engineer fixing type errors after a bug fix. Preserve the bug fix. Only fix the listed type errors, nothing else.

Четыре роли — четыре промпта

СИСТЕМНЫЙ ПРОМПТ CLAUDE CODE

"You are Claude Code, Anthropic's official CLI for Claude. You are an interactive agent that helps users with software engineering tasks."



Диагностика

You are a senior backend engineer performing root cause analysis. Read code carefully before reasoning. Never speculate about code you have not opened.



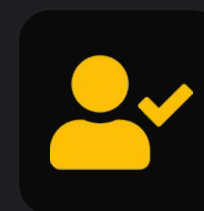
ФИКС

You are a senior backend engineer fixing a production bug. Make the smallest possible change. Match the existing code style exactly.



Type check fix

You are a senior backend engineer fixing type errors after a bug fix. Preserve the bug fix. Only fix the listed type errors, nothing else.



Code review

You are a senior code reviewer examining an AI-generated bug fix. Focus on correctness, edge cases, and whether the fix addresses the root cause.

Diagnosis prompt

1

Данные сверху

Sentry + Jira в XML-тегах идут первыми, инструкции — после.

2

Сначала прочитай код

Без чтения агент уходит в догадки по stack trace.

3

Поиск по документации

После анализа кода, до диагноза — ищет дополнительный контекст.

4

Самопроверка

Если вывод «код недостижим» — анализ неполный, вернись и проверь.

5

Структурированный вывод

Root Cause / Data Flow / Proposed Fix парсятся автоматически.

diagnosis_prompt.md

```
<error>{error_title}</error>
```

```
<sentry>{sentry}</sentry>
```

```
<jira>
```

```
<description>{jira_description}</description>
```

```
<comments>{jira_comments}</comments>
```

```
</jira>
```

Diagnosis prompt

1

Данные сверху

Sentry + Jira в XML-тегах идут первыми, инструкции — после.

2

Сначала прочитай код

Без чтения агент уходит в догадки по stack trace.

3

Поиск по документации

После анализа кода, до диагноза — ищет дополнительный контекст.

4

Самопроверка

Если вывод «код недостижим» — анализ неполный, вернись и проверь.

5

Структурированный вывод

Root Cause / Data Flow / Proposed Fix парсятся автоматически.

diagnosis_prompt.md

```
## Pre-Analysis (do this FIRST,
before writing any diagnosis)
1. Review Jira comments carefully
2. Analyze all data in Sentry context
3. Read source files in the stack trace
4. Find all callers – trace upstream
5. Find similar patterns in the codebase
6. Check existing test coverage
```

Diagnosis prompt

1

Данные сверху

Sentry + Jira в XML-тегах идут первыми, инструкции — после.

2

Сначала прочитай код

Без чтения агент уходит в догадки по stack trace.

3

Поиск по документации

После анализа кода, до диагноза — ищет дополнительный контекст.

4

Самопроверка

Если вывод «код недостижим» — анализ неполный, вернись и проверь.

5

Структурированный вывод

Root Cause / Data Flow / Proposed Fix парсятся автоматически.

diagnosis_prompt.md

```
<docs_phase>
```

```
## Then Diagnose
```

```
(using both code AND documentation)
```

```
1. What broke – the immediate error
```

```
2. Why it broke – trace data flow
```

```
3. Root cause – actual origin
```

```
4. Why it wasn't caught – missing test
```

```
5. Similar patterns at risk
```

Diagnosis prompt

1

Данные сверху

Sentry + Jira в XML-тегах идут первыми, инструкции — после.

2

Сначала прочитай код

Без чтения агент уходит в догадки по stack trace.

3

Поиск по документации

После анализа кода, до диагноза — ищет дополнительный контекст.

4

Самопроверка

Если вывод «код недостижим» — анализ неполный, вернись и проверь.

5

Структурированный вывод

Root Cause / Data Flow / Proposed Fix парсятся автоматически.

diagnosis_prompt.md

Unreachable Code Sanity Check

If your analysis concludes "unreachable" or "should never fire" – STOP and reconsider.

Common blind spots:

- Empty collections, skipped loops
- Callers bypassing the entry point
- Race conditions between async ops
- Previous code versions deployed

Diagnosis prompt

1

Данные сверху

Sentry + Jira в XML-тегах идут первыми, инструкции — после.

2

Сначала прочитай код

Без чтения агент уходит в догадки по stack trace.

3

Поиск по документации

После анализа кода, до диагноза — ищет дополнительный контекст.

4

Самопроверка

Если вывод «код недостижим» — анализ неполный, вернись и проверь.

