

RATE LIMITING

WITH

MINIMAL APIS

@romain-od 
www.code-review.tech



FIXED WINDOW LIMITER

Let's define limits such as “60 requests per minute”. Every minute, 60 requests can be made. One every second, but also 60 in one go.



```
builder.Services.AddRateLimiter(rateLimiterOptions =>
{
    rateLimiterOptions.AddFixedWindowLimiter("fixed", options =>
    {
        options.PermitLimit = 60;
        options.Window = TimeSpan.FromMinutes(1);
        options.QueueProcessingOrder = QueueProcessingOrder.OldestFirst;
        options.QueueLimit = 5;
    });
});
```

@romain-od



www.code-review.tech



SLIDING WINDOW LIMITER

Similar to the fixed window limit, but uses segments for more fine-grained limits. Think “60 requests per minute, with 1 request per second”



```
builder.Services.AddRateLimiter(rateLimiterOptions =>
{
    rateLimiterOptions.AddSlidingWindowLimiter("sliding", options =>
    {
        options.PermitLimit = 60;
        options.Window = TimeSpan.FromMinutes(1);
        options.SegmentsPerWindow = 60;
        options.QueueProcessingOrder = QueueProcessingOrder.OldestFirst;
        options.QueueLimit = 5;
    });
});
```

@romain-od



www.code-review.tech



TOKEN BUCKET LIMITER

“you are given 100 requests every minute”.

If you make all of them over 10 seconds, you'll have to wait for 1 minute before you are allowed more requests.



```
builder.Services.AddRateLimiter(rateLimiterOptions =>
{
    rateLimiterOptions.AddTokenBucketLimiter("token", options =>
    {
        options.TokenLimit = 100;
        options.QueueProcessingOrder = QueueProcessingOrder.OldestFirst;
        options.QueueLimit = 5;
        options.ReplenishmentPeriod = TimeSpan.FromMinutes(1);
        options.TokensPerPeriod = 10;
        options.AutoReplenishment = true;
    });
});
```

@romain-od



www.code-review.tech



CONCURRENCY LIMITER

Don't look at time, just at number of concurrent requests. "Allow 10 concurrent requests".



```
builder.Services.AddRateLimiter(rateLimiterOptions =>
{
    rateLimiterOptions.AddConcurrencyLimiter("concurrency", options =>
    {
        options.PermitLimit = 10;
        options.QueueProcessingOrder = QueueProcessingOrder.OldestFirst;
        options.QueueLimit = 5;
    });
});
```

@romain-od



www.code-review.tech



QUEUELIMIT

You noticed that I set *QueueLimit*
= 5 in all my example

The QueueLimit specifies how many
incoming requests will be queued but not
rejected when the PermitLimit is
reached.

@romain-od



www.code-review.tech



APPLY TO ENDPOINTGROUP

Now we can apply different
RateLimiting on our endpoint group



```
var items = app.MapGroup("api/items/")
    .GroupItems()
    .RequireRateLimiting("fixed")
    .WithTags("Items");

var users = app.MapGroup("api/users/")
    .GroupUsers()
    .RequireRateLimiting("token")
    .WithTags("Users");
```

@romain-od



www.code-review.tech






THANKS FOR READING
SHARE YOUR THOUGHTS

FOLLOW ME FOR MORE SHARING



@romain-od 

www.code-review.tech