*Программист .Net 4/6, C#*

**Тестовое задание.**

Windows 10/11, .Net 6. Сетевое приложение с двух- или трёхзвенной архитектурой.

**Сервер** генерирует поток данных. Данные – координаты прямоугольников. Координаты – случайные величины на старте сервера. Координаты генерируются множеством потоков: пусть это будет конечное случайное количество потоков. Количество прямоугольников – случайная величина в рамках заданной константы в одном потоке. *Можно задать константное количество потоков и константное количество прямоугольников в потоке.*

Размеры прямоугольников могут быть фиксированными.

В процессе работы координаты прямоугольников плавно меняются от стартовых в случайную сторону в рамках заданных констант ( то же, что и bouncing balls: нечто постоянно «летающее» от края до края экрана по диагоналям).

Координаты складываются потоками в единый массив (выбор вида массива – за разработчиком). Координаты прямоугольника – набор данных из собственно координат, размеров и, возможно, идентификатора прямоугольника.

Координаты передаются клиенту **отдельным потоком** непрерывно.

**Клиент** принимает данные и отображает их в окне в виде плавающих прямоугольников. Прямоугольник создаётся на клиенте, если такого прямоугольника ещё нет у клиента.

Пусть количество прямоугольников варьируется от 100 до 1000. Пусть «рамки» поля будут фиксированными. Пусть количество потоков, формирующих данные на сервере, будет фиксированным.

Для сетевого взаимодействия желательно использовать gRPC + Protobuf, но можно применить и свой протокол на базе UDP. Не нужно использовать WCF.

На клиенте:

- максимально использовать MVVM (стараться не использовать PRISM и т.п.). Code-behind не допускается.

На сервере:

- связь прямоугольника в контейнере-хранилище реализовать через идентификатор [потока-обработчика];

- не использовать volatile-массивы для хранения прямоугольников: вместо этого, показать использование примитивов синхронизации для реализации условий задания.

Результат в виде решения MS VS 2019/2022 залить на github (приватный или нет – за разработчиком). Репозиторий должен содержать простой ReadMe с указаниями по сборке.

Решение должно содержать проекты сервера и клиента. В случае использования gRPC – отдельный проект со структурами Protobuf.