5

Структурированный вывод

Root Cause / Data Flow / Proposed Fix парсятся автоматически.

diagnosis_prompt.md

```
## Respond With These Sections
## Root Cause
## Data Flow
## Affected Files
## Similar Patterns at Risk
## Proposed Fix
## Risk (Low / Medium / High)
## Affected Repos
```

Fix prompt

1

Данные из диагноза

Root Cause, Affected Files, Proposed Fix
в XML-тегах вверху.

2

Поиск вызывающих

Найди всех вызывающих функцию
проверь, что фикс не ломает их.

3

Выставление границ

Исправляй только root cause

4

Сохранение side effects

Логи, метрики, audit events — это observability,
не трогать.

fix_prompt.md

```
<root_cause>{root_cause}</root_cause>
```

```
<affected_files>  
  {affected_files}  
</affected_files>
```

```
<similar_patterns>  
  {similar_patterns}  
</similar_patterns>
```

```
<proposed_fix>{proposed_fix}</proposed_fix>
```

Fix prompt

1

Данные из диагноза

Root Cause, Affected Files, Proposed Fix
в XML-тегах вверху.

2

Поиск вызывающих

Найди всех вызывающих функцию
проверь, что фикс не ломает их.

3

Выставление границ

Исправляй только root cause.

4

Сохранение side effects

Логи, метрики, audit events — это observability,
не трогать.

fix_prompt.md

```
## Pre-Fix Research (do FIRST)
```

```
1. Study coding style:
```

```
    naming, error handling
```

```
2. Look at recent commits
```

```
3. Read existing tests
```

```
4. Check linter config
```

```
    (.eslintrc, ruff.toml)
```

```
## Pre-Fix Caller Check
```

```
5. Find all callers of functions you're about to  
modify. Verify exceptions and callbacks.
```

Fix prompt

1

Данные из диагноза

Root Cause, Affected Files, Proposed Fix в XML-тегах вверху.

2

Поиск вызывающих

Найди всех вызывающих функцию
проверь, что фикс не ломает их.

3

Выставление границ

Исправляй только root cause.

4

Сохранение side effects

Логи, метрики, audit events — это observability,
не трогать.

fix_prompt.md

Instructions

- Fix only the specific error from the root cause
- Match the existing code style exactly
- Limit changes to files directly involved in the reported error
- Similar Patterns at Risk are listed for context only – do not edit them
- Run existing tests if available

Fix prompt

1

Данные из диагноза

Root Cause, Affected Files, Proposed Fix
в XML-тегах вверху.

2

Поиск вызывающих

Найди всех вызывающих функцию
проверь, что фикс не ломает их.

3

Выставление границ

Исправляй только root cause

4

Сохранение side effects

Логи, метрики, audit events — это observability,
не трогать.

fix_prompt.md

- Keep side effects that aren't part of the bug:
 - logging
 - metrics
 - audit events

Свой code review — за и против

П Л Ю С Ы



Диагноз на входе + доступ к репе

Review-агент получает diff + готовый диагноз и запускается прямо в репе.



Свой code review — за и против

П Л Ю С Ы



Диагноз на входе + доступ к репе

Review-агент получает diff + готовый диагноз и запускается прямо в репе.



Контролируем всё

System prompt, формат вывода, логика вызовов
Наш стек целиком.



Свой code review — за и против

П Л Ю С Ы



Диагноз на входе + доступ к репе

Review-агент получает diff + готовый диагноз и запускается прямо в репе.



Контролируем всё

System prompt, формат вывода, логика вызовов
Наш стек целиком.



Multi-model ensemble

Claude и GPT параллельно — у моделей разные слепые пятна.



Свой code review — за и против

П Л Ю С Ы



Диагноз на входе + доступ к репе

Review-агент получает diff + готовый диагноз и запускается прямо в репе.



Контролируем всё

System prompt, формат вывода, логика вызовов
Наш стек целиком.



Multi-model ensemble

Claude и GPT параллельно — у моделей разные слепые пятна.



Расширение под свою архитектуру

Горячие файлы из git-истории, в таком случае severity автоматически выше.



Свой code review — за и против

П Л Ю С Ы



Диагноз на входе + доступ к репе

Review-агент получает diff + готовый диагноз и запускается прямо в репе.



Контролируем всё

System prompt, формат вывода, логика вызовов
Наш стек целиком.



Multi-model ensemble

Claude и GPT параллельно — у моделей разные слепые пятна.



Расширение под свою архитектуру

Горячие файлы из git-истории, в таком случае severity автоматически выше.



