Московский Авиационный Институт

Институт №3

# «Системы управления, информатика и электроэнергетика»

Кафедра №304.

«Вычислительные машины, системы и сети»



Отчет по практической работе №3

«Руководство по стилю и code review»

по учебной дисциплине «Технология разработки программного обеспечения»

Вариант №1

Выполнил:

Благушин Н.А.

Группа: М3О-107СВ-24

Принял:

Титов Ю. П.

Москва, 2024 г

**Этап 1. Руководство по стилю написания кода**

Вариант №1

**Руководство по стилю**

1. Определение стандартов и принципов стиля кодирования

Именование переменных, функций и классов

* Переменные и функции: Использовать camelCase (например, userData, calculateTotal).
* Классы и компоненты: Использовать PascalCase (например, UserProfile, OrderSummary).
* Константы: Использовать UPPER\_SNAKE\_CASE (например, MAX\_LIMIT, DEFAULT\_TIMEOUT).

Организация файлов и папок

* Файлы и директории: Именовать файлы в camelCase (например, userProfile.tsx, orderSummary.tsx).
* Структура проекта: Разделять функциональные модули по папкам /components, /hooks, /utils, /styles, /config.
* Один файл — один компонент: Один React-компонент или модуль на файл.

Форматирование кода

* Отступы: 2 пробела для отступов.
* Длина строки: Максимальная длина строки — 80 или 120 символов.
* Блоки кода: Разделять логические блоки пустыми строками.

Комментарии

* Краткие и понятные: Использовать комментарии для пояснения сложной логики.
* Многострочные комментарии: Использовать /\*\* ... \*/, для однострочных — // ....

2. Применение логических операторов и условий для улучшения читаемости кода

Использование явных сравнений

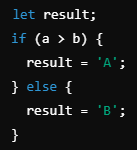
* Плохо: if (isActive)
* Хорошо: if (isActive === true)

Минимизировать вложенность условий

* Использовать return, break и continue для раннего выхода из функции.

Применение тернарных операторов

* Использовать для простых условий.
* Плохо:



* Хорошо:



Чистота условий

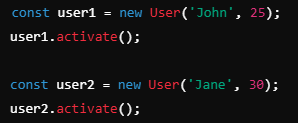
* Плохо: if (user.age > 18 && user.status === 'active')
* Хорошо:



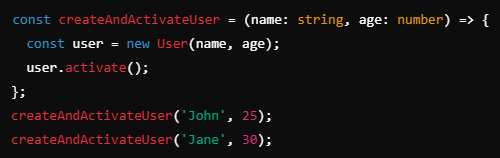
3. Соблюдение принципов DRY (“Don't Repeat Yourself”) и KISS (“Keep It Simple, Stupid”)

Принцип DRY

* Избегать дублирования кода: Вместо повторяющихся блоков кода — выделять функции или модули.
* Плохо:

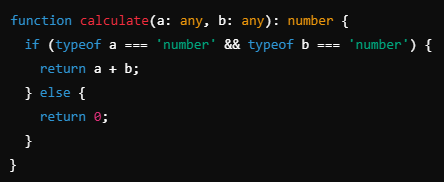


* Хорошо:



Принцип KISS

* Держать код простым и понятным.
* Плохо:



* Хорошо:

const calculate = (a: number, b: number): number => (typeof a === 'number' && typeof b === 'number') ? a + b : 0;

4. Работа с базами данных и обработка запросов к ним

SQL-запросы

* Использовать параметризованные запросы (чтобы избежать SQL-инъекций).
* Плохо: SELECT \* FROM users WHERE id = ${userId}
* Хорошо: SELECT \* FROM users WHERE id = ?

Оптимизация запросов

* Избегать выборки всех столбцов (SELECT \*).
* Использовать индексы на часто используемых столбцах.

Обработка ошибок

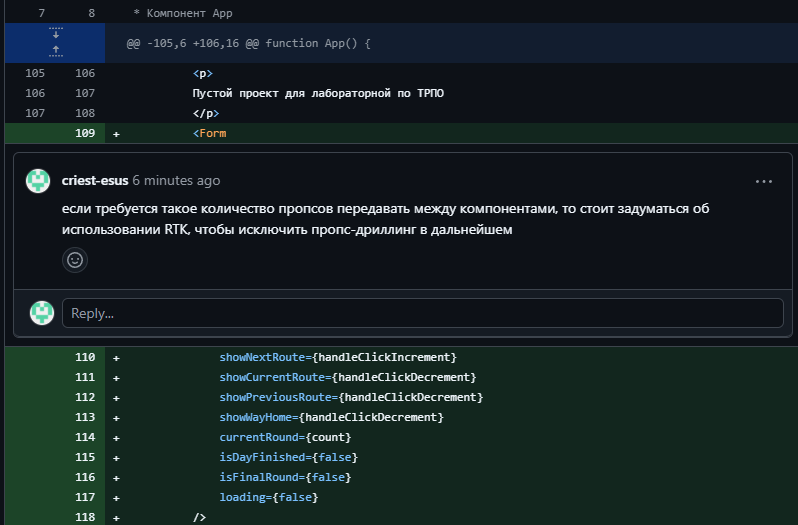
* Ловить ошибки с помощью try...catch при работе с базами данных.

