## ✅ Тестовое задание: TCP+TLS Клиент и Сервер на C#

### Общие требования

* Язык реализации: **C#**
* Предпочтительно использовать **.NET 9**
* Решение должно **собираться и работать на Linux (x86\_64)** системах
* Сборка приложения должна использовать **AOT-компиляцию** (Ahead-of-Time), быть **Self-contained** (самодостаточной, без зависимости от установленного .NET на целевой машине) и упакована в **один исполняемый файл** (**Single-file executable**).
* Пример команды сборки:

dotnet publish -r linux-x64 -c Release -p:PublishAot=true -p:SelfContained=true -p:PublishSingleFile=true

* Разрешается использование **любых сторонних библиотек**, совместимых с требованиями выше
* Не требуется полноценный production-ready код — делаем **прототип**, но с рабочей логикой
* **Unit-тесты** писать **не обязательно**
* Исходный код необходимо разместить на одном из общедоступных репозиториев:
  + [github.com](https://github.com), [gitlab.com](https://gitlab.com), [bitbucket.org](https://bitbucket.org) или аналогичных

## 🔹 Задание 1. CLI клиент TCP+TLS

Создать утилиту командной строки, которая:

* Принимает путь к конфигурационному **JSON-файлу** как аргумент командной строки
* Загружает конфигурацию из JSON:

{  
 "InputPath": "/var/data.json",  
 "ServerHost": "localhost",  
 "Server\_port": 5555,  
 "TlsEnabled": true,  
 "TlsValidateCert": false  
}

* Открывает TCP-соединение (при tls\_enabled: true — через TLS)
* Построчно читает файл, указанный в InputPath (строка заканчивается на ‘’)
* Каждую строку проверяет на валидность как **JSON**
* Каждую валидную строку **отправляет на сервер** в рамках **одной TCP/TLS-сессии с множественными сообщениями**
* Используем \n в качестве разделителя
* Невалидные строки — логировать в stderr
* Закрываем соединение по прочтению всего файла

Пример JSON файла:

{"type":"event","level":"info","message":"Service started"}  
{"type":"event","level":"debug","message":"Debugging something"}  
not a json line  
{"type":"event","level":"error","message":"Something went wrong"}  
{"type":"log","message":"Missing field 'level'}  
1234  
{"type":"event","level":"info","message":"All systems operational"}

## 🔸 Задание 2. TCP+TLS сервер

Создать серверное приложение, которое:

* Загружает JSON-конфигурацию следующего вида:

{  
 "ListenAddr": "0.0.0.0",  
 "ListenPort": 5555,  
 "TlsEnabled": true,  
 "TlsCrt": "/etc/some/path.crt",  
 "TlsKey": "/etc/some/path.key",  
 "Filters": [  
 {  
 "field": "type",  
 "operator": "equals",  
 "value": "event"  
 },  
 {  
 "field": "level",  
 "operator": "not\_equals",  
 "value": "debug"  
 }  
 ]  
}

* Слушает указанный TCP-порт (через TLS — если включено)
* Обрабатывает **несколько одновременных подключений** от клиентов
* От клиента приходят строки с JSON (по одной на строку)
* Каждую строку:
  + Проверяет на корректность как JSON
  + Применяет к ней список фильтров:
    - field — имя поля
    - operator — возможные значения: equals, not\_equals, contains, not\_contains
    - value — ожидаемое значение
  + Если сообщение **проходит все фильтры** — выводит его в консоль в **pretty JSON** формате
  + Иначе — игнорирует
* Необходимо создать **SystemD unit файл** для запуска сервера как systemd-сервиса