Top 15 Mistakes in API Development

Mistake 1: Not Using API Versioning

```
var builder = WebApplication.CreateBuilder(args);

builder.Services.AddApiVersioning(options ⇒
{
    options.DefaultApiVersion = new ApiVersion(1,0);
    options.AssumeDefaultVersionWhenUnspecified = true;
});

var app = builder.Build();

app.MapGet("/api/v1/products", () ⇒
{
    return new[] { "Product1", "Product2" };
});

app.Run();
```

Anton Martyniuk

Top 15 Mistakes in API Development

Mistake 2: Poor Error Handling

```
var builder = WebApplication.CreateBuilder(args);

builder.Services.AddProblemDetails();

var app = builder.Build();

app.UseExceptionHandler();

app.Run();

return Results.Problem(
  type: "Bad Request",
  title: "Validation Failed",
  detail: "Email should be in the correct format",
  statusCode: StatusCodes.Status400BadRequest);
```

Anton Martyniuk

Top 15 Mistakes in API Development

Mistake 3: Missing Authentication and Authorization

```
var builder = WebApplication.CreateBuilder(args);

builder.Services.AddAuthentication(JwtBearerDefaults.AuthenticationScheme)
    .AddJwtBearer(options =>
    {
        options.TokenValidationParameters = new TokenValidationParameters
        {
            ValidateIssuer = true,
            ValidateAudience = true,
            ValidateIssuerSigningKey = true,
            IssuerSigningKey = new SymmetricSecurityKey(
                Encoding.UTF8.GetBytes("your-secret-key"))
        };
    });

var app = builder.Build();

app.UseAuthentication();
app.UseAuthorization();

app.MapGet("/api/orders", () => new[] { "Order1", "Order2" })
    .RequireAuthorization();

app.Run();
```

Anton Martyniuk

Top 15 Mistakes in API Development

Mistake 4: Ignoring Asynchronous Programming

```
app.MapGet("/api/v1/products", (IProductService productService) ⇒
{
    var products = productService.GetItems();
    return Ok(products);
});

app.MapGet("/api/v1/products", async (IProductService productService) ⇒
{
    var products = await productService.GetItemsAsync();
    return Ok(products);
});
```

Anton Martyniuk

Top 15 Mistakes in API Development

Mistake 5: Not following RESTful conventions

```
// Not following REST: using GET for deletion
app.MapGet("/api/deleteUser?id=123", () ⇒
{
    // Delete logic here
    return Results.Ok("Deleted");
});

// Use proper HTTP method and resource naming
app.MapDelete("/api/users/{id}", (int id) ⇒
{
    // Delete user with the given id
    return Results.NoContent();
});
```

Anton Martyniuk

Top 15 Mistakes in API Development

Mistake 6: Not Validating Input Data

```
app.MapPost("/api/users", (CreateUserRequest request) ⇒
{
    // No validation performed
    return Results.Ok(user);
});

app.MapPost("/api/users", (CreateUserRequest request,
    IValidator<CreateUserRequest> validator) ⇒
{
    var validationResult = await validator.ValidateAsync(request);
    if (!validationResult.IsValid)
    {
        return Results.ValidationProblem(validationResult.ToDictionary());
    }

    return Results.Ok(user);
});
```

Anton Martyniuk

Top 15 Mistakes in API Development

Mistake 7: Ignoring Security Best Practices

```
app.MapGet("/api/product/{id}", (string name) ⇒
{
    // Vulnerable SQL string concatenation
    var command = new SqlCommand(
        "SELECT * FROM Products WHERE Name = " + name, connection);

    connection.Open();
    using var reader = command.ExecuteReader();
    // ...
    return Results.Ok();
});
```

Anton Martyniuk

Top 15 Mistakes in API Development

Mistake 7: Ignoring Security Best Practices

```
// Use safe methods from ORM like EF Core or Dapper
// Or use parameters to prevent SQL injection
app.MapGet("/api/product/{id}", (string name, ProductDbContext dbContext) ⇒
{
    var products = await dbContext.Products
        .Where(x ⇒ x.Name === name)
        .ToListAsync();

    return Results.Ok(products);
});
```

