**Правила коммитов**

Алгоритм действий для связывания коммитов с задачами в Jira:

1. Подготовка к работе:

* Открыть задачу в Jira
* Скопировать тег задачи (например, KAN-11)
* Убедиться, что локальный репозиторий актуален:

git fetch

git pull origin main

1. Создание ветки:

* Определить тип изменений (feature или bugfix)
* Создать ветку по формату:

Для новой функциональности: feature/(<TASK-ID>)<что изменили>

Для исправления ошибок: bugfix/(<TASK-ID>)<что изменили>

* Пример создания ветки:

# Пример для новой функциональности:

git checkout -b "feature/(KAN-11)add-project-structure"

# Пример для исправления ошибки:

git checkout -b "bugfix/(KAN-11)fix-readme"  
После чего мы окажемся в новой, созданной ветке. Проверить можно этой командой:  
git branch

* НЕ ЗАБЫВАЕМ:

В Git имена веток должны соответствовать определенным правилам:

* + - Не должны содержать пробелы
    - Не должны содержать специальные символы (кроме -, \_, /, .)

1. Работа с изменениями:

* Внести необходимые изменения
* Добавить изменения в индекс, например:

git add .

* Создать коммит по формату conventional commits, ознакомьтесь по ссылке:  
  <https://www.conventionalcommits.org/ru/v1.0.0/>
* Пример:

# Типы: feat, fix, docs, style, refactor, test, chore

# Формат: <type>(TASK-ID): <description>

git commit -m "feat(KAN-11): add project structure"

1. Отправка изменений:

* Если вы создали ветку в удаленном репозитории, то:

git push  
Может выдать ошибку, которая описана ниже ⬇️

* Если вы не создали ветку в удаленном репозитории, то:

git push --set-upstream origin "feature/(KAN-11)project-structure"

# Или

git push -u origin "feature/(KAN-11)project-structure"

Вообще, если возникла ошибка, после git push, то рекомендую прочитать содержимое ошибки, зачастую, там указано, что следует сделать.

1. Создание Pull Request:

* Перейти в GitHub
* Создать новый Pull Request
* В заголовке указать: [KAN-11] Add project structure
* В описании:
  + Добавить ссылку на задачу в Jira, например:

Jira: <https://manukovskiy.atlassian.net/browse/KAN-11>

* + Описать внесенные изменения(если необходимо)
  + Добавить инструкции по тестированию (если необходимо)
* По желанию можно изменить Reviewers

1. Ожидание ревью:

* Дождаться ревью от команды или замерджить самостоятельно
* Внести правки по комментариям (если необходимо)
* При внесении правок использовать тот же формат коммитов:

git commit -m "fix(KAN-11): fix structure"

1. Мерж изменений:

* После одобрения ревью использовать squash merge или rebase

Разница между squash и rebase:

Squash Merge:

* + Объединяет все коммиты из feature-ветки в один коммит
  + Создает новый коммит в целевой ветке (например, main)
  + История становится более чистой и линейной
  + Все промежуточные коммиты теряются

Rebase:

* + Переносит все коммиты из feature-ветки на вершину целевой ветки
  + Сохраняет все коммиты, но меняет их хеши
  + История становится линейной
  + Все коммиты сохраняются, но с новыми хешами
* Проверить, что изменения корректно отображаются в Jira