

Чек-лист для команды: как снизить риск отказа Codex в рабочем процессе

Этот чек-лист стоит вынести в [downloads/codex-agentic-workflow-checklist.pdf](#). Его можно использовать на ретро, при внедрении Codex или перед переносом агентных задач в регулярный процесс разработки.

Где Codex уже используется

- Зафиксировать задачи, которые разработчики реально делегируют Codex.
- Отдельно отметить ревью, миграции, подготовку diff'ов, исправления тестов и long-context анализ.
- Понять, используется ли Codex формально в процессе или только как личный инструмент отдельных инженеров.

Fallback при недоступности desktop app или CLI

- Описать, как продолжить задачу через обычный Git workflow.
- Проверить, что изменения можно забрать, посмотреть diff и прогнать тесты без конкретного AI-клиента.
- Назначить ответственного за ручное завершение срочных релизных задач при сбое Codex.

Локальная среда и Windows-сценарии

- Проверить обновление CLI в рабочих Windows/PowerShell-окружениях.
- Зафиксировать версию CLI и тип shell, если проблема повторяется.
- Не считать ошибку локальной оболочки проблемой модели без отдельной проверки.

Remote SSH, subagents и thread'ы

- Фиксировать, какой thread или subagent отвечает за конкретную задачу.
- Проверять признаки дублирования работы между main thread и subagent thread.
- Не принимать изменения агента без понимания, из какого контекста они пришли.

Long-context задачи

- Дробить большие ревью и отчеты на этапы: структура, модули, diff, тесты.
- Не передавать большие задачи как непрозрачный черный ящик без промежуточной проверки.
- Сохранять промежуточные результаты так, чтобы человек мог продолжить работу вручную.