Anton Martyniuk

Top 15 Mistakes in API Development

Mistake 8: Poor Logging and Monitoring

```
builder.Services
    .AddOpenTelemetry()
    .ConfigureResource(resource ⇒ resource.AddService("ShippingService"))
    .WithTracing(tracing ⇒
    {
        tracing
            .AddAspNetCoreInstrumentation()
            .AddHttpClientInstrumentation()
            .AddEntityFrameworkCoreInstrumentation()
            .AddRedisInstrumentation()
            .AddNpgsql();

        tracing.AddOtlpExporter();
    });
```

Anton Martyniuk

Top 15 Mistakes in API Development

Mistake 9: Lack of API Documentation

```
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen(c ⇒
{
    c.SwaggerDoc("v1", new OpenApiInfo { Title = "My API", Version = "v1" });
});

var app = builder.Build();

if (app.Environment.IsDevelopment())
{
    app.UseSwagger();
    app.UseSwaggerUI();
}

app.MapGet("/api/products", () ⇒ new[] { "Product1", "Product2" });

app.Run();
```

Anton Martyniuk

Top 15 Mistakes in API Development

Mistake 10: Not Optimizing Database

```
// Fetching ALL columns
var book = await context.Books
    .Include(b => b.Author)
    .FirstOrDefaultAsync(b => b.Id == id, cancellationToken);

// Fetching only needed columns
var book = await context.Books
    .Where(b => b.Id == id)
    .Select(b => new BooksPreviewResponse
    {
        Title = b.Title, Author = b.Author.Name, Year = b.Year
    })
    .FirstOrDefaultAsync(cancellationToken);
```

Anton Martyniuk

Top 15 Mistakes in API Development

Mistake 11: Returning Too Much Data no Filtering, Sorting, Pagination

```
// Selecting all books (entire database)
var allBooks = await context.Books
    .Include(b => b.Author)
    .ToListAsync();

// Use paging to select fixed number of records
int pageSize = 50;
int pageNumber = 1;

var books = context.Books
    .AsNoTracking()
    .OrderBy(p => p.Title)
    .Skip((pageNumber - 1) * pageSize)
    .Take(pageSize)
    .ToList();
```

Anton Martyniuk

Top 15 Mistakes in API Development

Mistake 12: Not Using Caching

```
builder.Services.AddHybridCache();

[HttpGet("orders/{id}")]
public async Task<IActionResult> GetOrderAsync(int id,
    [FromServices] IHybridCache cache)
{
    string cacheKey = $"Order_{id}";
    var order = await cache.GetOrCreateAsync(cacheKey, async entry ⇒
    {
        entry.AbsoluteExpirationRelativeToNow = TimeSpan.FromMinutes(10);
        using var context = new AppDbContext();
        return await context.Orders.FindAsync(id);
    });

    if (order == null)
        return NotFound();

    return Ok(order);
}
```

Anton Martyniuk

Top 15 Mistakes in API Development

Mistake 13: Returning Big Payloads without Compression

```
builder.Services.AddResponseCompression(options ⇒
{
    options.EnableForHttps = true;
    options.Providers.Add<BrotliCompressionProvider>();
    options.Providers.Add<GzipCompressionProvider>();
});

builder.Services.Configure<BrotliCompressionProviderOptions>(options ⇒
{
    options.Level = System.IO.Compression.CompressionLevel.Fastest;
});

builder.Services.Configure<GzipCompressionProviderOptions>(options ⇒
{
    options.Level = System.IO.Compression.CompressionLevel.Fastest;
});

var app = builder.Build();
app.UseResponseCompression();
```

Anton Martyniuk

Top 15 Mistakes in API Development

Mistake 14: Fat Controllers

```
public class ProductsController(
    IProductRepository productRepository,
    ILoggingService loggingService,
    ICacheService cacheService,
    IEmailService emailService,
    IAuthenticationService authService,
    IReportGenerator reportGenerator,
    IFeatureFlagService featureFlagService
) : ControllerBase
{
    public IActionResult GetAllProducts() { }
    public IActionResult GetProductById(int id) { }
    public IActionResult CreateProduct() { }
    public IActionResult UpdateProduct(int id) { }
    public IActionResult DeleteProduct(int id) { }
    public IActionResult GetProductsByCategory(string category) { }
    public IActionResult ExportProducts()  { }
    public IActionResult SendProductNewsletter()  { }
    public IActionResult GetProductStats()  { }
    public IActionResult GetProductRecommendations() { }
}
```

