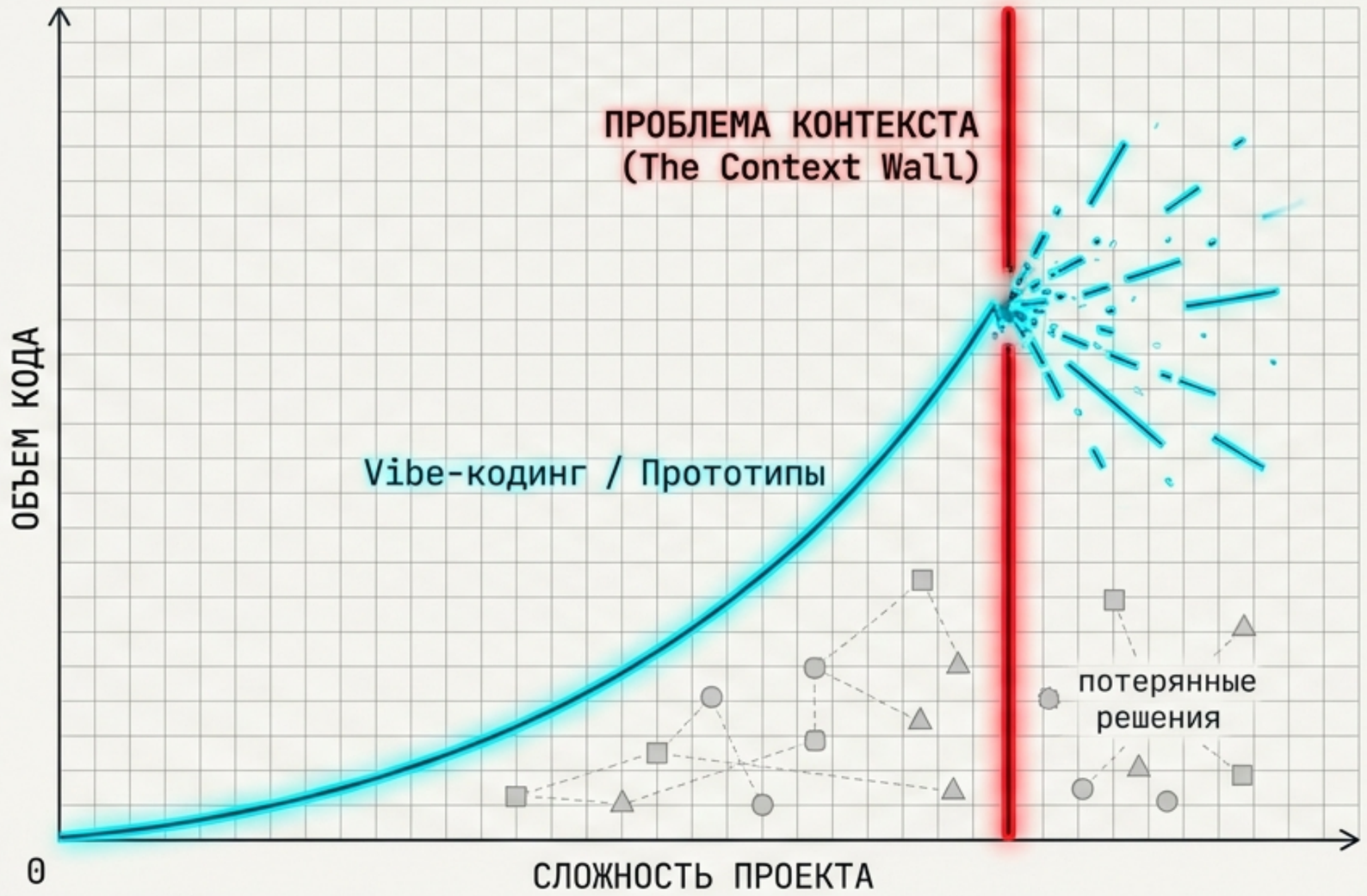
1
2
3
4
5

BMad Method V6: Архитектура AI-Driven Разработки

Как превратить хаос «vibe-кодинга» в предсказуемый Agile-процесс

```
> Day_1: Philosophy, Planning_Funnel & Execution.sh_
```

Генерация кода больше не проблема. Проблема — это контекст.