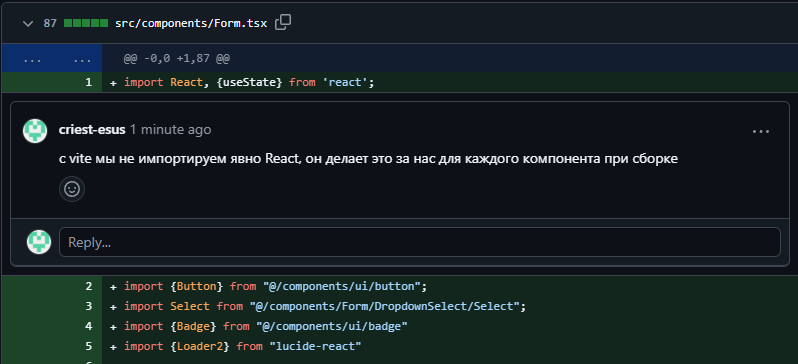
Логирование запросов

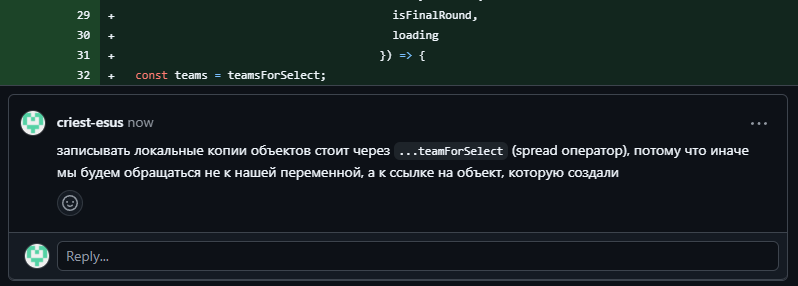
* Логировать критические запросы и ошибки для отладки

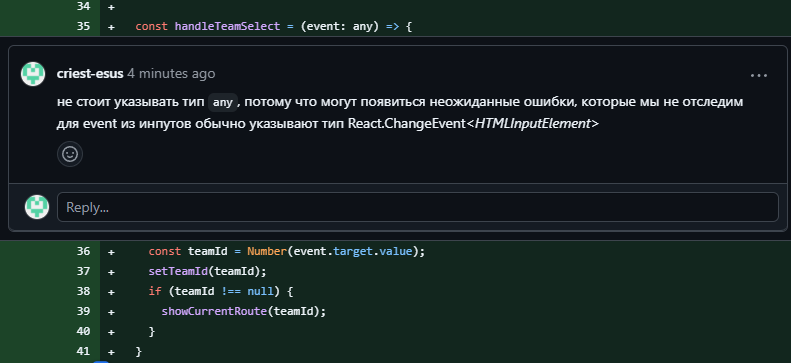
**Этап 2. Code review**

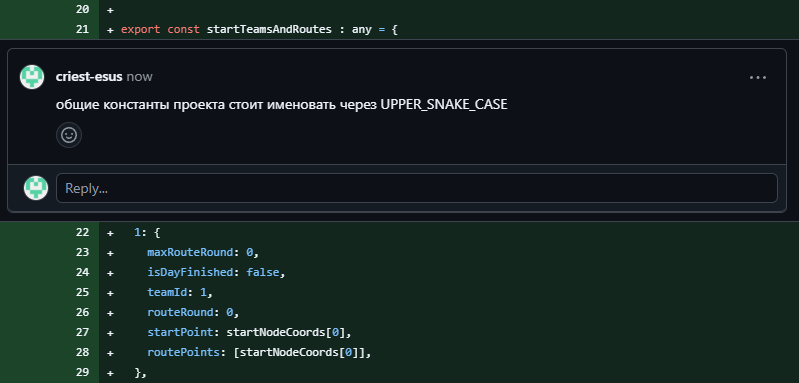
Фронтенд проект лабораторной по ТРПО

****

****

****

****

****