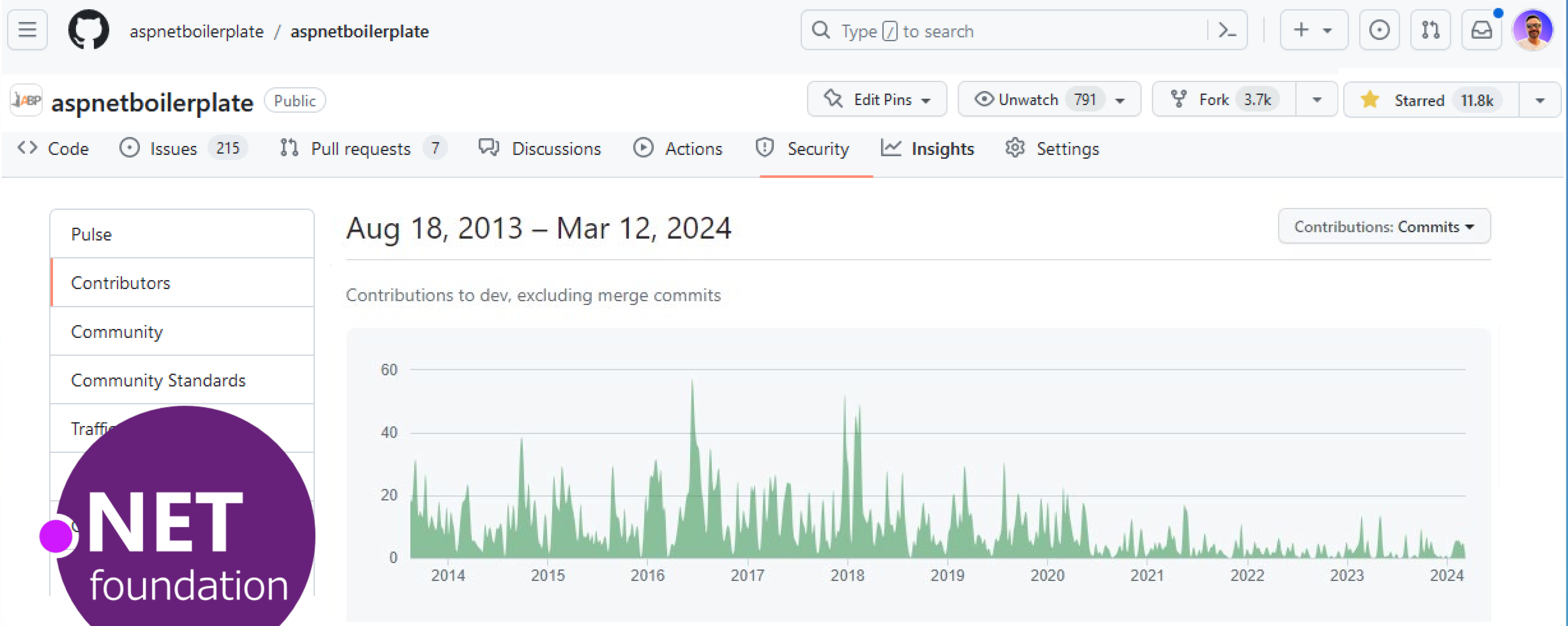
DotNext 2024

Building Multi-Tenant ASP.NET Core Applications

Alper Ebicoglu
Co-Founder of **Volosoft**
alper.ebicoglu@volosoft.com
twitter.com/**alperebicoglu**

abp.io

Open-source Framework on ASP.NET Core



aspnetboilerplate / aspnetboilerplate

aspnetboilerplate Public

Edit Pins · Unwatch 791 · Fork 3.7k · Starred 11.8k

Code · Issues 215 · Pull requests 7 · Discussions · Actions · Security · Insights · Settings

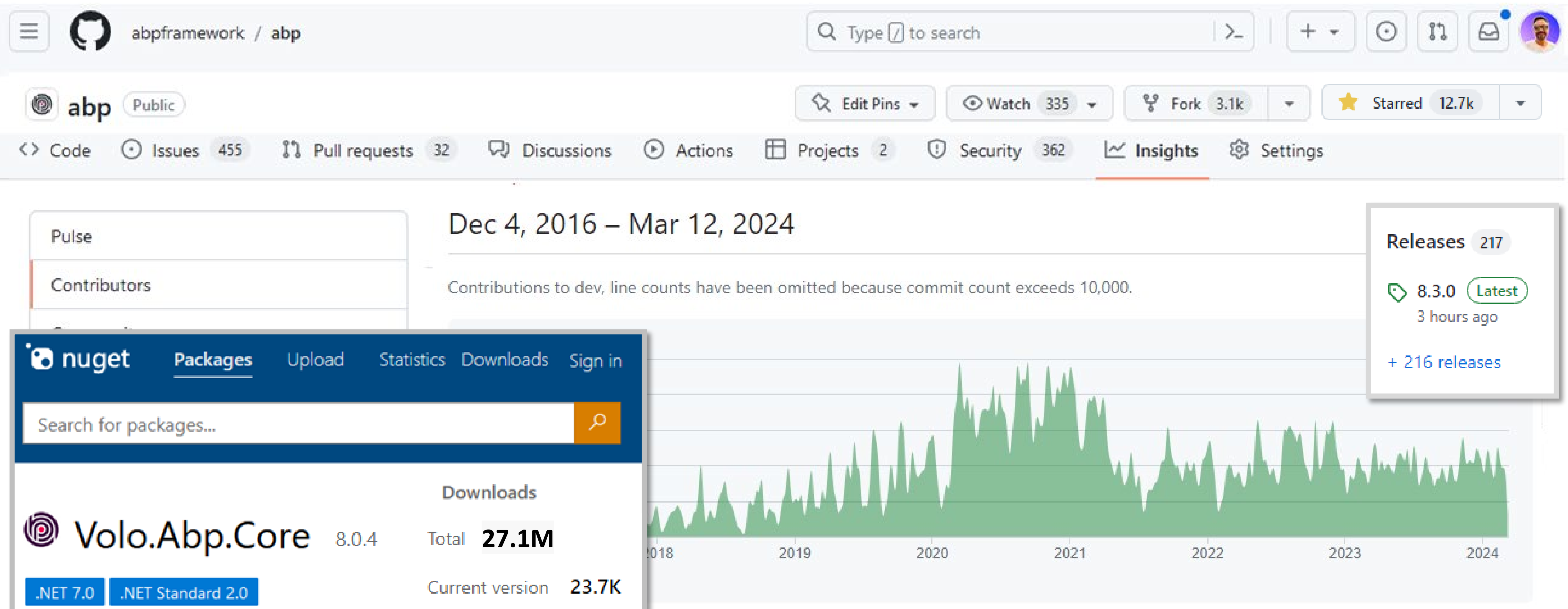
Pulse
Contributors
Community
Community Standards
Traffi

Aug 18, 2013 – Mar 12, 2024

Contributions: Commits

Contributions to dev, excluding merge commits

abp.io | Open-source Framework on ASP.NET Core


abpframework / abp

abp Public

Edit Pins | Watch 335 | Fork 3.1k | Starred 12.7k

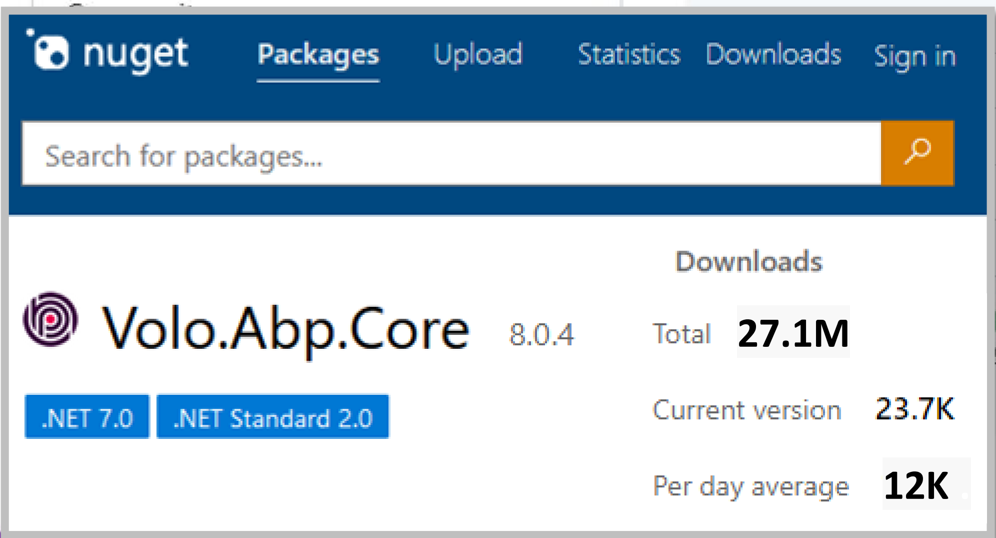
Code | Issues 455 | Pull requests 32 | Discussions | Actions | Projects 2 | Security 362 | Insights | Settings

Pulse
Contributors

Dec 4, 2016 – Mar 12, 2024

Contributions to dev, line counts have been omitted because commit count exceeds 10,000.

Releases 217
8.3.0 Latest
3 hours ago
+ 216 releases


nuget | Packages | Upload | Statistics | Downloads | Sign in

Search for packages...

Volo.Abp.Core 8.0.4

.NET 7.0 | .NET Standard 2.0

Downloads	
Total	27.1M
Current version	23.7K
Per day average	12K

What is **ABP Framework?**

Your Application

Focus on your business code

- do what you do best

ABP Web Framework

An opinionated architecture
to build line-of-business web apps

- Multi-tenancy
- Audit logging
- Exception handling
- Background jobs
- Modularity
- Event bus
- Unit of work
- etc...

ASP.NET Core Web Framework

Generic web framework

- Routing
- Dependency injection
- Session management
- Request / response
- Security
- etc...

Agenda

- **Introduction** to SaaS & Multi-Tenancy
- **Pros and Cons** of Multi-Tenancy
- Database & **Deployment Scenarios**
- **Identifying** and Changing the **Active Tenant**
- **Data Isolation**
- Conditionally Turning **Multi-Tenancy On / Off**
- Handling **Database Migrations**
- Do You **Need Multi-Tenancy**?

What is Multi-Tenancy?

- A common approach to build SaaS solutions
- Resources are shared between tenants
- Application data is isolated between tenants