- Обычный vibe-кодинг идеален для прототипов, но ломается на этапе продакшена.
- Когда код пишется без спецификаций, архитектурные решения нельзя отследить.
- **Итог:** ИИ теряет контекст, ломает кодовую базу, а traceabirility стремится к нулю.

Ключевой постулат: Process, Not a Prompt

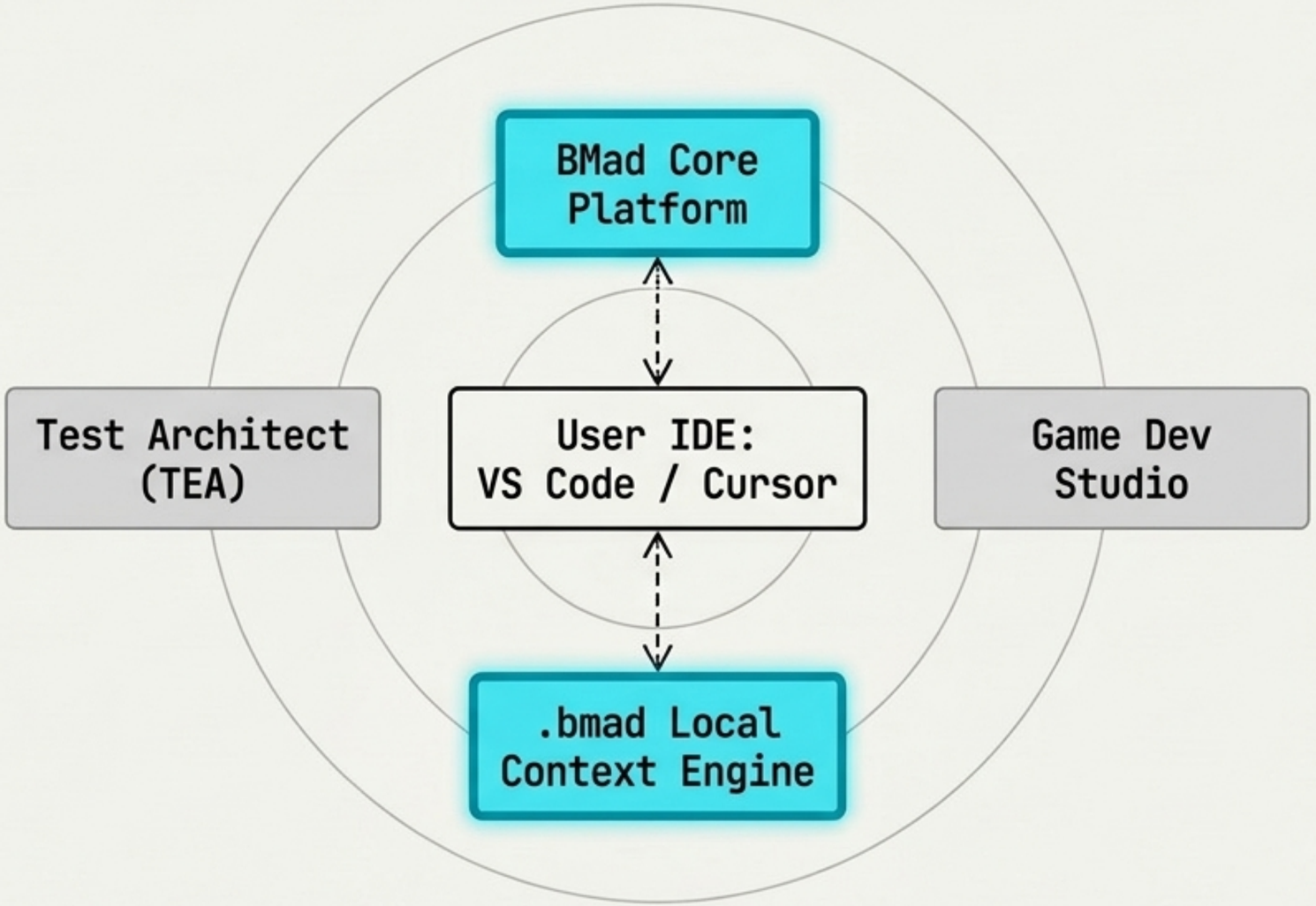
[Папка с промптами] ≠ [Enterprise Метод]

