Description

This project with implement examples with Retry and Circuit Breaker resilience patterns using the Polly lib.

Input Client Configuration

```
{
    "UrlConfiguration": {
        "BaseUrl": string,
        "Action": string,
        "Method": "GET/POST/PUT"
    },
    "RequestConfiguration": {
        "SuccessRequests": int,
        "MaxRequests": int,
        "Delay": int,
        "ProbabilityError": int,
        "Timeout": int
    },
    "RunPolicy": "NONE/RETRY/CIRCUIT_BREAKER",
    "RetryConfiguration": {
        "Count": int,
        "SleepDuration": int,
        "SleepDurationType": "FIXED/EXPONENTIAL_BACKOFF"
    },
    "CircuitBreakerConfiguration": {
        "IsSimpleConfiguration": boolean,
        "DurationOfBreaking": int, // (milisecond) duração que o circuito irá ficar aberto.
        "ExceptionsAllowedBeforeBreaking": int, // (usado somente quando for Simples)
quantidade de exceções consecutivas para o circuito ficar aberto.
        "FailureThreshold": double, // (porcentagem) O limite de falha no qual o circuito
será interrompido (um número entre 0 e 1; por exemplo, 0,5 representa a interrupção se 50%
ou mais das ações resultarem em uma falha controlada.
        "SamplingDuration": int, // (milisecond) A duração do tempo sobre o qual as taxas
de falha são avaliadas.
        "MinimumThroughput": int // Número mínimo de requisições no intervalo de tempo
descrito pelo SamplingDuration.
    }
}
```

Result Client Configuration

```
{
    "TotalTime": "00h:00m:00s:000ms",
    "ClientToModule": {
        "Success": int,
        "Error": int,
        "Total": int
```

```
    },
    "ResilienceModuleToExternalService": {
        "Success": int,
        "Error": int,
        "Total": int
    },
    "RetryMetrics": {
        "RetryCount": int,
        "TotalTimeout": "00h:00m:00s:000ms"
    },
    "CircuitBreakerMetrics": {
        "BreakCount": int,
        "ResetStatCount": int,
        "TotalOfBreak": "00h:00m:00s:000ms"
    }
}
```

Automatic Scenario Runner Configuration

```
{
    "AutomaticRunnerConfiguration": {
        "ScenariosPath": "D:\\Wokspace\\Universidade\\ResiliencePatternsDotNet\\Scenarios"
    }
}
```

Scenario Configuration

```
{
    "Run": boolean,
    "Count": int,
    "AsyncClients": boolean,
    "UrlFetch": {
        "BaseUrl": string,
        "ActionUrl": string,
        "Method": "GET/POST/PUT"
    },
    "ResultType": "TXT/CSV",
    "ProxyConfiguration": {
        "DockerComposePath": string,
        "VaurienConfigPath": string,
        "RestartVaurienContainerCommand": "docker-compose restart vaurien", // 'vaurien' é
o nome do container dentro do meu docker-compose
        "Behavior": "0:blackout,0:delay,0:error,0:hang" // falta implementar os behavior do
delay (caso precise mudar o delay)
    },
    "Parameters": {
        "UrlConfiguration": {
            "BaseUrl": string,
            "Action": string,
            "Method": "GET/POST/PUT"
```

```
        },
        "RequestConfiguration": {
            "SuccessRequests": int,
            "MaxRequests": int,
            "Delay": int,
            "ProbabilityError": int,
            "Timeout": int
        },
        "RunPolicy": "NONE/RETRY/CIRCUIT_BREAKER",
        "RetryConfiguration": {
            "Count": int,
            "SleepDuration": int,
            "SleepDurationType": "FIXED/EXPONENTIAL_BACKOFF"
        },
        "CircuitBreakerConfiguration": {
            "IsSimpleConfiguration": boolean,
            "DurationOfBreaking": int,
            "ExceptionsAllowedBeforeBreaking": int,
            "FailureThreshold": double,
            "SamplingDuration": int,
            "MinimumThroughput": int
        }
    }
}
```