Parties

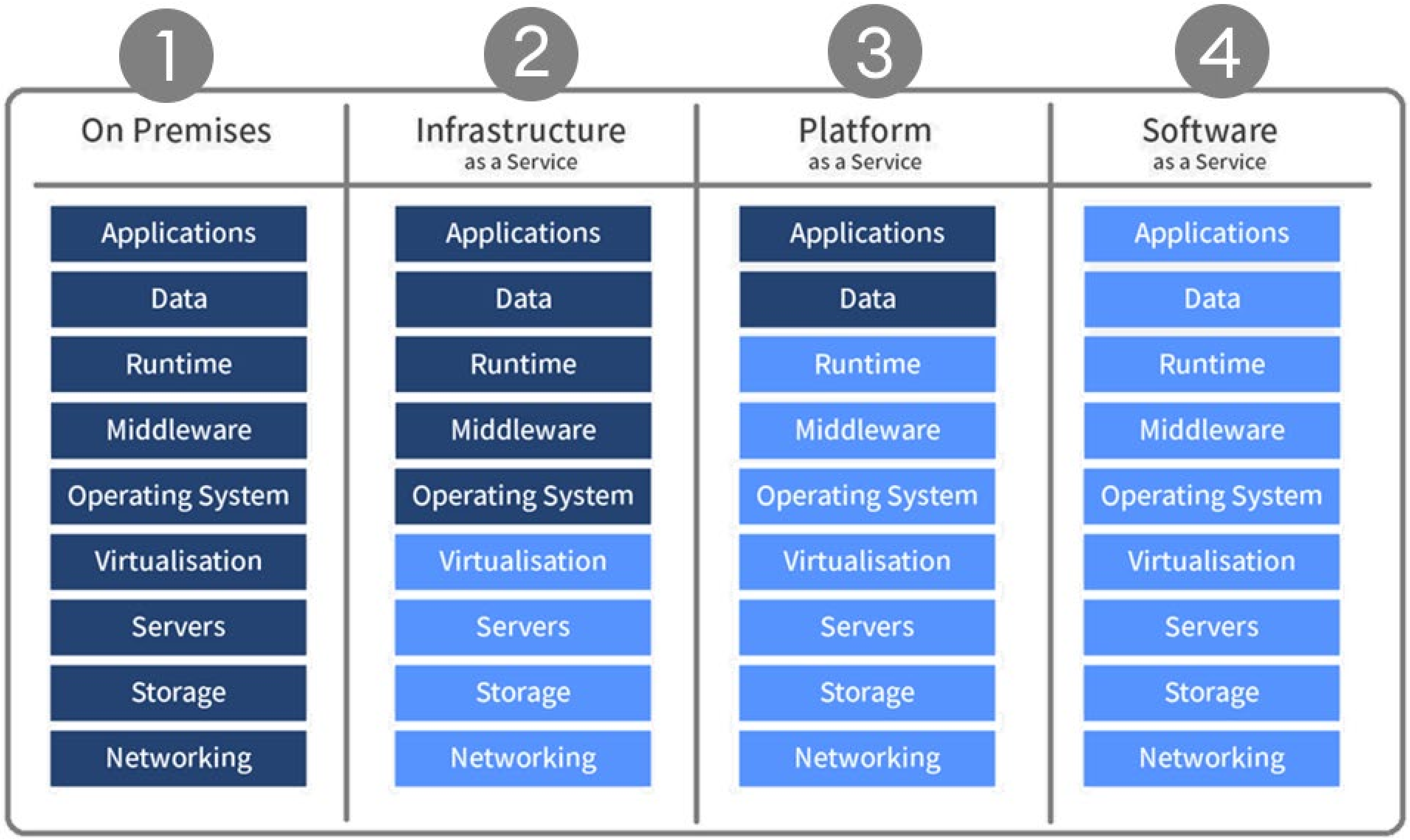
- **Tenants:** Our clients, using the service
- **Host:** Service provider

An ideal multi-tenant application should be

✓ Unaware of multi-tenancy as much as possible!

✓ Deployable to on-premise as well

As-a-Service Business Models



1 On Premises	2 Infrastructure as a Service	3 Platform as a Service	4 Software as a Service
■ Applications	■ Applications	■ Applications	□ Applications
■ Data	■ Data	■ Data	□ Data
■ Runtime	■ Runtime	□ Runtime	□ Runtime
■ Middleware	■ Middleware	□ Middleware	□ Middleware
■ Operating System	■ Operating System	□ Operating System	□ Operating System
■ Virtualisation	□ Virtualisation	□ Virtualisation	□ Virtualisation
■ Servers	□ Servers	□ Servers	□ Servers
■ Storage	□ Storage	□ Storage	□ Storage
■ Networking	□ Networking	□ Networking	□ Networking

■ You manage

□ 3rd Party Manages

abp.io

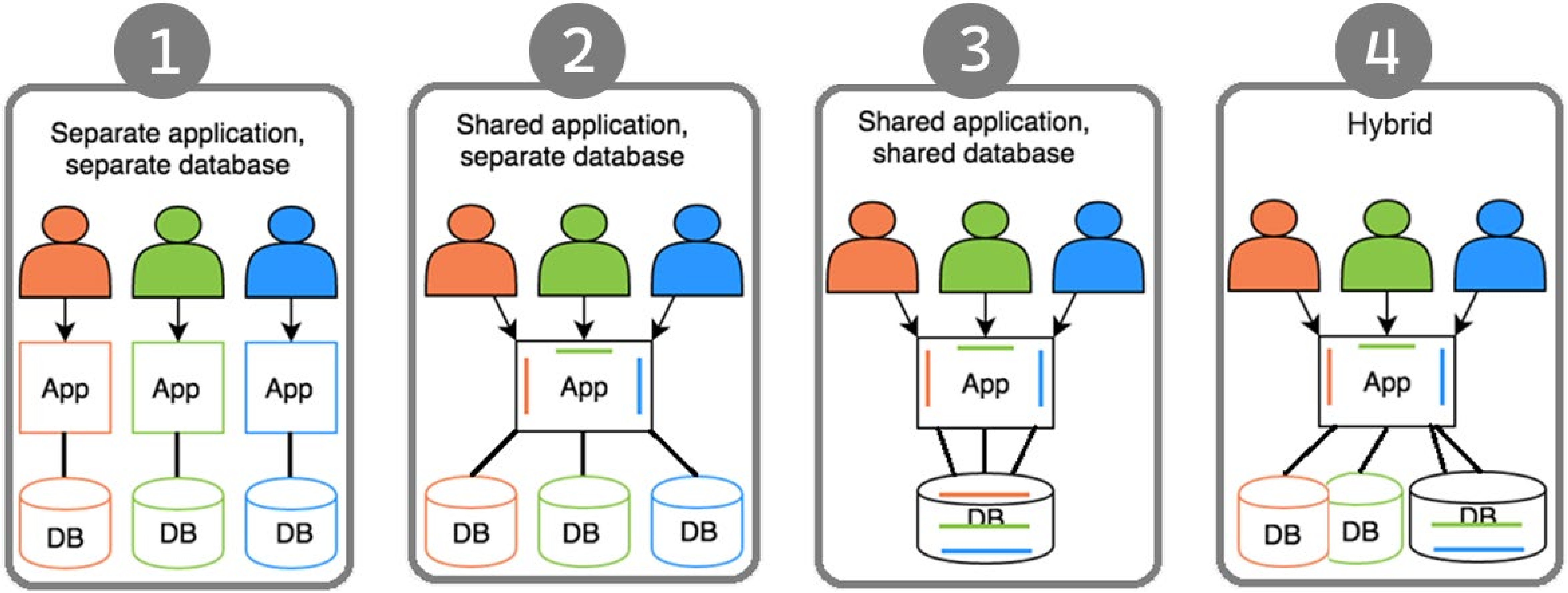
Advantages of Multi-Tenancy

1. Cost efficiency – max utilization
2. Consistent user experience
3. Ease of maintenance
4. Scalability
5. Rapid deployment for new users

Challenges of Multi-Tenancy

1. **Data isolation**
2. **Configuration** & **customization** per tenant
3. Performance balance: **Noisy neighbors!**
4. **Security**
5. **Backup** and recovery

Deployment & Database Architectures


1
Separate application, separate database
App
App
App
DB
DB
DB
2
Shared application, separate database
App
DB
DB
DB
3
Shared application, shared database
App
DB
4
Hybrid
App
DB
DB
DB

abp.io

Maintaining Application States

Application code & services should be stateless!

Where should we save the state? 🤔

- ✓ **HTTP Request** (cookie, header, query string, payload)
- ✓ **Authentication ticket**
- ✓ **Database**
- ✓ **Distributed cache** (Redis, Memcached, ...)

Identifying the Active Tenant

Identifying the Active Tenant

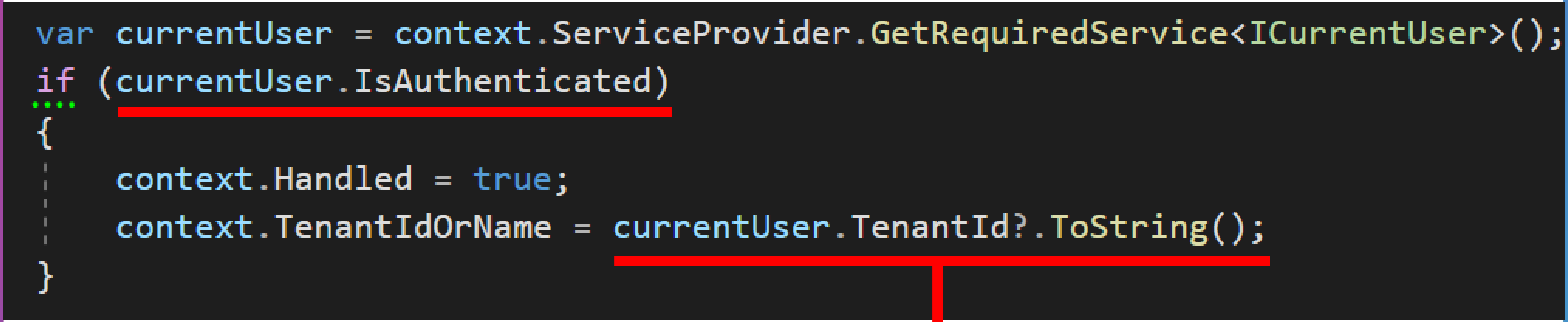
How to determine the current tenant? 🤔

1. CurrentUserTenantResolveContributor
2. QueryStringTenantResolveContributor
3. RouteTenantResolveContributor
4. HeaderTenantResolveContributor
5. CookieTenantResolveContributor
6. DomainTenantResolver

abp.io

Identifying the Active Tenant

1. Current User (claims)

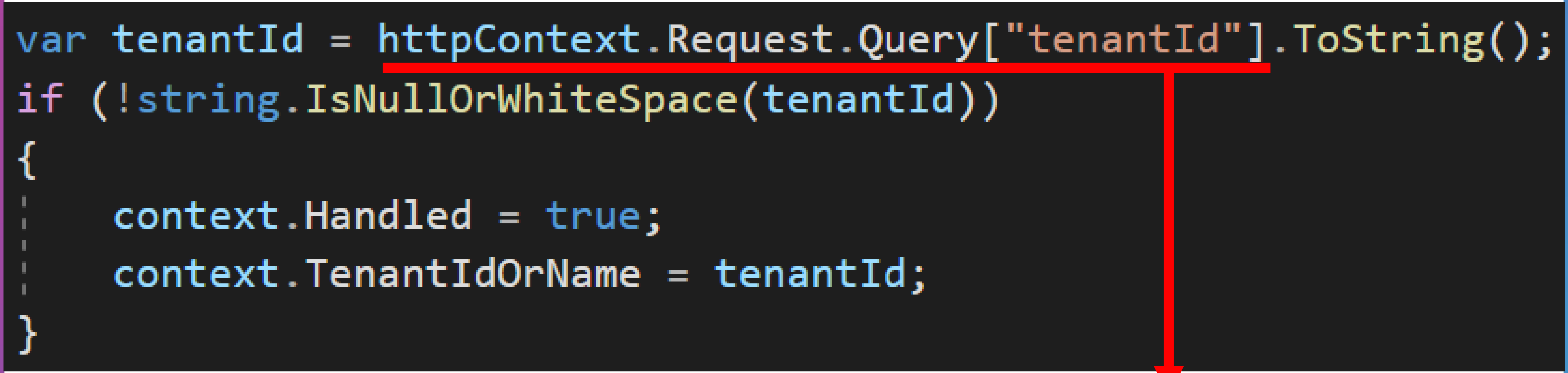


```
var currentUser = context.ServiceProvider.GetRequiredService<ICurrentUser>();
if (currentUser.IsAuthenticated)
{
    context.Handled = true;
    context.TenantIdOrName = currentUser.TenantId?.ToString();
}
```

```
HttpContext.User.Identity.Claims
.FirstOrDefault(c => c.Type == “TenantId”)
```

