Код ревью

Если ты разработчик, переходи по [ссылке], чтобы подробнее изучить инфраструктуру департамента разработки и узнать много новой полезной информации! А пока изучи необходимые теги для код ревью.

Теги комментариев

В рамках эксперимента все комментарии которые оставляются в процессе код ревью, помечаются тегами, чтобы более четко донести суть и назначение комментария и показать что это не просто придирка. Ниже перечислены используемые теги и их краткое описание.

Пример того, как это работает в действии можно посмотреть в одном из недавних MR: [ссылка]

Вопрос

Отмечается местах, где ревьюверу непонятно, почему используется выбранное решение, либо если ревьювер хочет убедиться, что автор кода продумал все нюансы.

Обычно не требует исправления в коде до принятия совместного решения ревьювера и автора кода о дальнейших действиях.

FYI

Комментарий с информацией призванной задуматься и учесть её в будущем.

Обычно не требует изменений в коде.

Ошибка

Отмечается в местах, где скорее всего есть техническая ошибка, приводящая к некорректному поведению кода. Т.е. код делает не то, что от него ожидается.

Обычно является критическим блокером для принятия MR.

Бизнес-логика

Отмечается в местах, где вероятно есть ошибка связанная с бизнес-логикой проекта, т.е. когда код корректный технически, но его поведение не соответствует исходным требованиям.

Нейминг

Отмечается в местах, где выбрано неудачное названия для кодовых абстракций (переменных, функций, классов и т.п.). Обычно даются рекомендации по выбору более удачного названия.

Оптимизация

Отмечается в местах, где можно легко сэкономить в производительности (в том числе в размере бандла), а также в местах, которые могут вызывать проблемы с производительностью.

Архитектура

Отмечается в местах, где код рекомендуется переструктурировать для следования архитектурным принципам.

Поддерживаемость

Отмечается в местах, где в текущем виде код может быть неудобно поддерживать через несколько месяцев при росте и развитии проекта. Обычно предлагается способ улучшения поддерживаемости.

Надежность

Отмечается в местах, где текущий код может падать при определенных обстоятельствах. Предлагаются изменения, помогающие избежать проблему.

Упрощение

Отмечается в местах, где код переусложнен и можно сделать проще, что приведет к улучшению читаемости и понятности кода.

Переиспользование

Отмечается в местах, где можно уменьшить копипасту, чтобы более четко обозначить зависимости между абстракциями внутри кода.

Унификация

Отмечается в местах, где примененное решение расходится с принятыми в проекте или проскоме способами. Унификация в том числе повышает понятность и поддерживаемость кода в долгосрочной перспективе.

Лишний код

Отмечается в местах, где в проекте остался лишний код, который надо убрать, чтобы не путаться в будущем.

Типизация

Отмечается в местах, где некорректно используются TypeScript типы.

Дизайн

Отмечается в местах фронтенда, где реализация не соответствует дизайну. В случае непонятных ситуаций предлагается пообщаться с дизайнером.

UX

Отмечается в местах фронтенда, где можно улучшить UX, либо где нарушены общие UX принципы, принятые в проекте или проскоме. Даются рекомендации по улучшению UX и предлагается пообщаться с дизайнером или аналитиком продукта.

Бекенд

Отмечается в местах фронтенда, где взаимодействие с бекендом не удовлетворяет требованиям бекенда. Обычно в этом случае надо перечитать документацию к бекенду и возможно пообщаться с бекендерами.