М И Н У С Ы



Поддержка на нас

Промпт, категории проверки, интеграция, retry-логика — наш код.



Свой code review — за и против

П Л Ю С Ы



Диагноз на входе + доступ к репе

Review-агент получает diff + готовый диагноз и запускается прямо в репе.



Контролируем всё

System prompt, формат вывода, логика вызовов
Наш стек целиком.



Multi-model ensemble

Claude и GPT параллельно — у моделей разные слепые пятна.



Расширение под свою архитектуру

Горячие файлы из git-истории, в таком случае severity автоматически выше.



М И Н У С Ы



Поддержка на нас

Промпт, категории проверки, интеграция, retry-логика — наш код.



UI хуже

У готовых — inline-комментарии в PR.
У нас — markdown в Slack.



Свой code review — за и против

П Л Ю С Ы



Диагноз на входе + доступ к репе

Review-агент получает diff + готовый диагноз и запускается прямо в репе.



Контролируем всё

System prompt, формат вывода, логика вызовов
Наш стек целиком.



Multi-model ensemble

Claude и GPT параллельно — у моделей разные слепые пятна.



Расширение под свою архитектуру

Горячие файлы из git-истории, в таком случае severity автоматически выше.



М И Н У С Ы



Поддержка на нас

Промпт, категории проверки, интеграция, retry-логика — наш код.



UI хуже

У готовых — inline-комментарии в PR.
У нас — markdown в Slack.



Нет коллективного опыта

Готовые тулы крутятся на тысячах репо.
Мы учимся на своих данных.