Identifying the Active Tenant

2. Query String

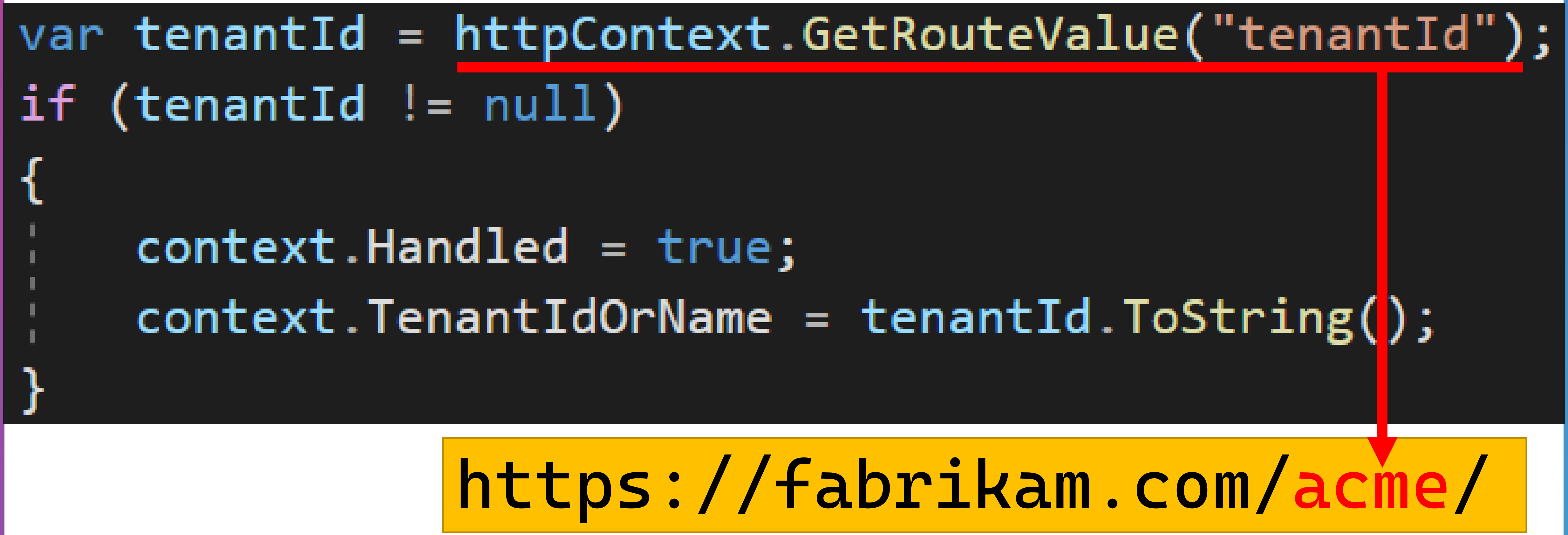


```
var tenantId = httpContext.Request.Query["tenantId"].ToString();
if (!string.IsNullOrWhiteSpace(tenantId))
{
    context.Handled = true;
    context.TenantIdOrName = tenantId;
}
```

`https://fabrikam.com?tenantId=3`

Identifying the Active Tenant

3. Route

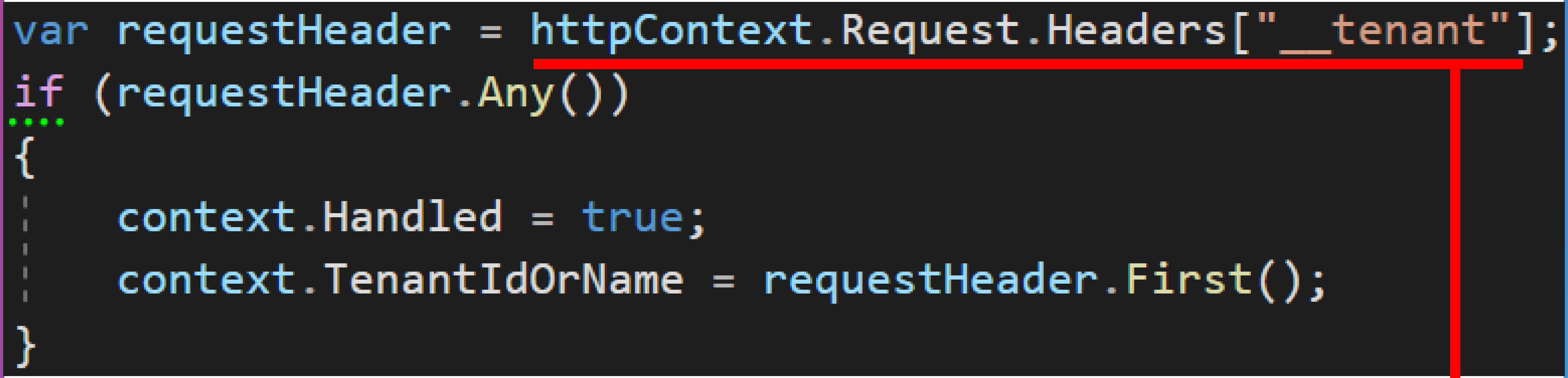


```
var tenantId = httpContext.GetRouteValue("tenantId");
if (tenantId != null)
{
    context.Handled = true;
    context.TenantIdOrName = tenantId.ToString();
}
```

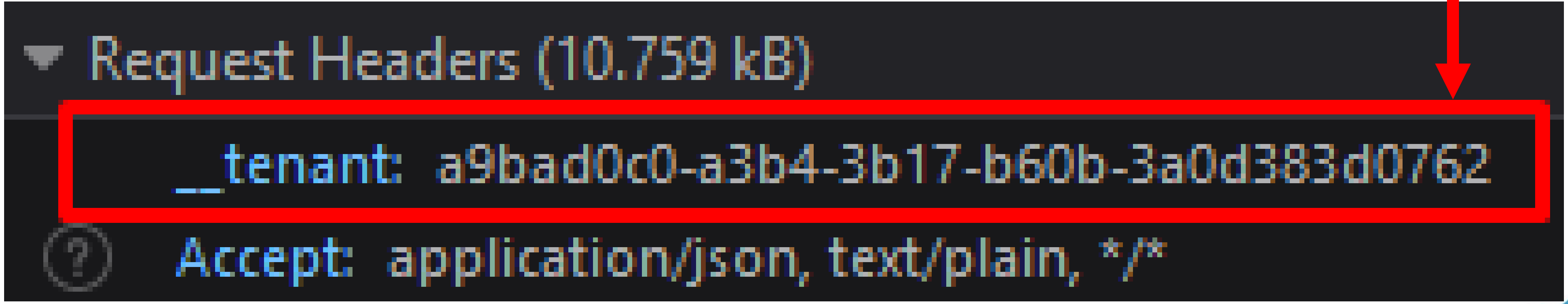
https://fabrikam.com/acme/

Identifying the Active Tenant

4. Header



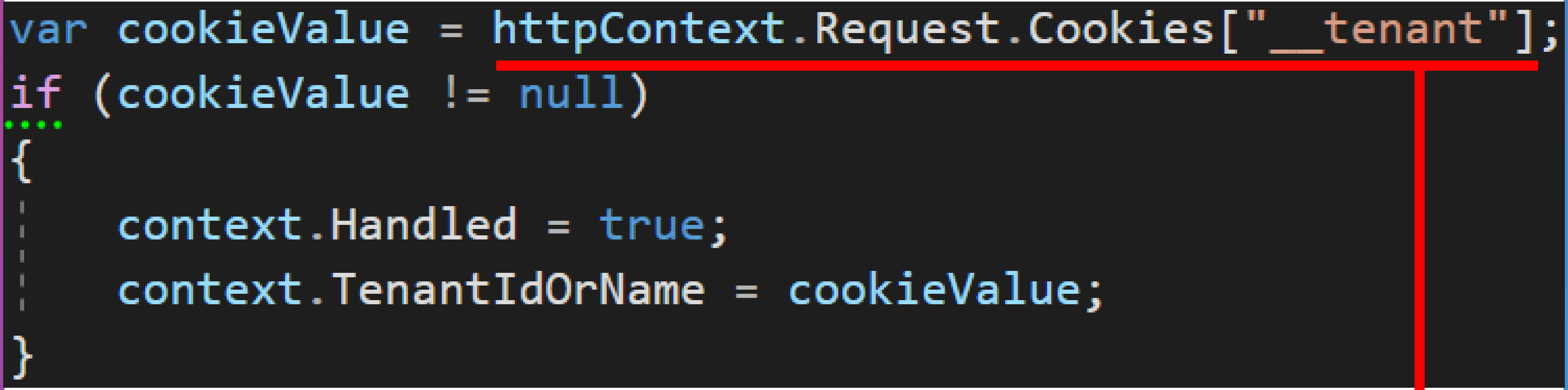
```
var requestHeader = httpContext.Request.Headers["__tenant"];
if (requestHeader.Any())
{
    context.Handled = true;
    context.TenantIdOrName = requestHeader.First();
}
```



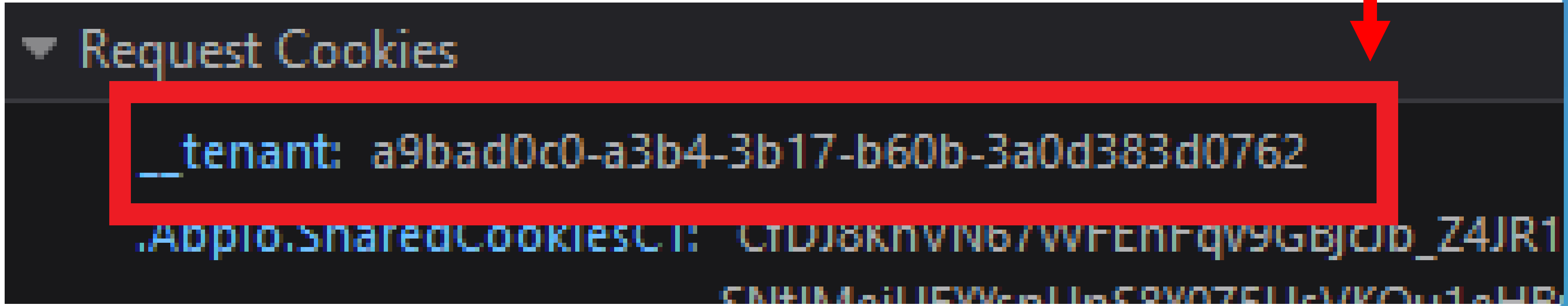
```
Request Headers (10.759 kB)
    __tenant: a9bad0c0-a3b4-3b17-b60b-3a0d383d0762
    Accept: application/json, text/plain, */*
```

Identifying the Active Tenant

5. Cookie



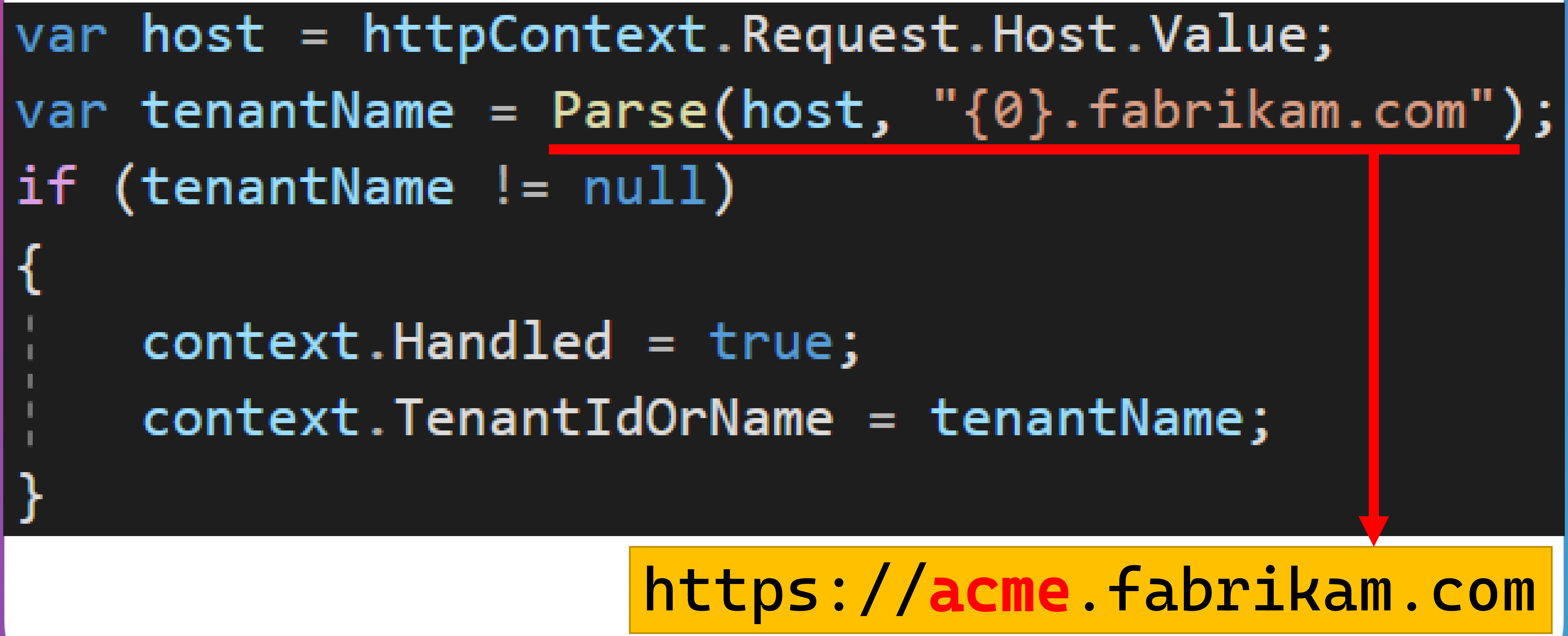
```
var cookieValue = httpContext.Request.Cookies["__tenant"];
if (cookieValue != null)
{
    context.Handled = true;
    context.TenantIdOrName = cookieValue;
}
```



```
▼ Request Cookies
    __tenant: a9bad0c0-a3b4-3b17-b60b-3a0d383d0762
    .AbpIo.SharedCookiesCT: CfDJ8KhVN67WFEhFqv9GBjQb_Z4JR1
```