Задает правила, принимает архитектурные решения, проверяет гипотезы и утверждает планы.

Фасилитатор. Собирает контекст, задает сложные вопросы, пишет детерминированный код.

> **Key Insight:** Решения сохраняются в репозитории как Markdown и переживают сессию чата. Ничего не теряется при очистке контекста.

V6 — Масштабируемая локальная платформа



Local First

Вся логика и артефакты хранятся локально в директории **.bmad**. Никакой привязки к внешним базам.

IDE Integration

Нативная работа внутри **Claude Code, Cursor, Copilot** (45+ инструментов).

Modular

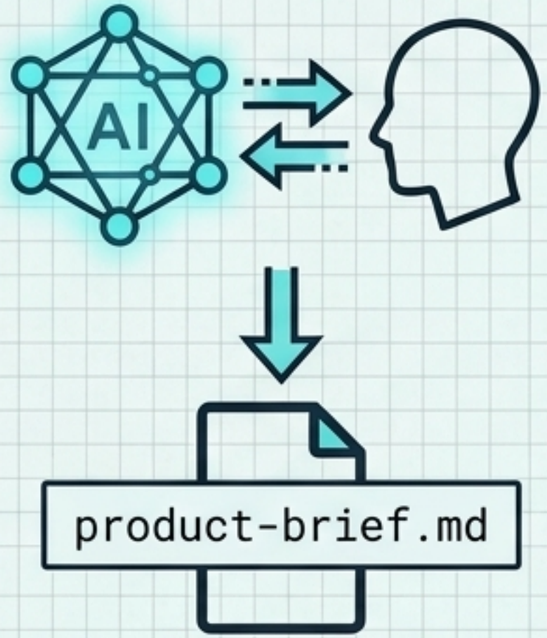
Масштабируется от простого багфикса до проектирования SaaS или GameDev с помощью изолированных модулей.

Воронка планирования: От туманной идеи к строгому коду

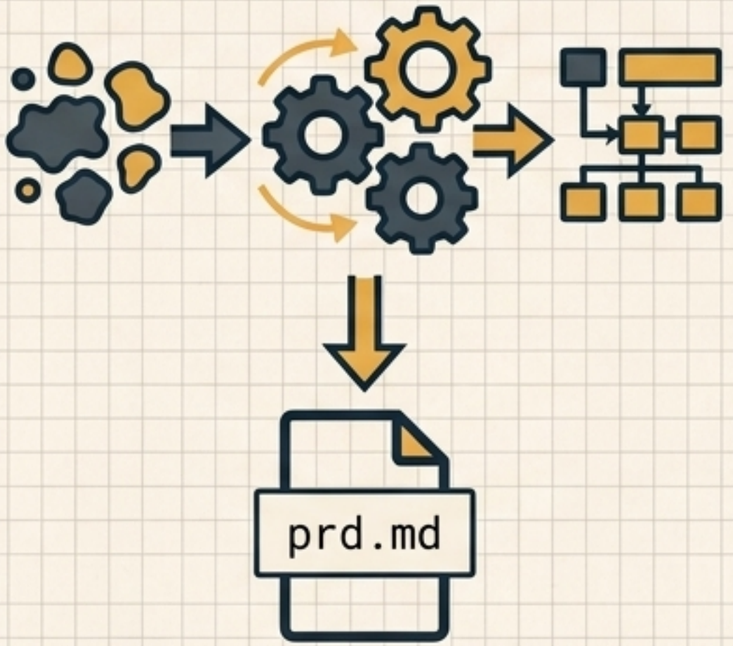
Агенты BMad не пытаются угадать всё сразу. Они передают концентрированный контекст друг другу.