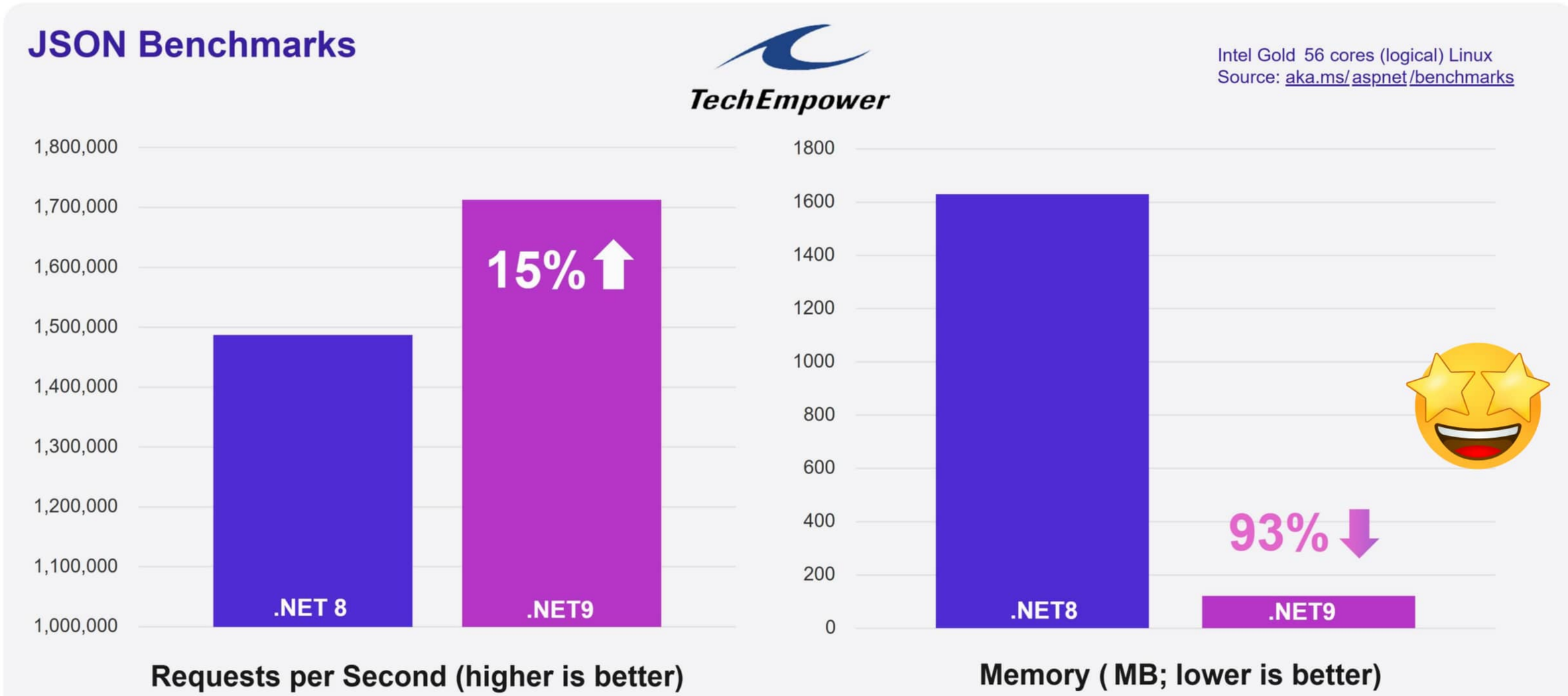
Anton Martyniuk

antondevtips.com

Top 15 Mistakes in API Development

Mistake 15: Ignoring Minimal APIs

.NET 9 Minimal API Performance



Anton Martyniuk

antondevtips.com

THANKS FOR READING

- [x] LETS CONNECT ON LINKED-IN
- [x] REPOST CONTENT TO OTHERS
- [x] SUBSCRIBE TO **ANTONDEVTIPS.COM** TO IMPROVE YOUR SKILLS IN **.NET**

Anton Martyniuk

antondevtips.com

Repost