Identifying the Active Tenant

6. Domain



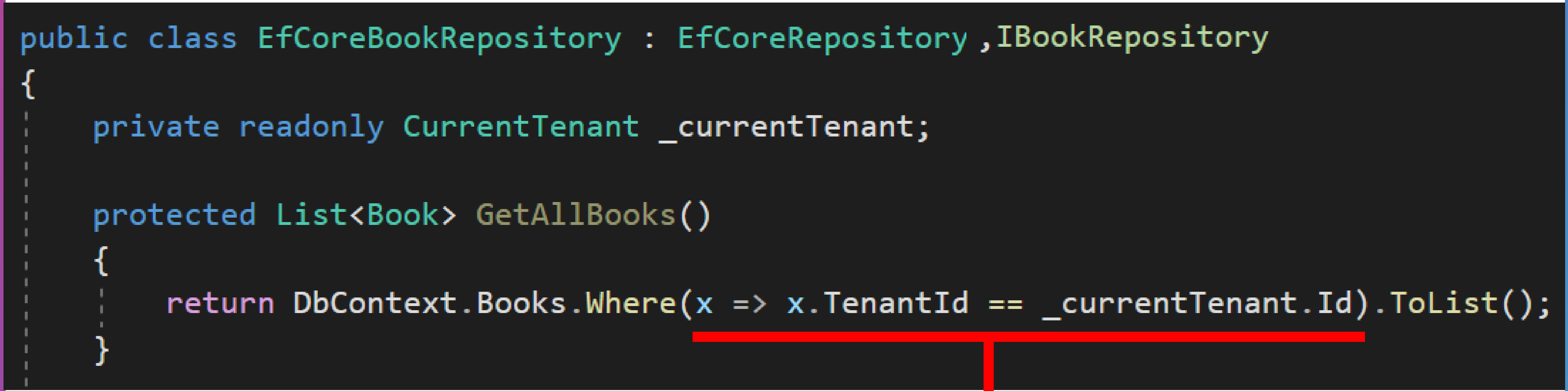
```
var host = httpContext.Request.Host.Value;
var tenantName = Parse(host, "{0}.fabrikam.com");
if (tenantName != null)
{
    context.Handled = true;
    context.TenantIdOrName = tenantName;
}
```

https://acme.fabrikam.com

✓ Identifying the Active Tenant

Data Isolation

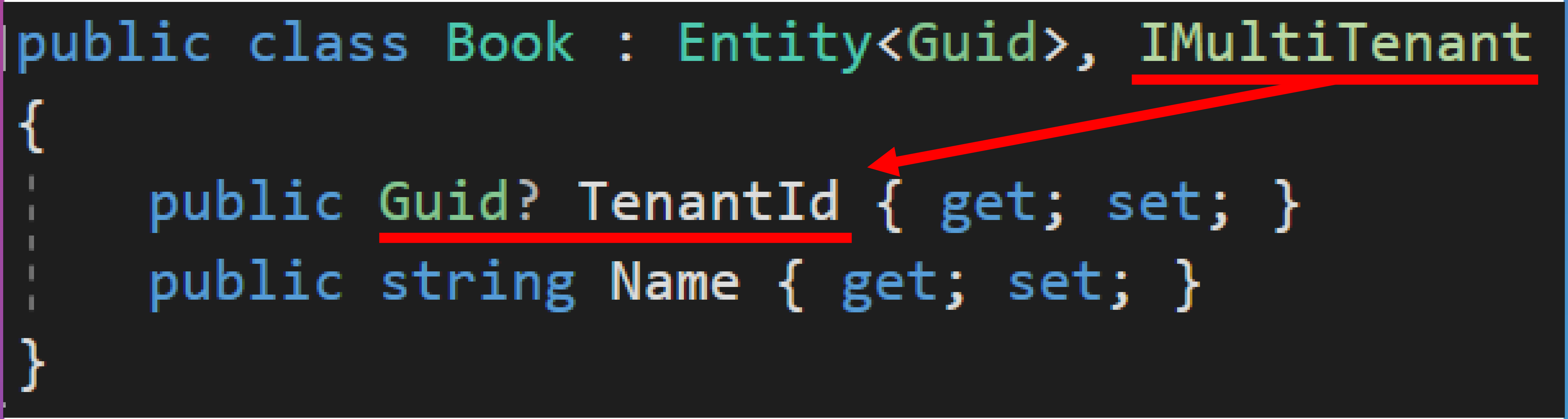
Data Isolation – Traditional way



```
public class EfCoreBookRepository : EfCoreRepository ,IBookRepository
{
    private readonly CurrentTenant _currentTenant;

    protected List<Book> GetAllBooks()
    {
        return DbContext.Books.Where(x => x.TenantId == _currentTenant.Id).ToList();
    }
```

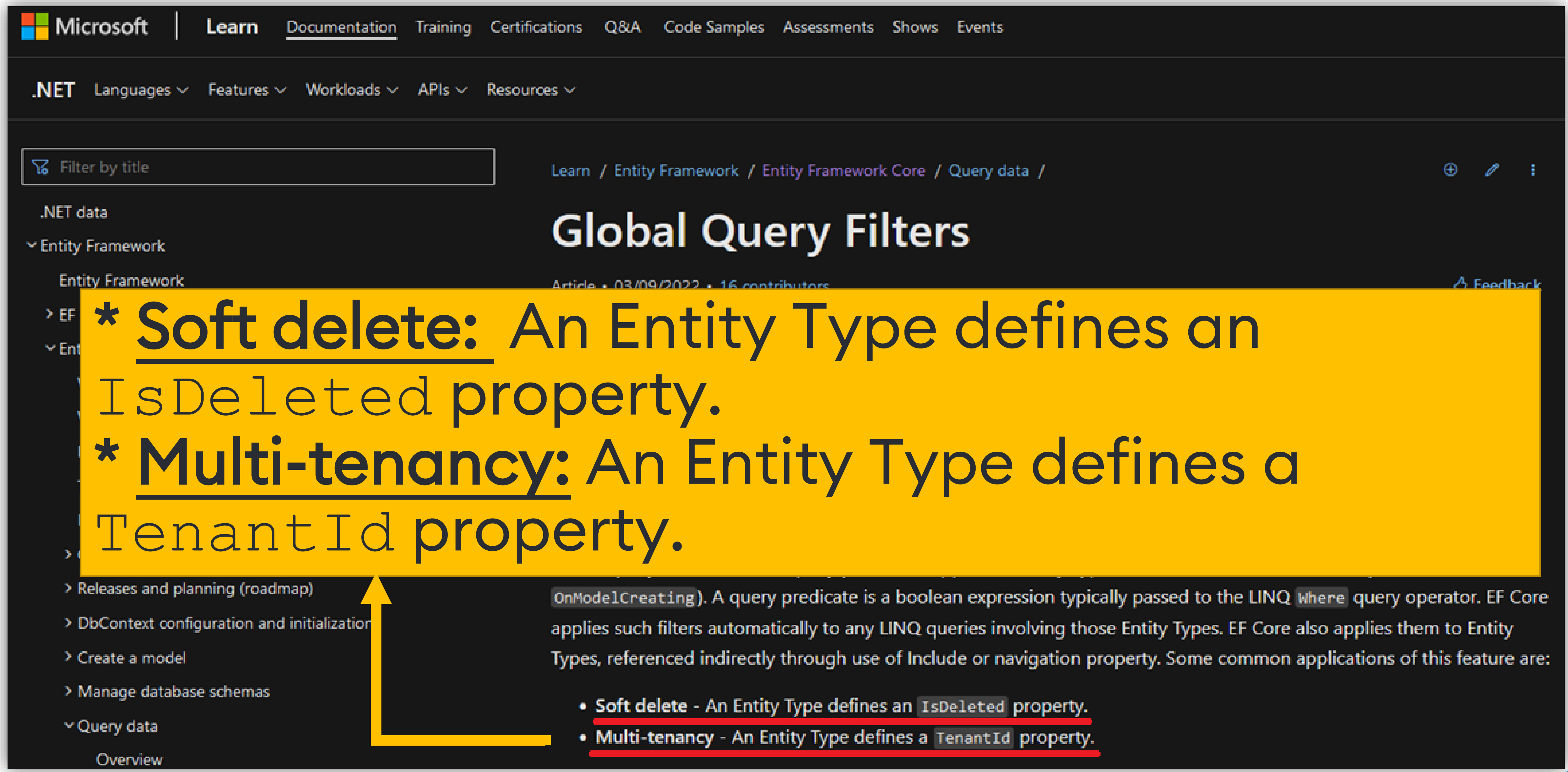
You normally do this

Data Isolation



```
public class Book : Entity<Guid>, IMultiTenant
{
    public Guid? TenantId { get; set; }
    public string Name { get; set; }
}
```

Data Isolation – EF Core



Microsoft | Learn Documentation Training Certifications Q&A Code Samples Assessments Shows Events

.NET Languages Features Workloads APIs Resources

Filter by title

.NET data
Entity Framework
Entity Framework
EF
Ent
Releases and planning (roadmap)
DbContext configuration and initialization
Create a model
Manage database schemas
Query data
Overview

Learn / Entity Framework / Entity Framework Core / Query data /

Global Query Filters

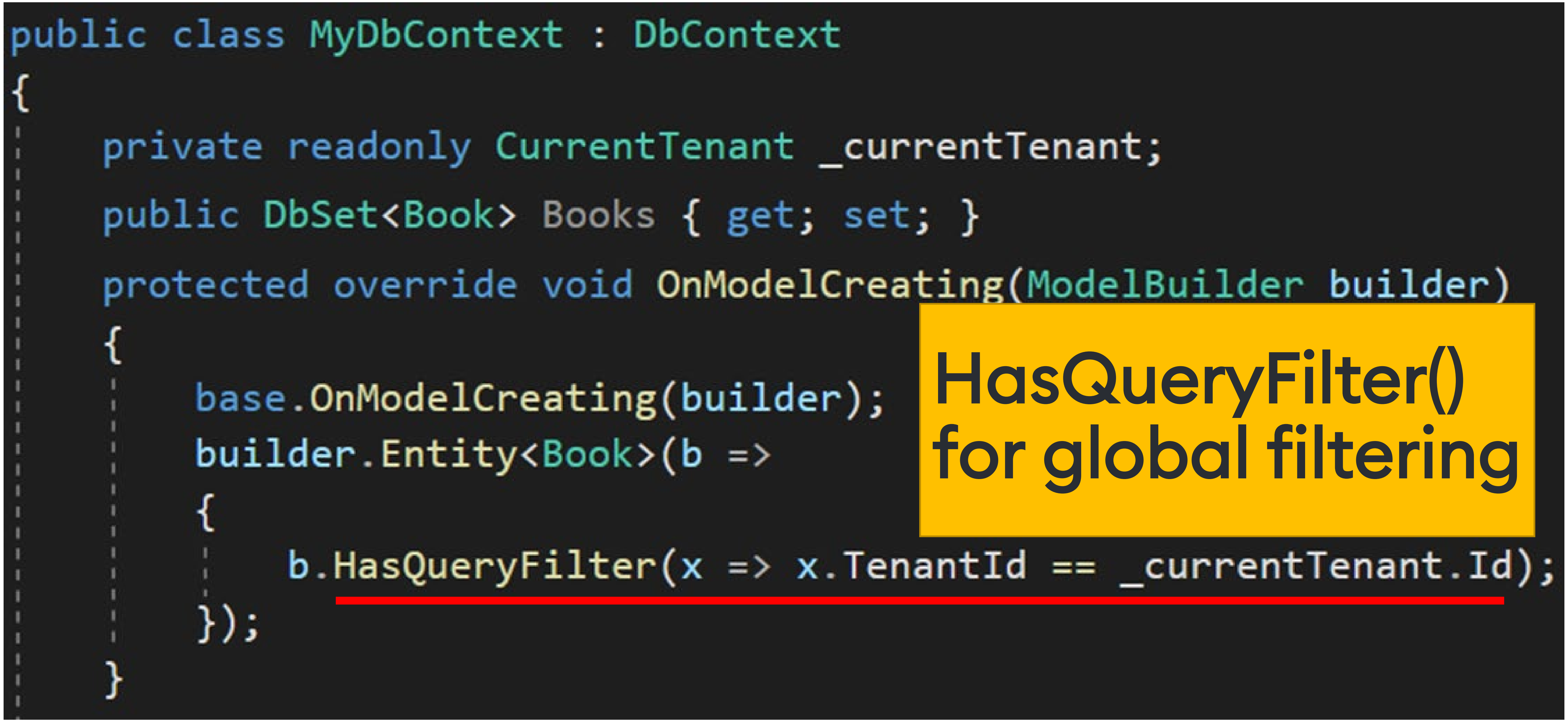
Article • 03/09/2022 • 16 contributors   Feedback

* **Soft delete:** An Entity Type defines an `IsDeleted` property.
* **Multi-tenancy:** An Entity Type defines a `TenantId` property.