На каждом этапе отсекается лишнее, а результат фиксируется в виде технических артефактов. Инженерный замысел сохраняется на 100%.

Этапы 1 и 2: Брейншторм и Требования

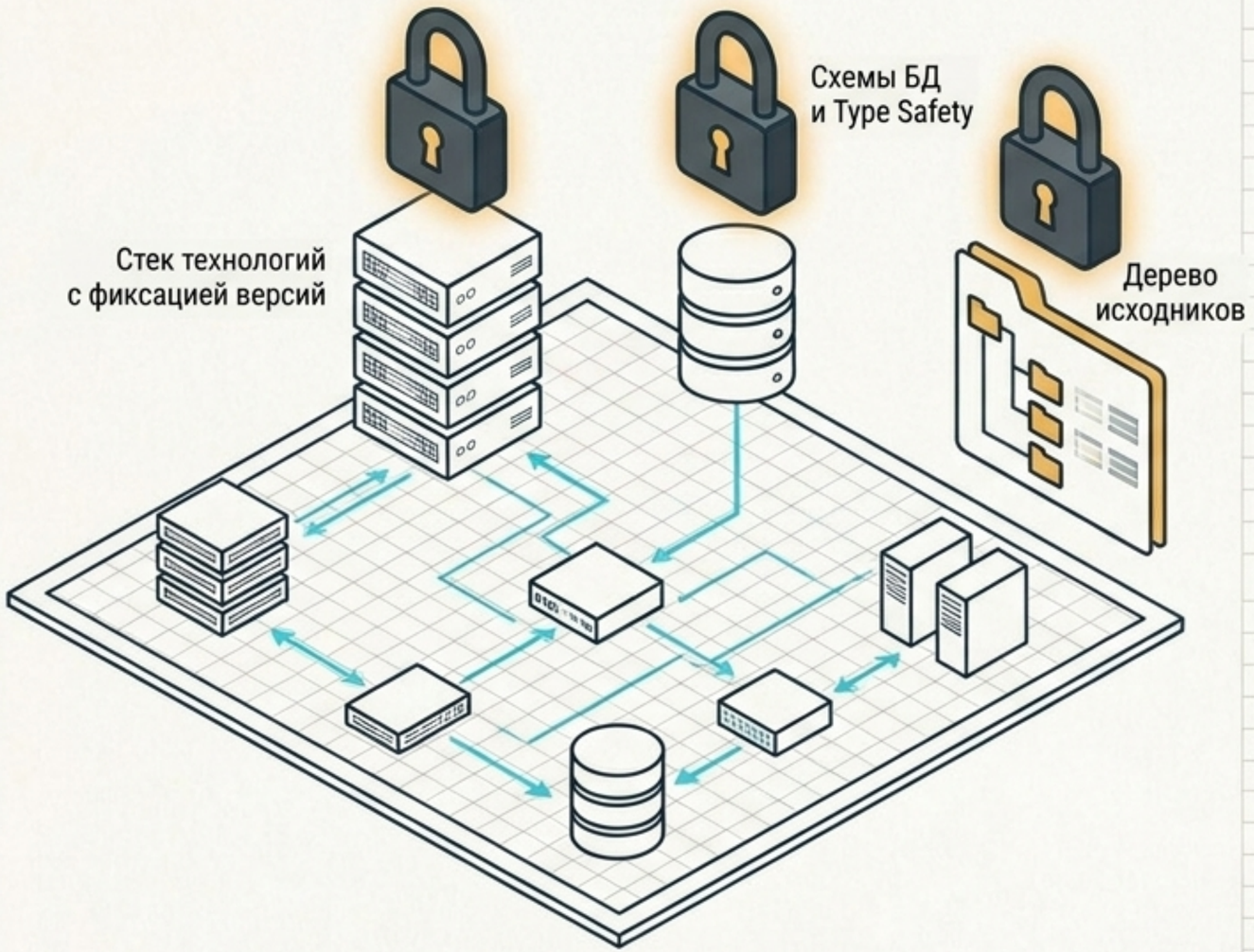


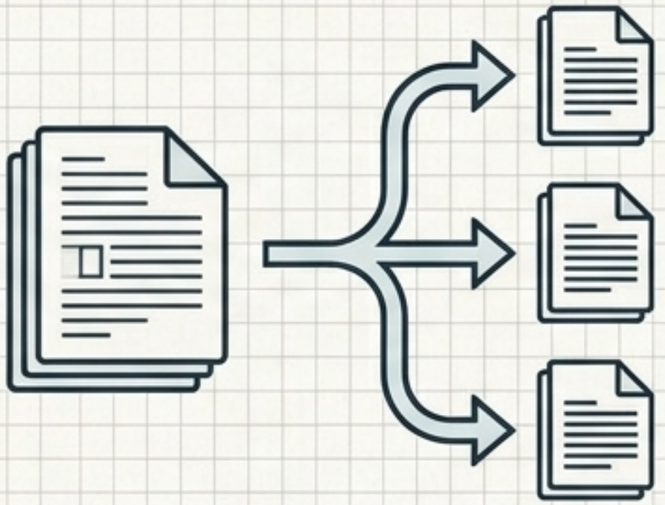
Analyst: ИИ не думает за вас, он задает сложные вопросы (Advanced Elicitation). Цель — выявить скрытые ограничения до того, как они сломают архитектуру.