Две точки контроля

УРОВЕНЬ 1

Инструментальный



Линтер

автофикс форматирования



Type check

AI чинит свои же ошибки



До 2 раундов авто-починки

Две точки контроля

УРОВЕНЬ 1

Инструментальный



Линтер

автофикс форматирования



Type check

AI чинит свои же ошибки



До 2 раундов авто-починки

УРОВЕНЬ 2

AI Review



Вызов Claude code

после фикса, перед пушем



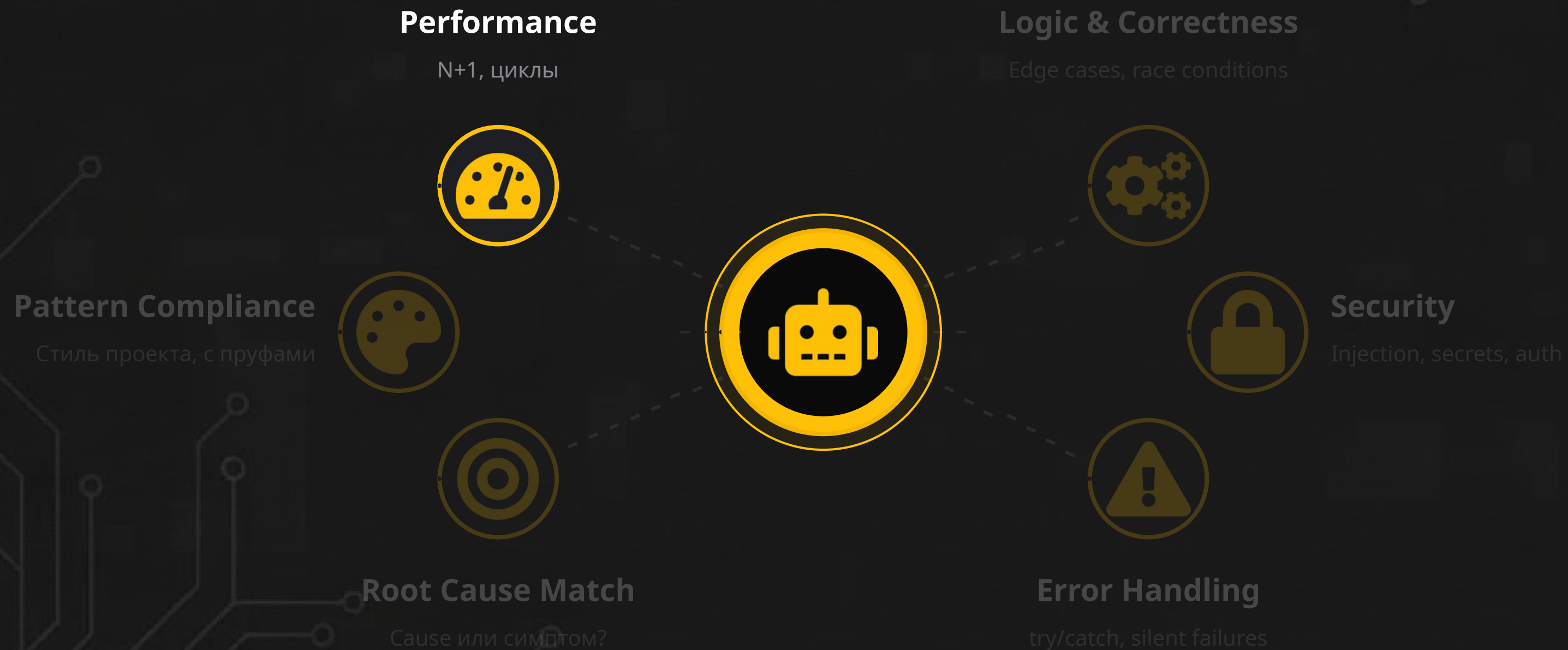
Multi-model ensemble

опционально: Claude + GPT параллельно

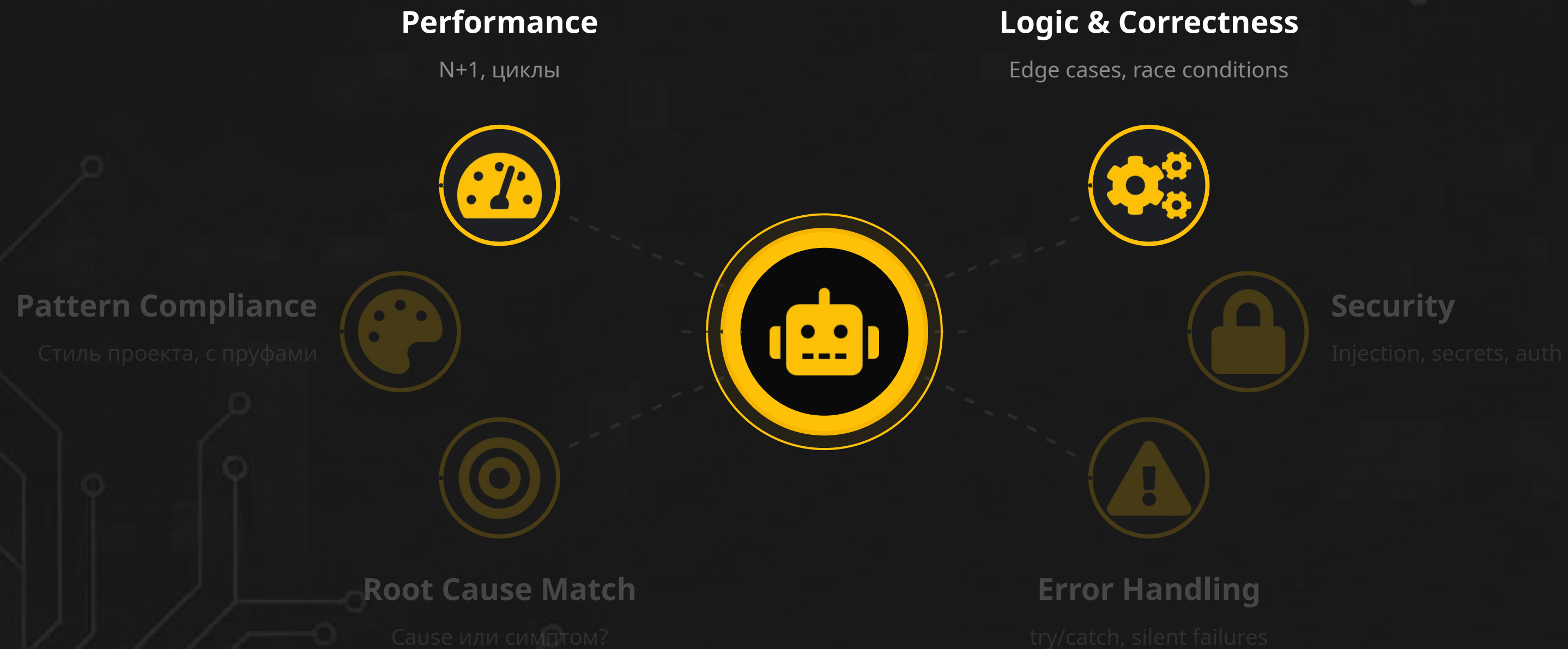