`OnModelCreating`). A query predicate is a boolean expression typically passed to the LINQ `Where` query operator. EF Core applies such filters automatically to any LINQ queries involving those Entity Types. EF Core also applies them to Entity Types, referenced indirectly through use of Include or navigation property. Some common applications of this feature are:

- **Soft delete** - An Entity Type defines an `IsDeleted` property.
- **Multi-tenancy** - An Entity Type defines a `TenantId` property.

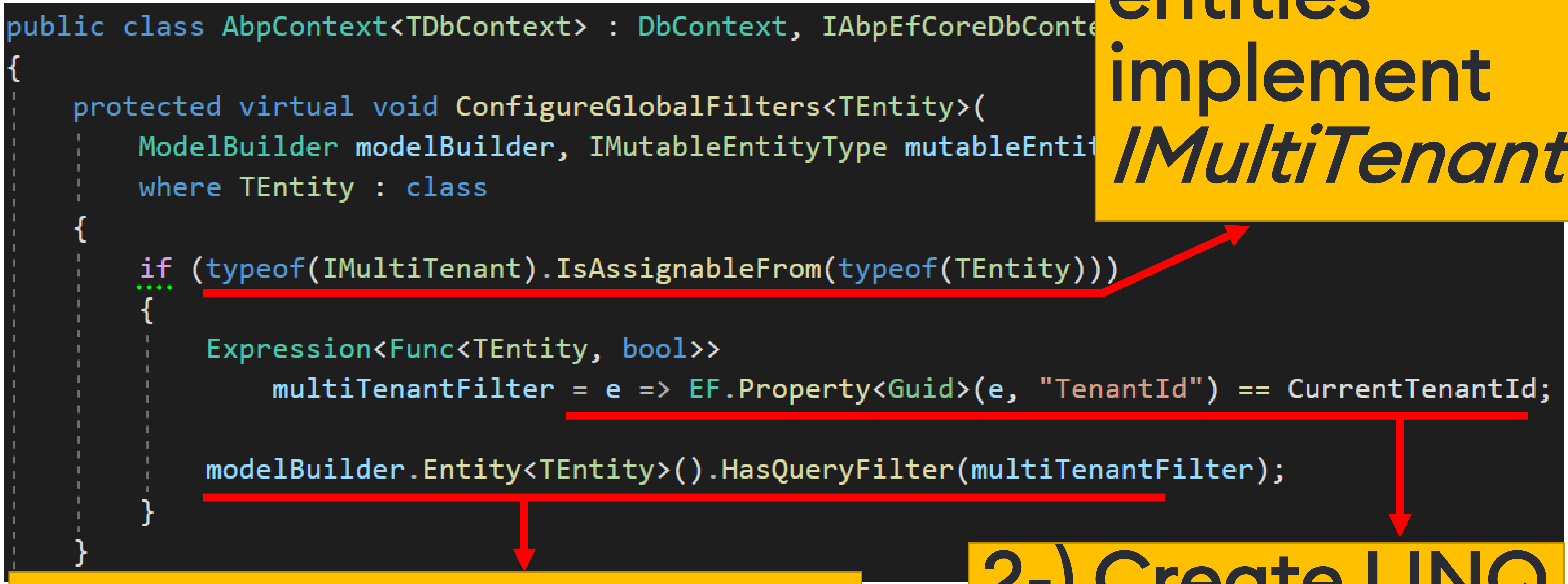
Data Isolation – EF Core Manual Way



```
public class MyDbContext : DbContext
{
    private readonly CurrentTenant _currentTenant;
    public DbSet<Book> Books { get; set; }
    protected override void OnModelCreating(ModelBuilder builder)
    {
        base.OnModelCreating(builder);
        builder.Entity<Book>(b =>
        {
            b.HasQueryFilter(x => x.TenantId == _currentTenant.Id);
        });
    }
```

HasQueryFilter() for global filtering

Data Isolation – EF Core



```
public class AbpContext<TDbContext> : DbContext, IAbpEfCoreDbConte
{
    protected virtual void ConfigureGlobalFilters<TEntity>(
        ModelBuilder modelBuilder, IMutableEntityType mutableEntit
        where TEntity : class
    {
        if (typeof(IMultiTenant).IsAssignableFrom(typeof(TEntity)))
        {
            Expression<Func<TEntity, bool>>
                multiTenantFilter = e => EF.Property<Guid>(e, "TenantId") == CurrentTenantId;

            modelBuilder.Entity<TEntity>().HasQueryFilter(multiTenantFilter);
        }
    }
```

1-) Find all entities implement *IMultiTenant*

2-) Create LINQ expression

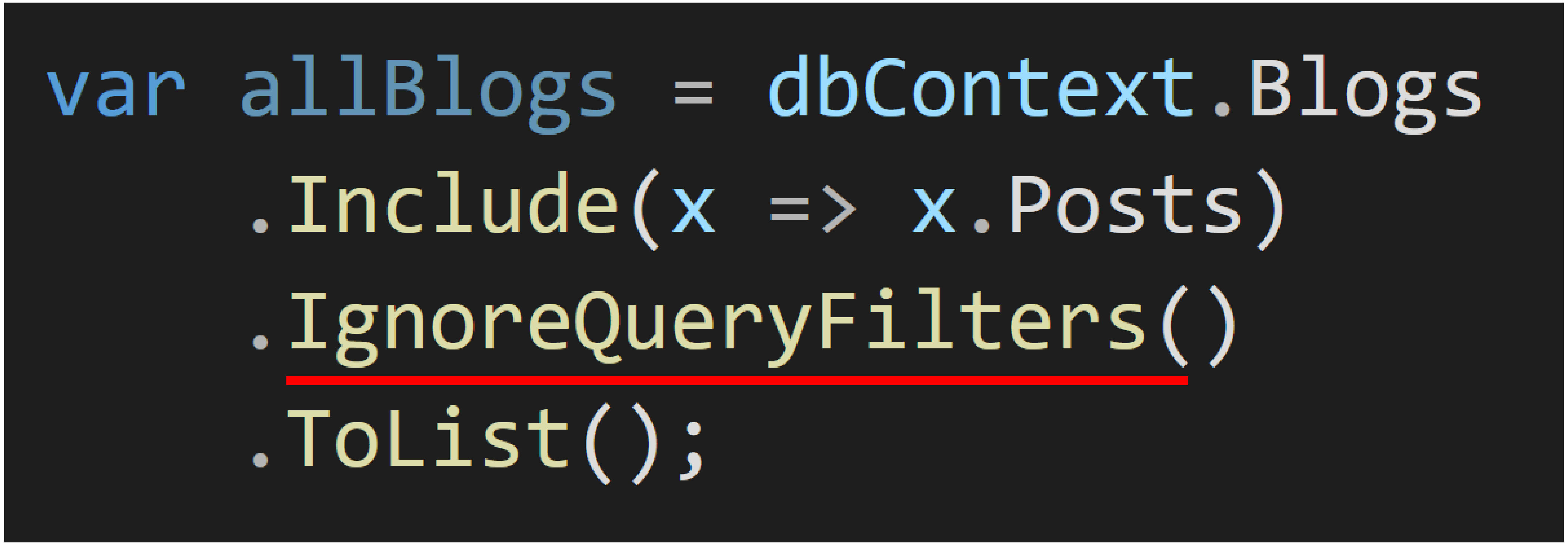
3-) Add to global filters

Data Isolation – EF Core PROS & CONS

- 😊 Easy to implement
- 😊 Supports navigation properties as well
- 🙁 Works only with EF Core

Data Isolation – EF Core PROS & CONS

😡 `IgnoreQueryFilters()` disables all filters

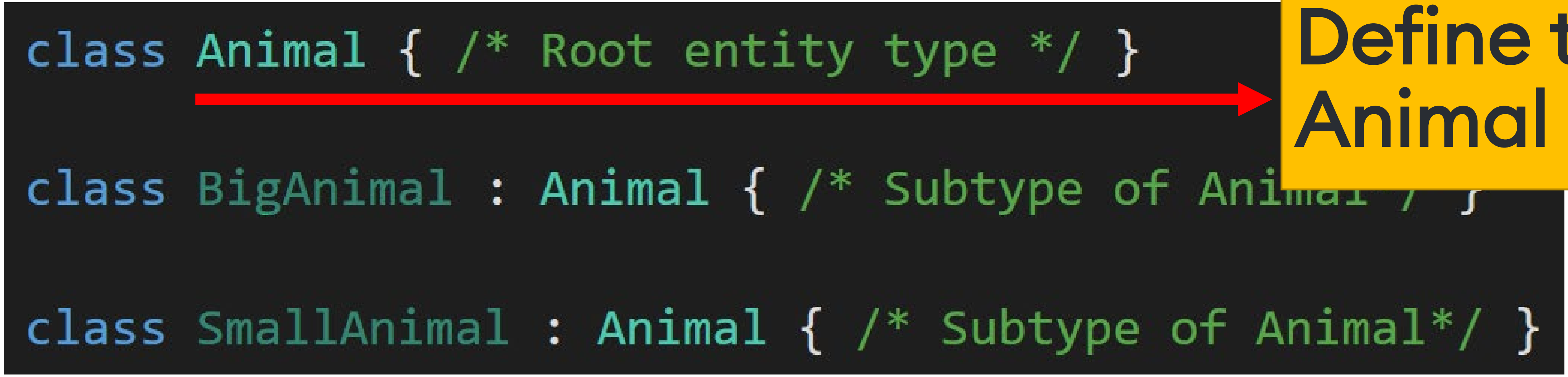


```
var allBlogs = dbContext.Blogs
    .Include(x => x.Posts)
    .IgnoreQueryFilters()
    .ToList();
```