PM: Преобразование брифа в функциональные и нефункциональные требования. Формирование четких Эпиков (Epics) и границ MVP.

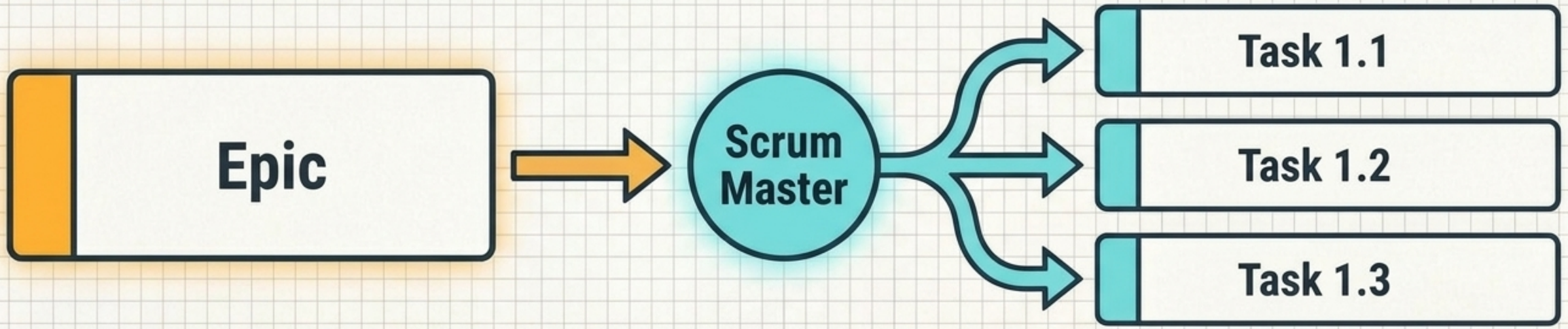
Этап 3: Архитектура и Дробление Контекста





Crucial Concept: Sharding (Дробление)
Огромный архитектурный документ автоматически дробится на атомарные файлы. Это гарантирует, что ИИ-кодер загрузит только нужный контекст, избежав галлюцинаций от переизбытка данных.