Проверка по 6 направлениями

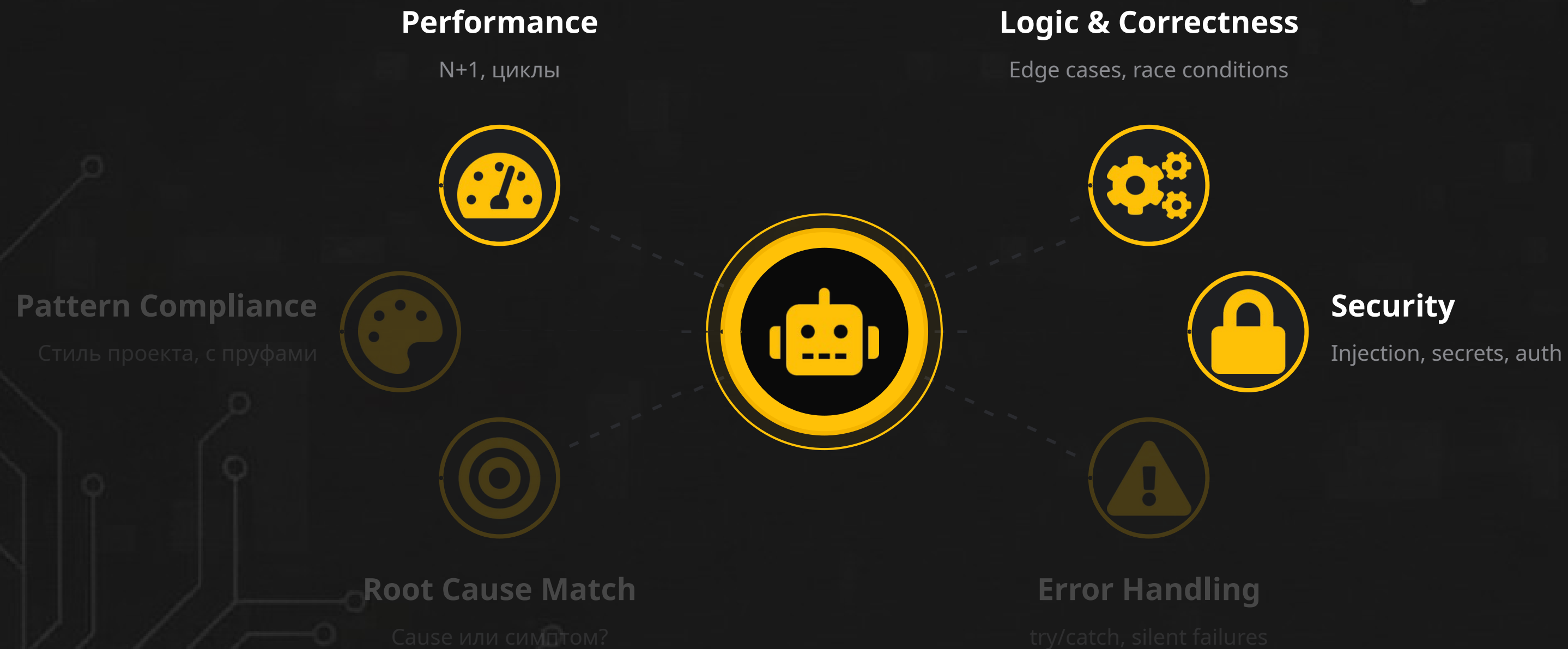
Review: 6 направлений



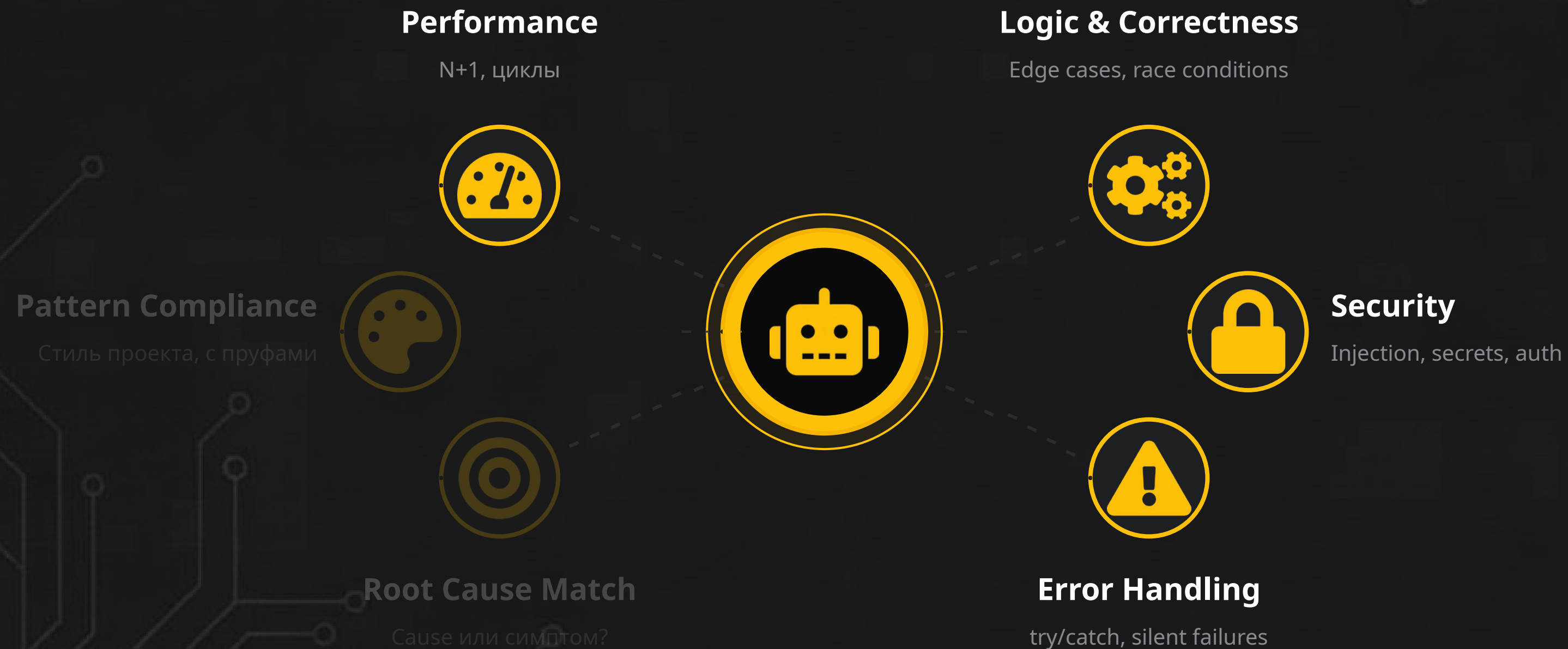
Review: 6 направлений



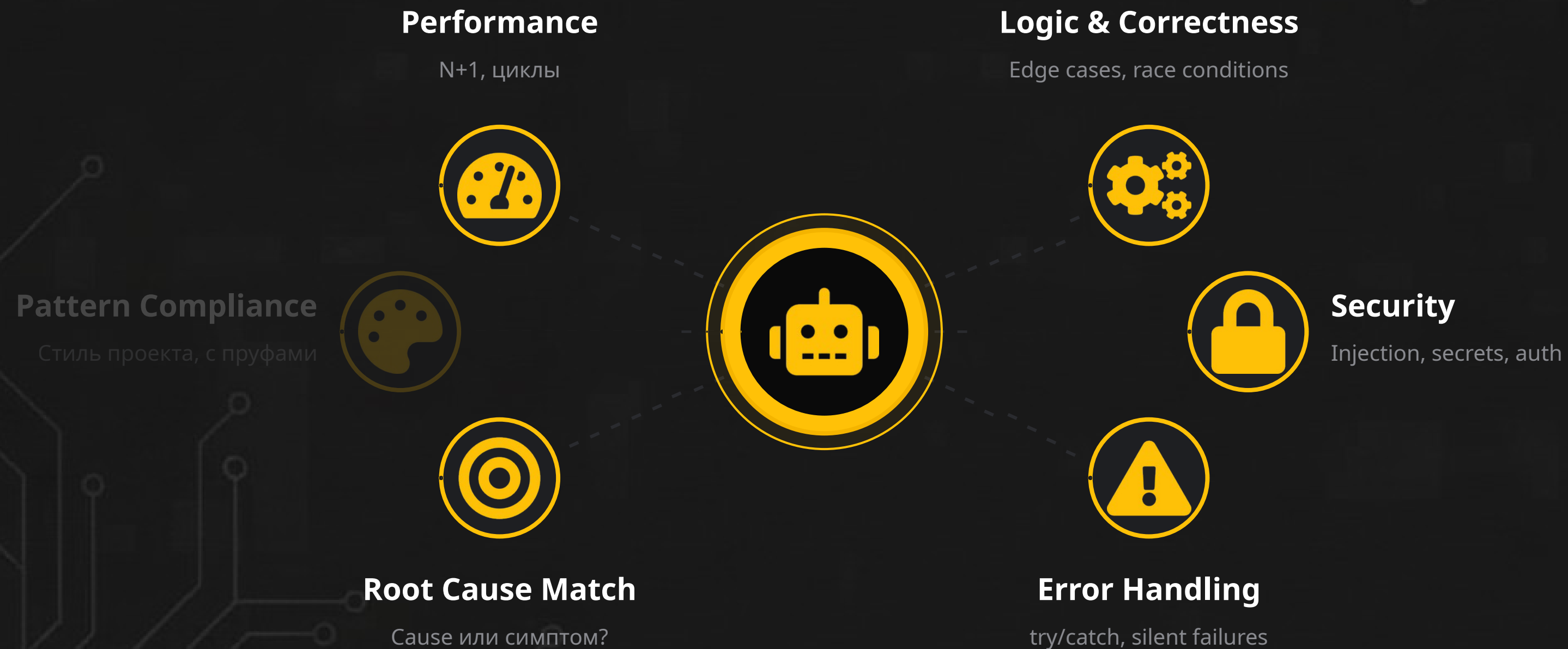
Review: 6 направлений



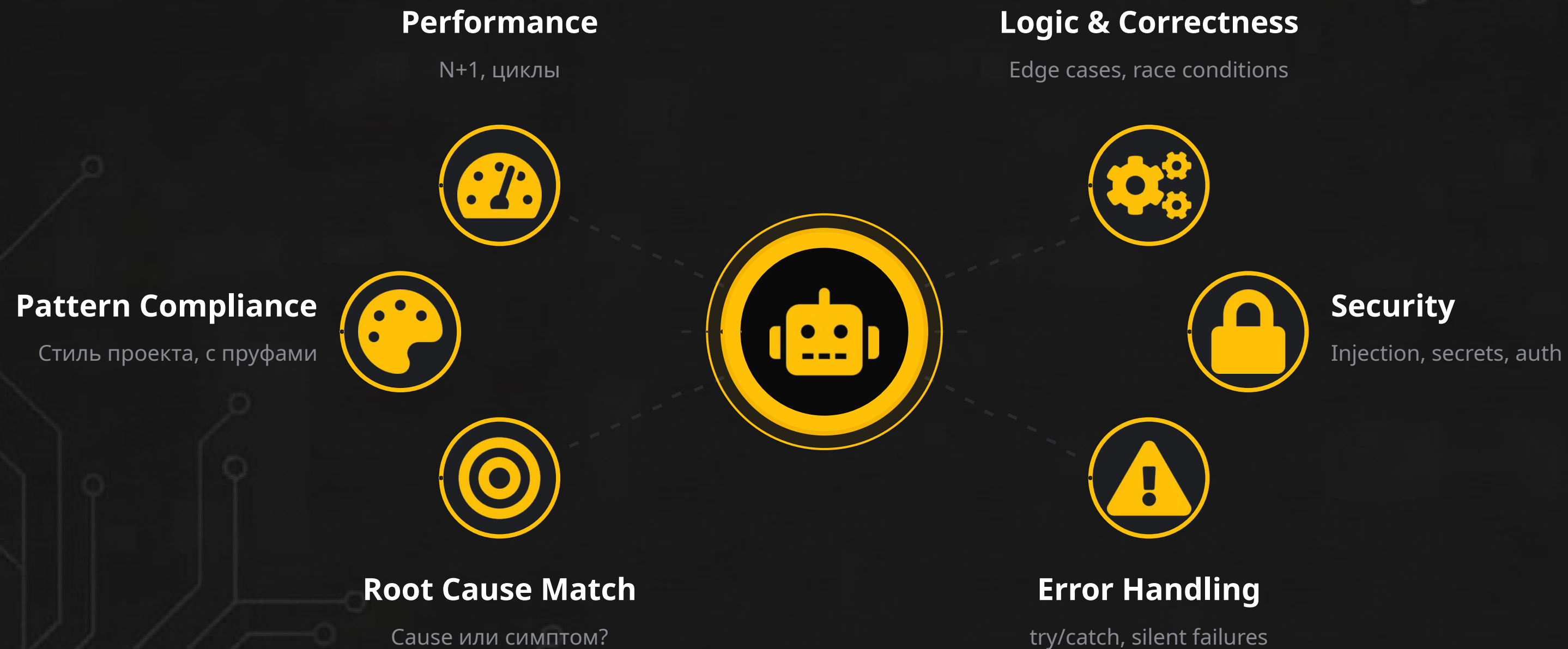
Review: 6 направлений



Review: 6 направлений



Review: 6 направлений



Цифры на сегодня

Среднее: 10 минут и ~80 центов на попытку фикса.



92%

доходят до PR

из всех запусков pipeline

Цифры на сегодня

Среднее: 10 минут и ~80 центов на попытку фикса.