abp.io

Data Isolation – EF Core PROS & CONS

😡 Can be defined for the root entity of the inheritance hierarchy



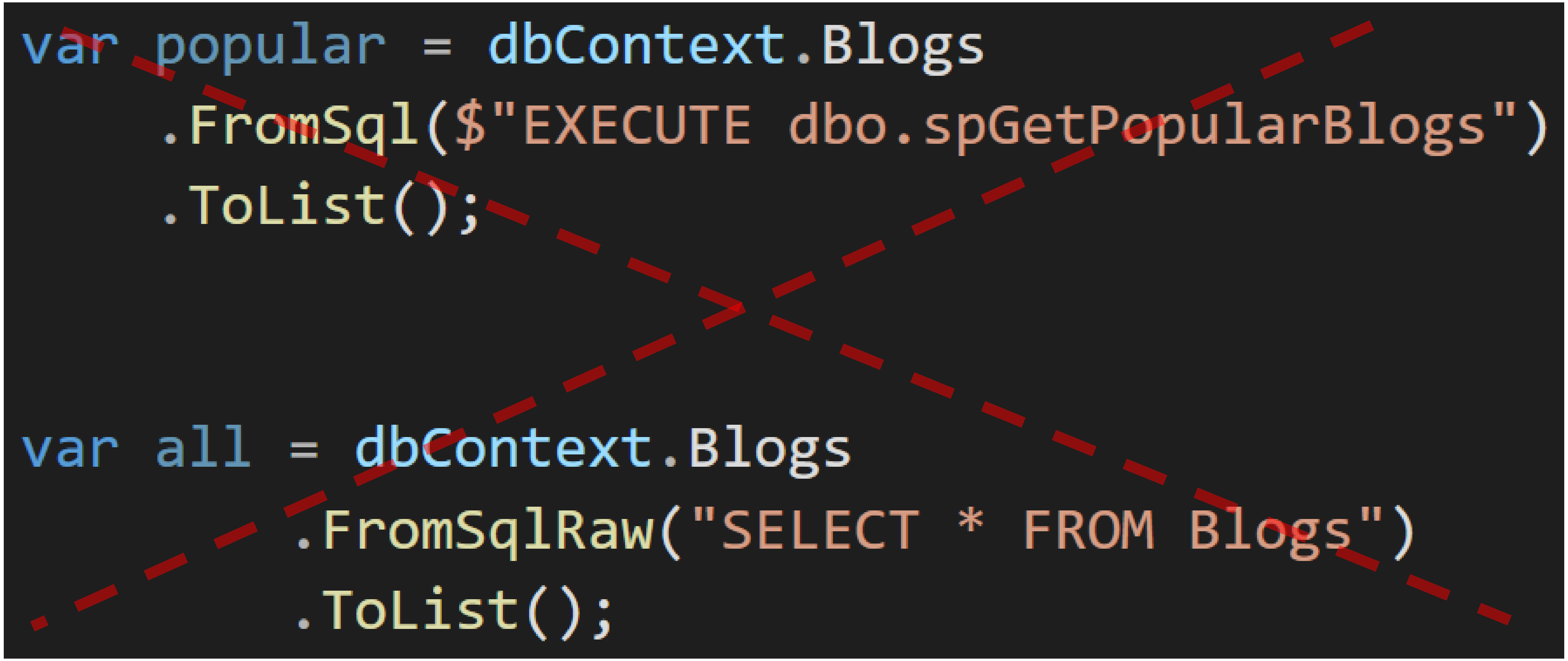
```
class Animal { /* Root entity type */ }

class BigAnimal : Animal { /* Subtype of Animal */ }

class SmallAnimal : Animal { /* Subtype of Animal*/ }
```

Define to Animal

abp.io

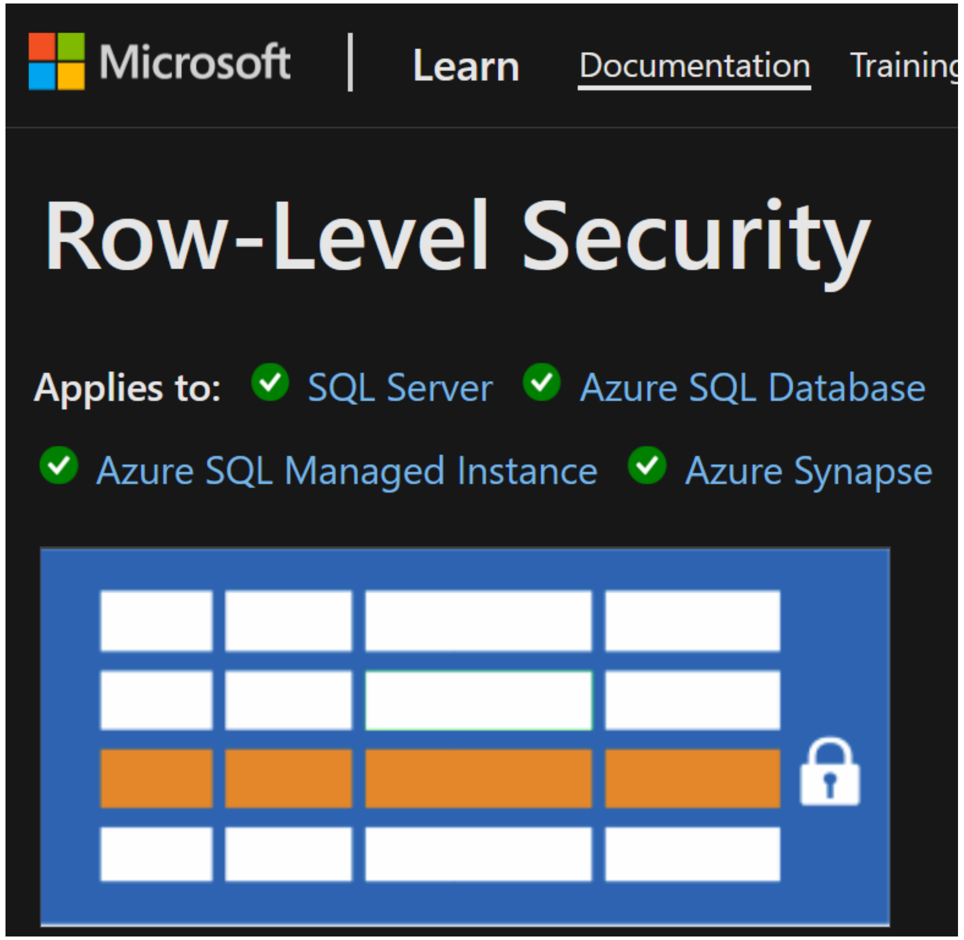
Data Isolation – EF Core PROS & CONS

😡 Does not support Stored Procedures or T-SQL



```
var popular = dbContext.Blogs
    .FromSql($"EXECUTE dbo.spGetPopularBlogs")
    .ToList();


var all = dbContext.Blogs
        .FromSqlRaw("SELECT * FROM Blogs")
        .ToList();
```

Data Isolation – EF Core PROS & CONS


Microsoft | Learn Documentation Training

Row-Level Security

Applies to: ✓ SQL Server ✓ Azure SQL Database ✓ Azure SQL Managed Instance ✓ Azure Synapse

Database level solution

☞ Row Level Security

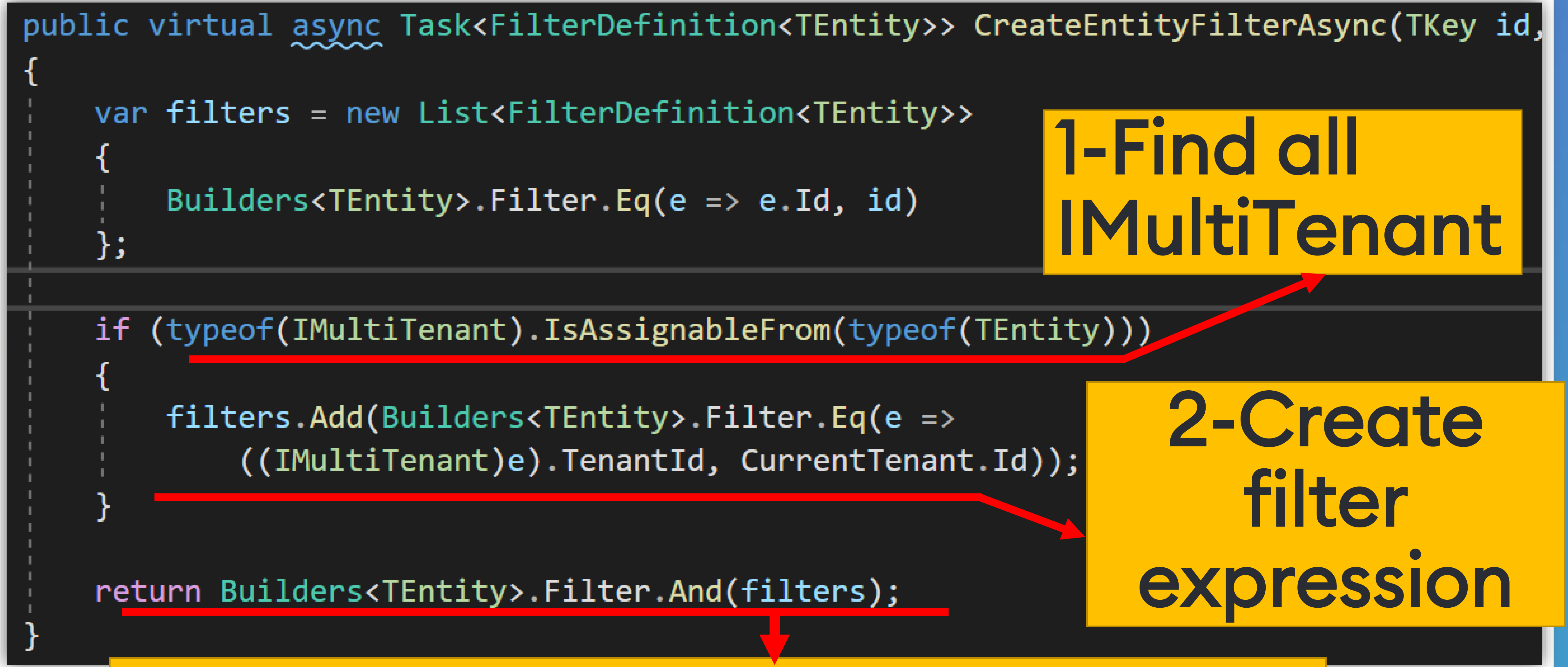
Rows filtered based on user roles, attributes

Restriction logic is done in the DB

https://learn.microsoft.com/en-us/sql/relational-databases/security/row-level-security

abp.io

Data Isolation – MongoDB



```
public virtual async Task<FilterDefinition<TEntity>> CreateEntityFilterAsync(TKey id,
{
    var filters = new List<FilterDefinition<TEntity>>
    {
        Builders<TEntity>.Filter.Eq(e => e.Id, id)
    };

    if (typeof(IMultiTenant).IsAssignableFrom(typeof(TEntity)))
    {
        filters.Add(Builders<TEntity>.Filter.Eq(e =>
            ((IMultiTenant)e).TenantId, CurrentTenant.Id));
    }

    return Builders<TEntity>.Filter.And(filters);
}
```