Этап 4: Эпики, Истории и Детерминированный Разработчик



Scrum Master берет PRD и Архитектуру, нарезая их на атомарные, строго последовательные истории (User Stories).

```
{
  "dev-load-always-files": [
    "tech-stack.md",
    "coding-standards.md"
  ]
}
```

Dev Agent намеренно «глупый». Он не принимает архитектурных решений, а читает конкретную историю и выполняет задачу с хирургической точностью.

Scale Adaptive: Адаптивность под масштаб задачи

Quickflow

Full Flow

Вам не нужно проходить всю воронку агентов (Analyst -> PM -> Architect) для исправления одной опечатки.

Архитектура подстраивается под вас. Вы сами выбираете нужный уровень формализации в зависимости от сложности тикета.

Сравнение режимов выполнения

Feature	Full Flow	Quickflow
Сценарий	Запуск SaaS, проектирование крупной фичи	Багфикс, точечное улучшение существующей функции
Процесс	Полная воронка агентов (Analyst -> PM -> Arch -> Dev)	Прямая команда разработчику (bmad-build)
Артефакты	PRD, Архитектура, Эпики, Истории, Код	Только Код и Ревью
Скорость	Дни / Недели	Секунды / Минуты

Работа в существующем коде (Brownfield)

- Большинство задач — это не Greenfield (разработка с нуля). BMad умеет работать с существующей кодовой базой.
- Запуск команды генерации контекста заставляет агентов изучить ваше легаси и создать карту проекта.
- Архитектор и PM имеют специальные brownfield режимы для учета существующих ограничений и зависимостей.

Практика: Документация по стандарту Diátaxis

`docs.bmad-method.org`

Tutorials
(Quick Start)

How-To Guides
(Task-oriented)

Explanation
(Theory/Architecture)

Reference
(Commands/Agents)

Документация V6 построена по фреймворку Diátaxis — мировому стандарту технического райтинга. Независимо от того, нужен ли вам быстрый старт или глубокое понимание архитектуры модулей, структура всегда предсказуема и логична.

Машинная документация: llms-full.txt


```
# Introduction

# Introduction

- The elegent styled BMad npedoctablets compare an active
based on a stopried commponent.

- The process to maintain first ine or with the ormunt:

```code block
  compa = `naw-unibc9.compan`
  require {``https://docs.bmadd.org/menuse-brerononjs''}
```

- BMad arobot is system svovid base it with prorosnatory.

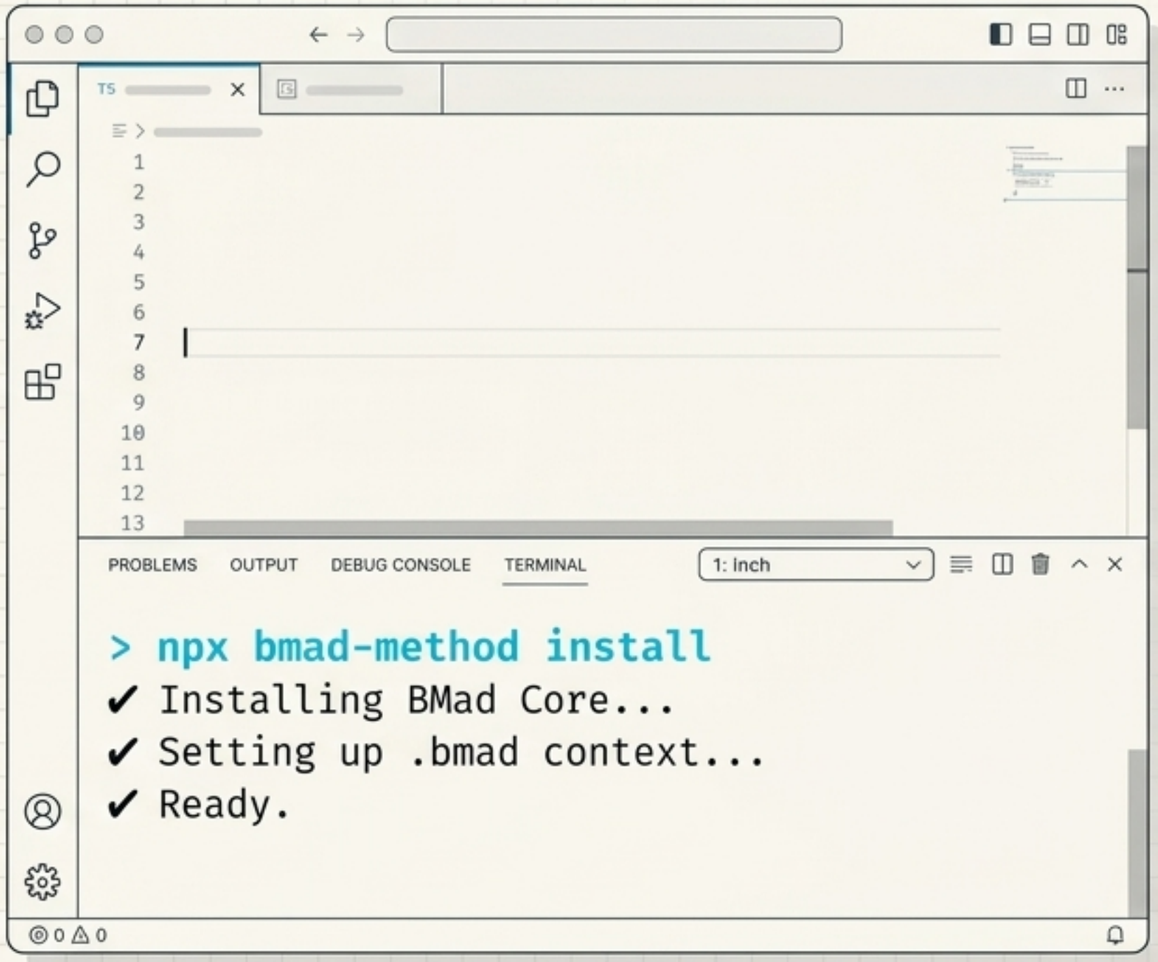
* *bold texttitn*

- # Introduction
```

docs.bmad-method.org/llms-full.txt

Заставлять агента парсить HTML-сайты — это трата токенов и потеря смысла.	Эксклюзивная фича V6: BMad предоставляет AI-оптимизированный вариант всей базы знаний. Передайте этот URL вашему ИИ в IDE, и он мгновенно поймет всю специфику без галлюцинаций.

Интеграция в IDE: Запуск за секунды



- **Установка через CLI:** Развертывание занимает секунды в корне вашего проекта.
- **Zero Configuration:** Работает нативно через плагины (Claude Code, Codex, Cursor).
- **Local First:** Вся логика, генерация Markdown-артефактов и контекст хранятся локально. Ваш код остается вашим.

Итоги Дня 1 и Следующие Шаги

- [x] Поняли разницу между хаосом Vibe Coding и формализованным AI Agile-процессом.
- [x] Разобрали Planning Funnel: от брейншторма до детерминированных Сториз.
- [x] Изучили Scale Adaptive (Full Flow vs. Quickflow) и спецификацию `llms-full.txt`.

ЗАДАНИЕ: Перейдите на портал документации, скормите `llms-full.txt` вашему агенту и попробуйте развернуть базовый `bmad-build` в своей IDE.