1. Общие принципы

- Используем **GitHub Flow** с feature-ветками.
- Вся работа ведется **через пулл-реквесты (PR)**, коммиты в **main** запрещены.
- Описываем изменения в **PR и коммитах** четко и понятно.
- Проводим **код-ревью** перед мержем в main.
- Все задачи фиксируются в **GitHub Issues**.
- Работаем с **GitHub Projects** для управления процессом.

---

2. Ветка разработки

- Основная ветка: main (только стабильный код).
- Ветка разработки: develop (для интеграции изменений).
- Ветки фич: feature/{название-фичи}.
- Ветки багфиксов: fix/{название-багфикса}.
- Ветки релизов: release/{версия} (по необходимости).

---

3. Работа с ветками

Создание ветки

От `develop`

git checkout develop

git pull origin develop

git checkout -b feature/new-feature

Перед началом работы

git pull origin develop # Обновляем ветку перед началом работы

Перед созданием PR

git rebase develop # Подтягиваем изменения из develop

git push origin feature/new-feature # Отправляем ветку в удаленный репозиторий

---

4. Пулл-реквесты (PR)

- PR создается **только в develop**, если не указано иное.
- Название PR: [Feature] Добавлена новая механика игры.
- Описание:
  -
  - Что сделано.
  - Какие файлы изменены.
  - Как протестировать.
- Минимум **1-2 апрува** перед мержем.
- После мержа **ветку удаляем**.

---

5. Код-ревью

- Код-ревью проводят **разработчики** (1 разработчик).
- Проверяем:
  -
  - Чистоту кода (code style).
  - Логичность и соответствие требованиям.
  - Производительность и безопасность.
- Оставляем **конструктивные комментарии**, а не просто «перепиши».

---

6. Коммиты

Формат сообщений

git commit -m "[Номер задачи в треке] Добавлена система очков"

Примеры коммитов

---

7. Конфликты и работа в команде

- Перед коммитом **обновляем свою ветку**: git pull --rebase origin develop.
- Конфликты решаем **локально** перед отправкой PR.
- Если ветка устарела – **ребейзим, а не мержим!**

git fetch origin

git rebase origin/develop

---

8. CI/CD и автоматизация

- **GitHub Actions** для линтеров, тестов и деплоя.
- PR не мержится, если тесты **не пройдены**.
- Линтеры и форматирование проверяются **автоматически** (например, ESLint, Prettier, Black).

---

9. Документация и ведение репозитория

- **README.md** содержит:
  -
  - Описание проекта.
  - Инструкции по развертыванию.
  - Примеры API.
- **CHANGELOG.md** ведем для фиксации изменений.
- **CONTRIBUTING.md** – правила для новых разработчиков.

---

10. Что делать в случае форс-мажора?

- **Случайно закоммитили в main** – git revert.
- **Удалили нужную ветку** – git reflog поможет восстановить.
- **Забыли сделать PR, а уже много изменений** – разбиваем на несколько маленьких.
- **Большие изменения** – обсуждаем перед началом работы