92%

доходят до PR

из всех запусков pipeline



56%

PR merged в прод

из решённых тикетов

Цифры на сегодня

Среднее: 10 минут и ~80 центов на попытку фикса.



92%

доходят до PR

из всех запусков pipeline



56%

PR merged в прод

из решённых тикетов



~10 МИН

среднее время на баг

от Slack до PR

Цифры на сегодня

Среднее: 10 минут и ~80 центов на попытку фикса.



92%

доходят до PR

из всех запусков pipeline



56%

PR merged в прод

из решённых тикетов



~10 МИН

среднее время на баг

от Slack до PR



\$0.80

стоимость на баг

за токены LLM

Что pipeline пока не умеет

Очевидно — не серебряная пуля, но а разве они сейчас есть?



С фронтом пока не работаем



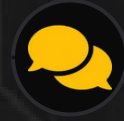
Неполная диагностика



Не учится на ошибках



Поддержка на нас










Нет интерактивности



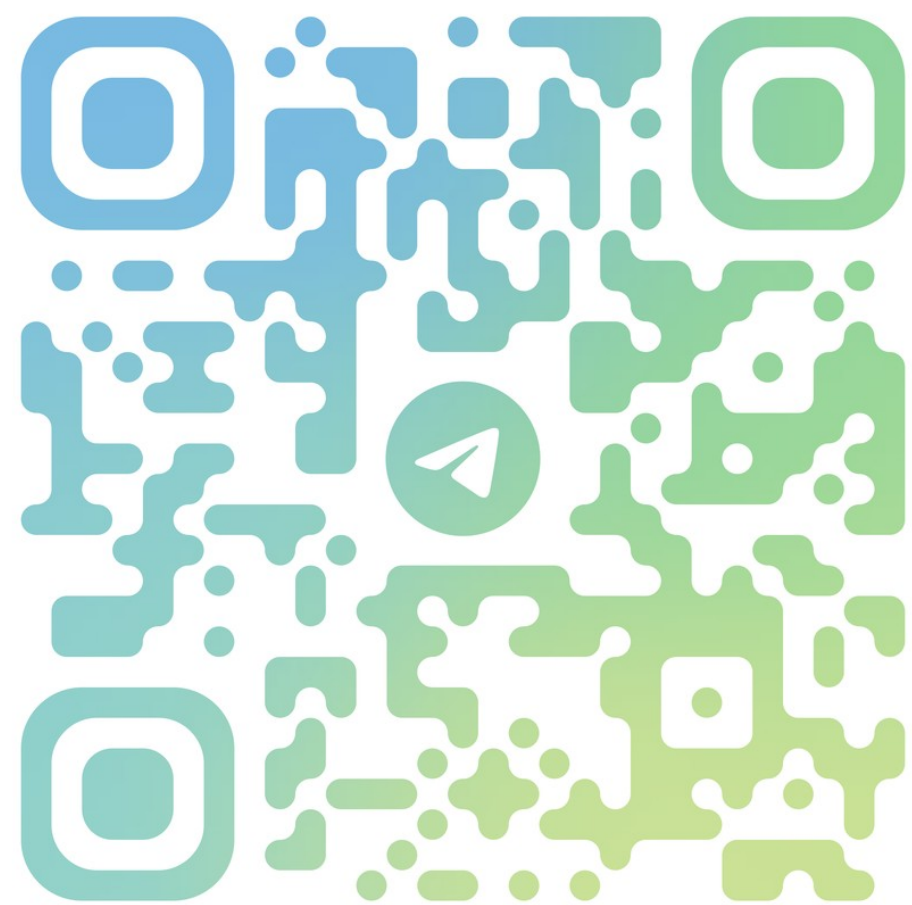
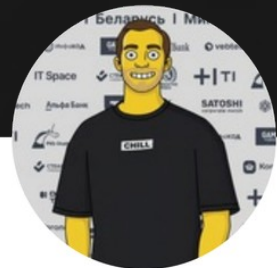
Human review обязателен

Что дальше

Roadmap: то, что хотим довести в ближайшие месяцы.

- 01  Разбор тикетов из Jira
- 02  Регрессионные тесты
- 03  Безопасное взаимодействие с БД
- 04  Свой UI для pipeline
- 05  Обучение на отклонённых
- 06  Несколько вариантов фикса
- 07  Мета-ревью второй моделью

Спасибо!



@IVANCHIKOVITCLUB

Илья



t.me/IlyaIvanchikov



linkedin.com/in/ilyaivanchikov