**.NET Request Logger & Rate Limiter**

**Overview**

This project is a **.NET Core Web API** extension that provides **request logging** and **rate limiting** capabilities. It logs incoming requests, including method, IP address, response time, and status, while blocking excessive requests from the same IP to prevent server overload.

**Features**

* **Request Logging**: Logs HTTP requests with details (method, IP, duration, status code).
* **Rate Limiting**: Blocks requests exceeding a configured threshold within a time window.
* **Configuration via appsettings.json**: Allows customizing limits and logging levels.
* **Swagger Integration**: API testing via Swagger UI.

**Technologies Used**

* .NET 8
* ASP.NET Core Web API
* C#
* Middleware Architecture
* Logging (Microsoft.Extensions.Logging)

**Setup Instructions**

**1. Clone the Repository**

git clone <https://github.com/nichifor-dragos14/framework-design-assignment.git>

cd dotnet-request-limiter-and-logger

**2. Install Dependencies**

Ensure you have the .NET SDK installed. If not, download it from [Microsoft .NET](https://dotnet.microsoft.com/en-us/).

**3. Run the Application**

dotnet run

The API will start at <https://localhost:7209> (or the default assigned port).

**Configuration**

Modify appsettings.json to adjust rate limiting settings:

{

"RateLimiting": {

"RequestLimit": 3,

"TimeWindowSeconds": 1

},

"Logging": {

"LogLevel": {

"Default": "Information"

}

}

}

* RequestLimit: Maximum number of requests per IP in the given time window.
* TimeWindowSeconds: Time window for rate limiting (in seconds).

**Middleware Implementation**

1. **Logging Middleware**

Captures and logs incoming request details, such as the HTTP method, path, IP address, response status, and duration.

If a request is blocked due to exceeding the rate limit, it logs an error indicating the blocked IP.

2. **Rate Limiting Middleware**

Monitors and restricts the number of requests per IP address based on configuration settings.

Maintains a record of recent requests per IP and blocks further requests if the limit is exceeded within the specified time frame.

Returns an HTTP 429 (Too Many Requests) status when a request is blocked, along with a message indicating the rate limit breach.

**Usage & Testing**

**Test with Swagger UI**

Run the API and navigate to https://localhost:7209/swagger to test endpoints.

**Send API Requests**

Test the rate limiting by sending multiple requests:

curl -X GET "https://localhost:7209/api/test" -H "Accept: application/json"

If you exceed the limit, you'll receive:

[Rate Limit] IP ::1 blocked. Too many requests

**Conclusion**

This project provides a robust logging and request-limiting middleware that helps secure APIs from abuse and ensures performance stability. 🚀

For further improvements, consider:

* Implementing dynamic configuration changes without restarting the app.
* Logging request details into a database for analytics.
* Adding JWT-based authentication to prevent abuse from unauthorized users.