1-Find all IMultiTenant

2-Create filter expression

3-Add to our custom global filters

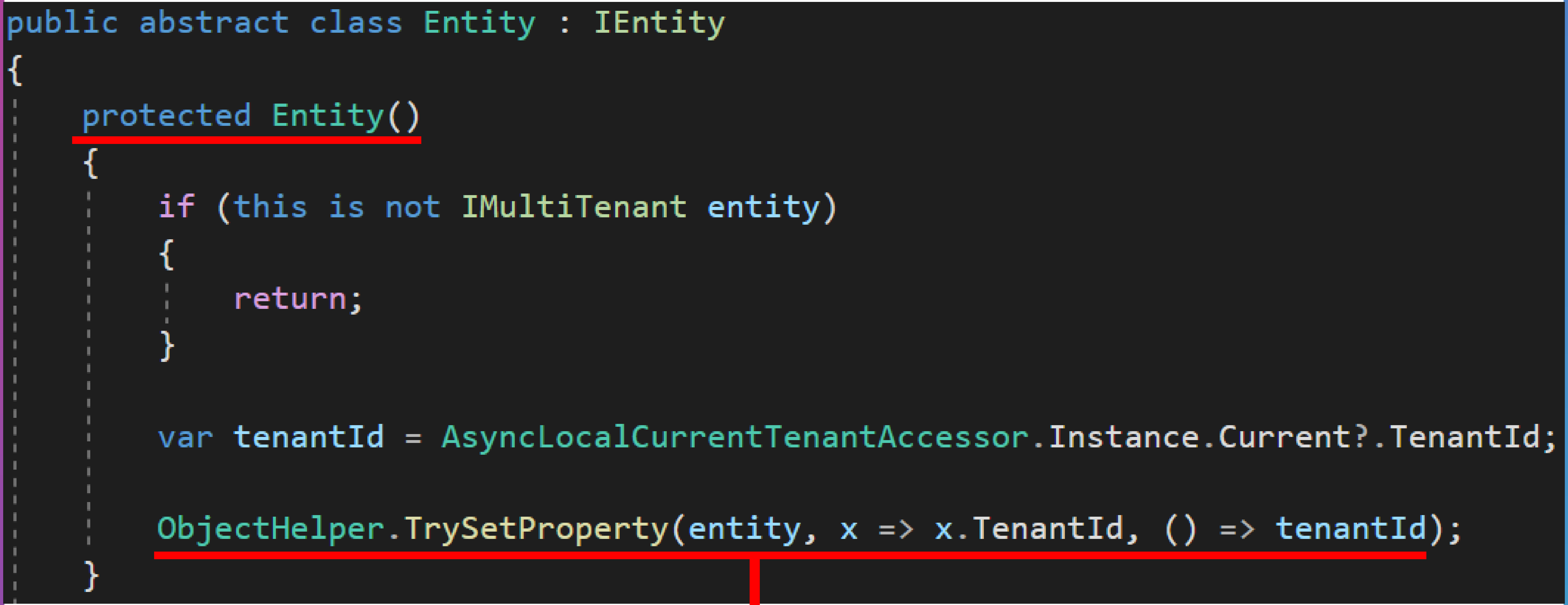
abp.io

✓ Identifying the Active Tenant

✓ Data Isolation

Set TenantId for New Entities

Set TenantId for New Entities



```
public abstract class Entity : IEntity
{
    protected Entity()
    {
        if (this is not IMultiTenant entity)
        {
            return;
        }

        var tenantId = AsyncLocalCurrentTenantAccessor.Instance.Current?.TenantId;

        ObjectHelper.TrySetProperty(entity, x => x.TenantId, () => tenantId);
    }
```

Set TenantId by reflection

abp.io

✓ Identifying the Active Tenant

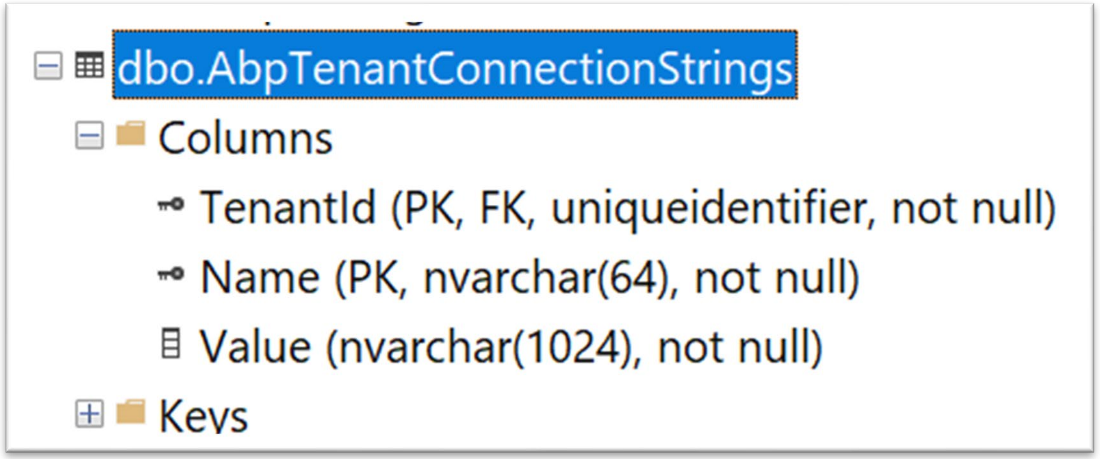
✓ Data Isolation

✓ Set TenantId for New Entities

DB Connection String Selection

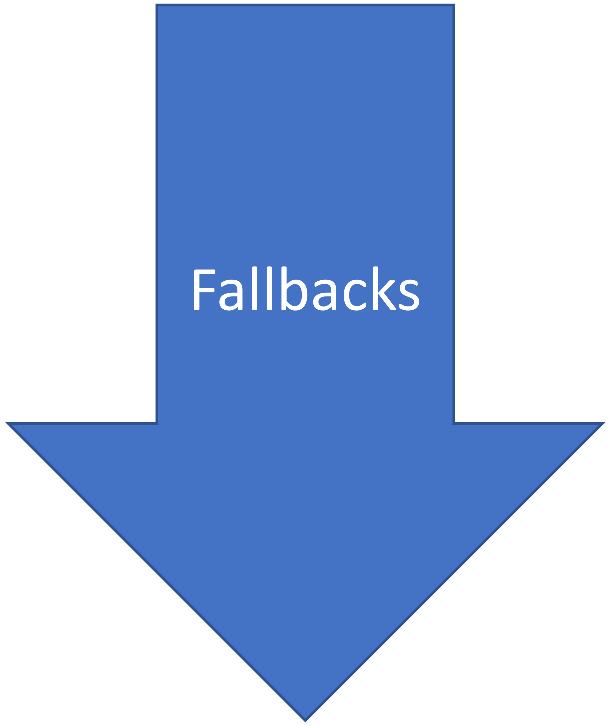
Connection String Selection – DB

1. The **current tenant**

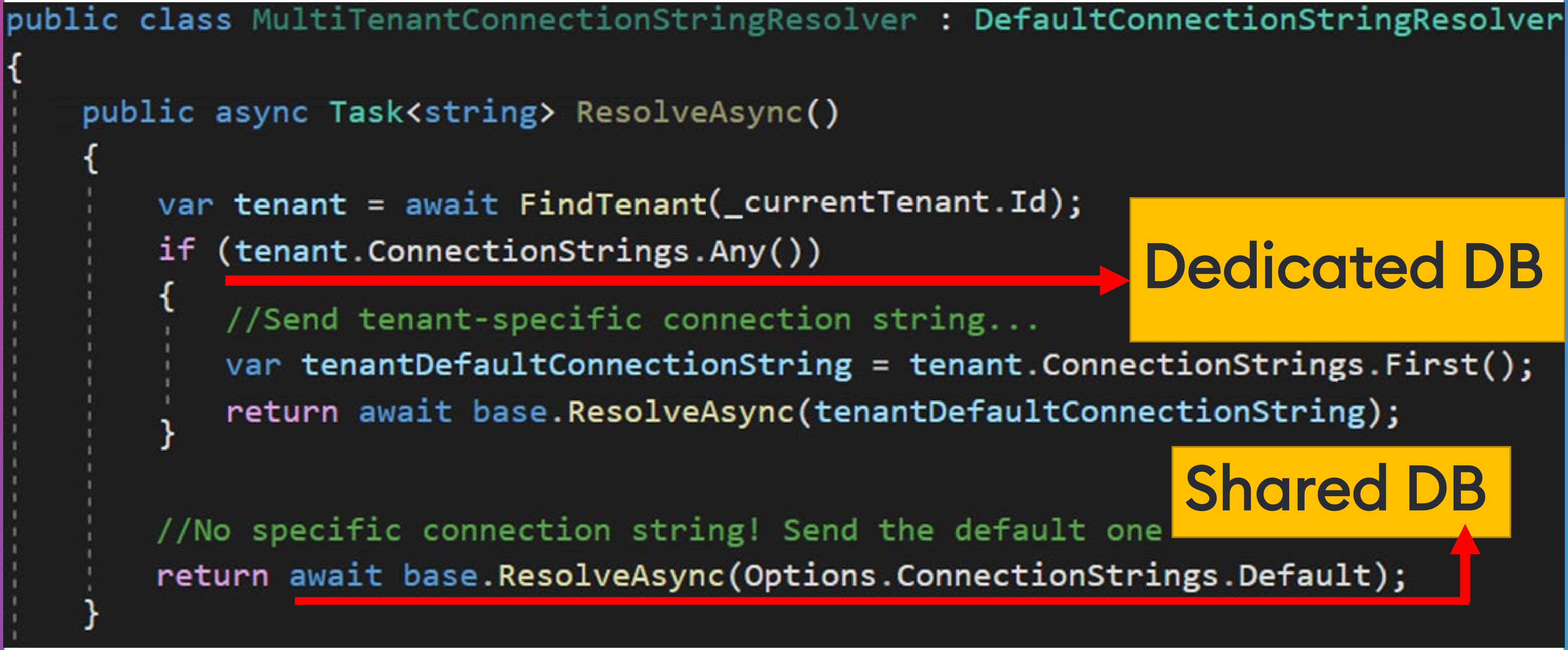


```
⊟ dbo.AbpTenantConnectionStrings
  ⊟ Columns
      TenantId (PK, FK, uniqueidentifier, not null)
      Name (PK, nvarchar(64), not null)
      Value (nvarchar(1024), not null)
  ⊞ Keys
```

2. The **current module / microservice**
3. The **default** connection string



Connection String Selection – Code



```
public class MultiTenantConnectionStringResolver : DefaultConnectionStringResolver
{
    public async Task<string> ResolveAsync()
    {
        var tenant = await FindTenant(_currentTenant.Id);
        if (tenant.ConnectionStrings.Any())
        {
            //Send tenant-specific connection string...
            var tenantDefaultConnectionString = tenant.ConnectionStrings.First();
            return await base.ResolveAsync(tenantDefaultConnectionString);
        }

        //No specific connection string! Send the default one
        return await base.ResolveAsync(Options.ConnectionStrings.Default);
    }
```

abp.io

✓ Identifying the Active Tenant

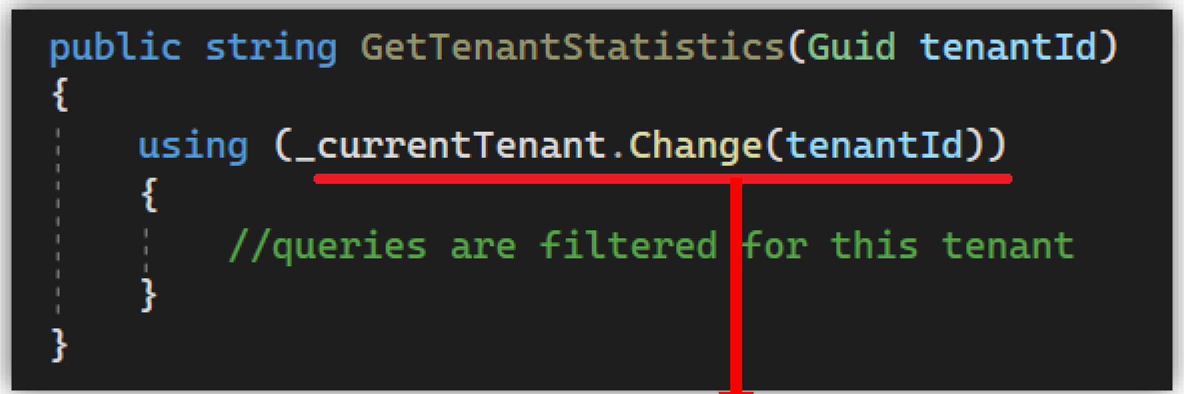
✓ Data Isolation

✓ Set TenantId for New Entities

✓ DB Connection String Selection

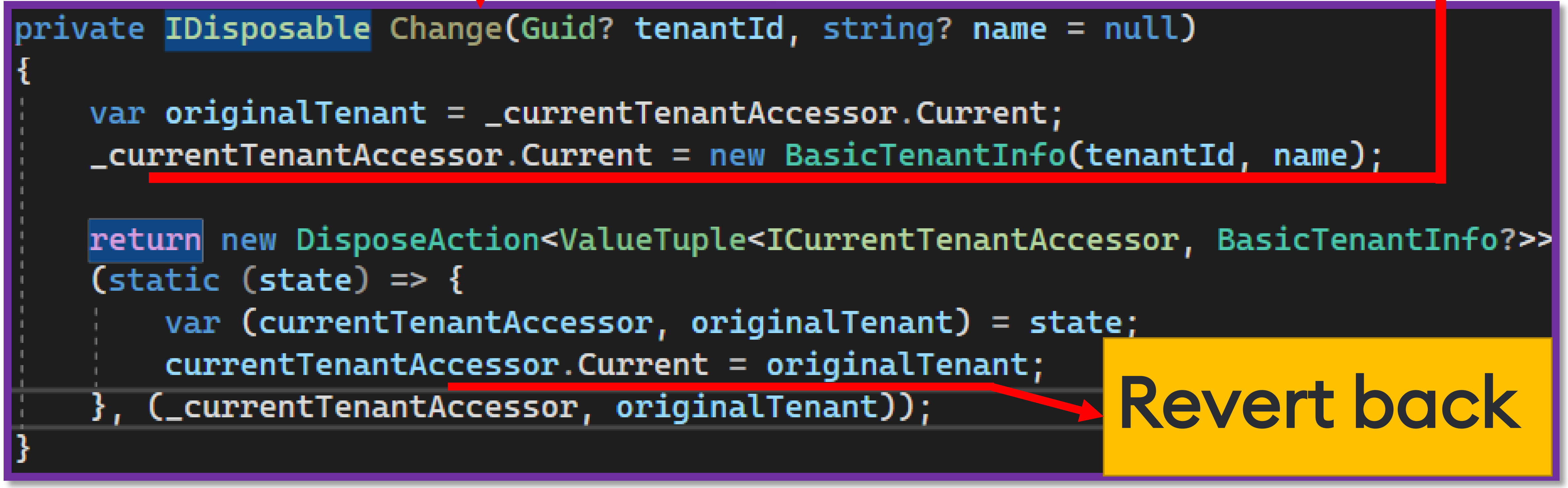
Changing the Active Tenant

Changing the Active Tenant



```
public string GetTenantStatistics(Guid tenantId)
{
    using (_currentTenant.Change(tenantId))
    {
        //queries are filtered for this tenant
    }
}
```

