Лабораторная работа “Технический долг”

- Понятный и легко читабельный код;
- Нет дублирующегося кода;
- Отсутствует автоматизация, нет тестов, нет сборки, нет развертывания;
- Простая архитектура с отсутствием сложных зависимостей;
- Присутствуют медленные средства (AudioSource заметно замедляет переход между сценами);
- Репозиторий с проектом в состоянии app-to-date;
- Если считать UserStory технической документацией, то проект соответствует описанию;
- В Unity присутствует NUnit.Framework;
- У нас не используется слияние рабочих копий в общую основную ветвь разработки несколько раз в день и выполнение частых автоматизированных сборок проекта для скорейшего выявления и решения интеграционных проблем.

План:

1. Использовать NUnit.Framework для написания тестов, чтобы автоматизировать процесс тестирования.
2. Чаще использовать слияние рабочих копий в общую основную ветвь.
3. Автоматизировать сборку проекта в среде Unity.

Оценка плана:

План значительно ускорит разработку и поддержку проекта в дальнейшем, потому что он поможет выявить ситуации, в которых поведение программы является неправильным, нежелательным или не соответствующим спецификации в сжатых сроках.

Сравнение объёма долга и недоимплементированных фич:

Все задачи выполняются так, как запланировано на спринте, кроме задачи: “Как обычный пользователь, я хочу автоматического приостановления игры при потере одной из меток, чтобы мне не засчитывали это проигрышем”.

Вывод:

Технический долг приводит к проблемам у пользователей, которые постепенно прекращают пользоваться продуктом. А игнорирование технического долга как разработчиками, так и менеджерами только способствует его накоплению.