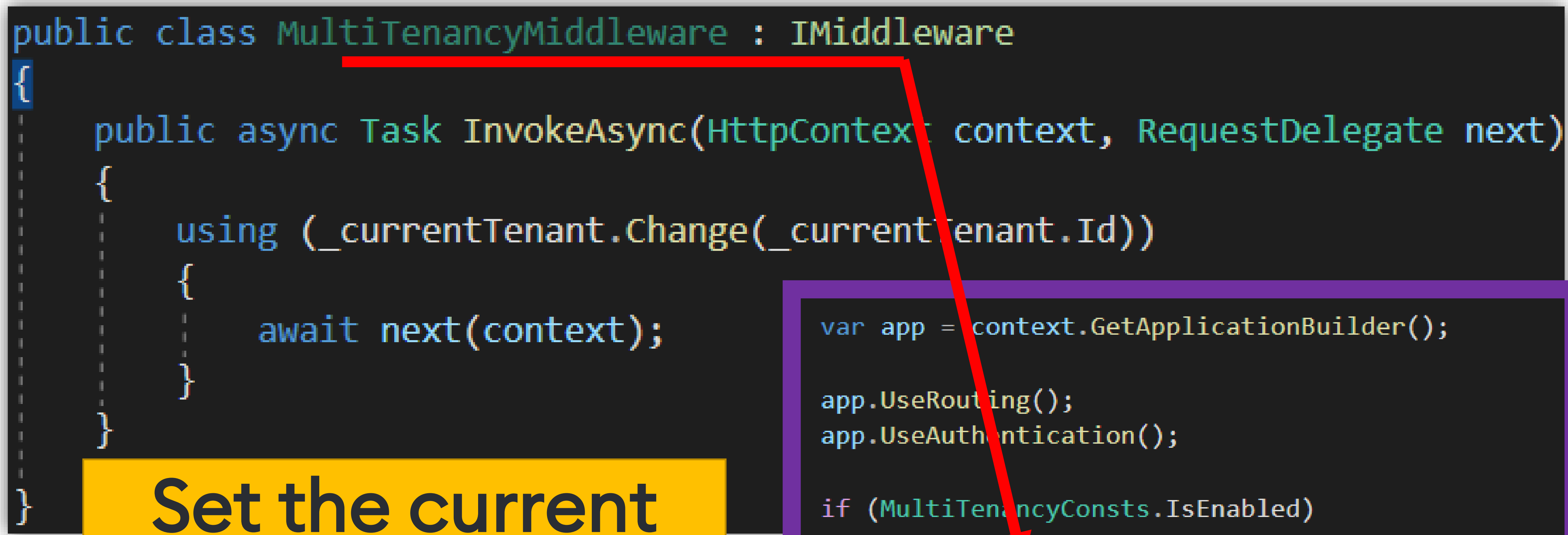
Set active tenant



```
private IDisposable Change(Guid? tenantId, string? name = null)
{
    var originalTenant = _currentTenantAccessor.Current;
    _currentTenantAccessor.Current = new BasicTenantInfo(tenantId, name);

    return new DisposeAction<ValueTuple<ICurrentTenantAccessor, BasicTenantInfo?>>
    (static (state) => {
        var (currentTenantAccessor, originalTenant) = state;
        currentTenantAccessor.Current = originalTenant;
    }, (_currentTenantAccessor, originalTenant));
}
```

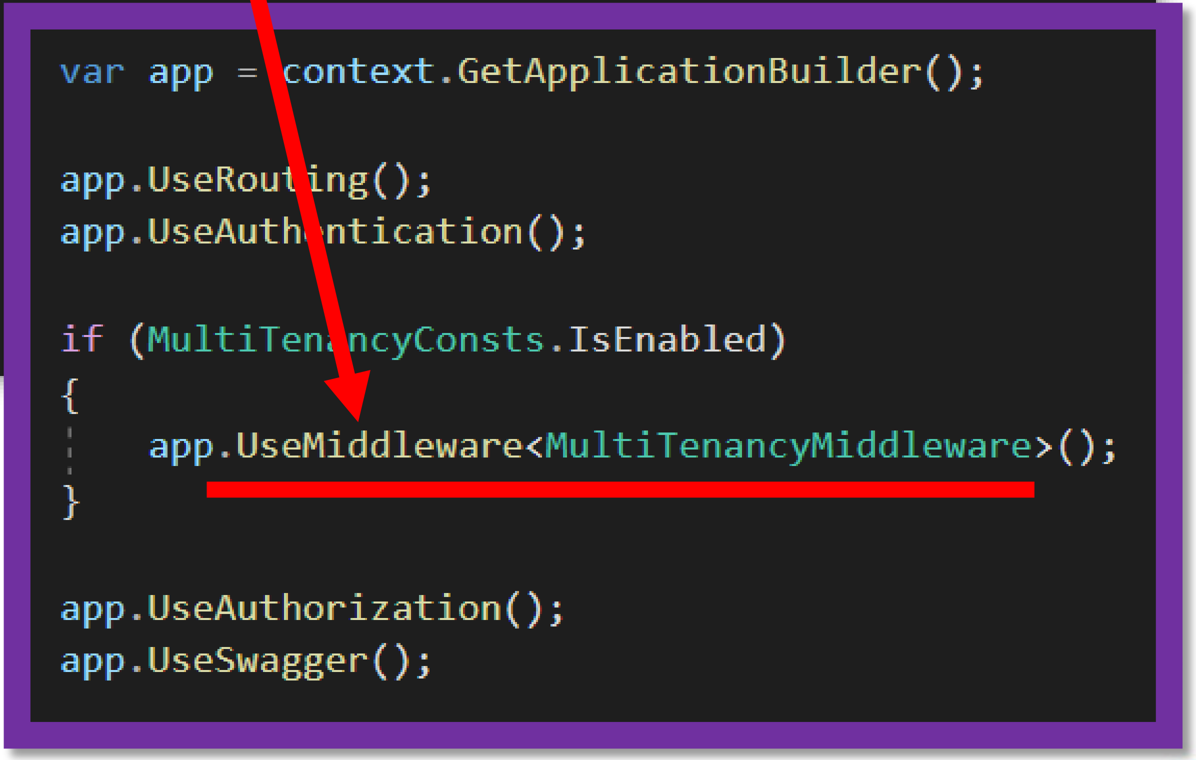
Revert back

Setting the Active Tenant in Middleware



```
public class MultiTenancyMiddleware : IMiddleware
{
    public async Task InvokeAsync(HttpContext context, RequestDelegate next)
    {
        using (_currentTenant.Change(_currentTenant.Id))
        {
            await next(context);
        }
    }
}
```

Set the current tenant within the middleware

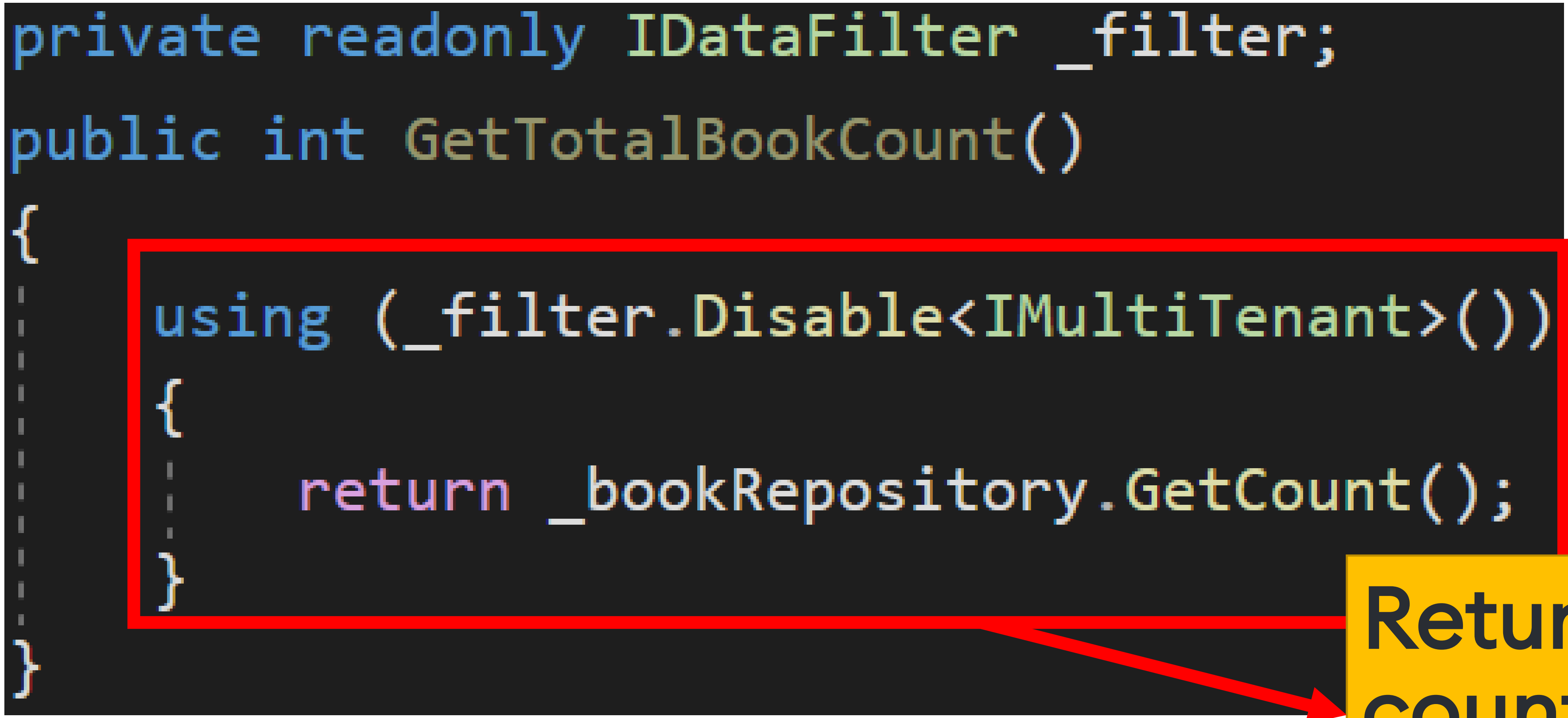


```
var app = context.GetApplicationBuilder();

app.UseRouting();
app.UseAuthentication();

if (MultiTenancyConsts.IsEnabled)
{
    app.UseMiddleware<MultiTenancyMiddleware>();
}

app.UseAuthorization();
app.UseSwagger();
```

✓ Identifying the Active Tenant
✓ Data Isolation
✓ Set TenantId for New Entities
✓ DB Connection String Selection
✓ Changing the Active Tenant

Temporarily Disable Multi-Tenancy

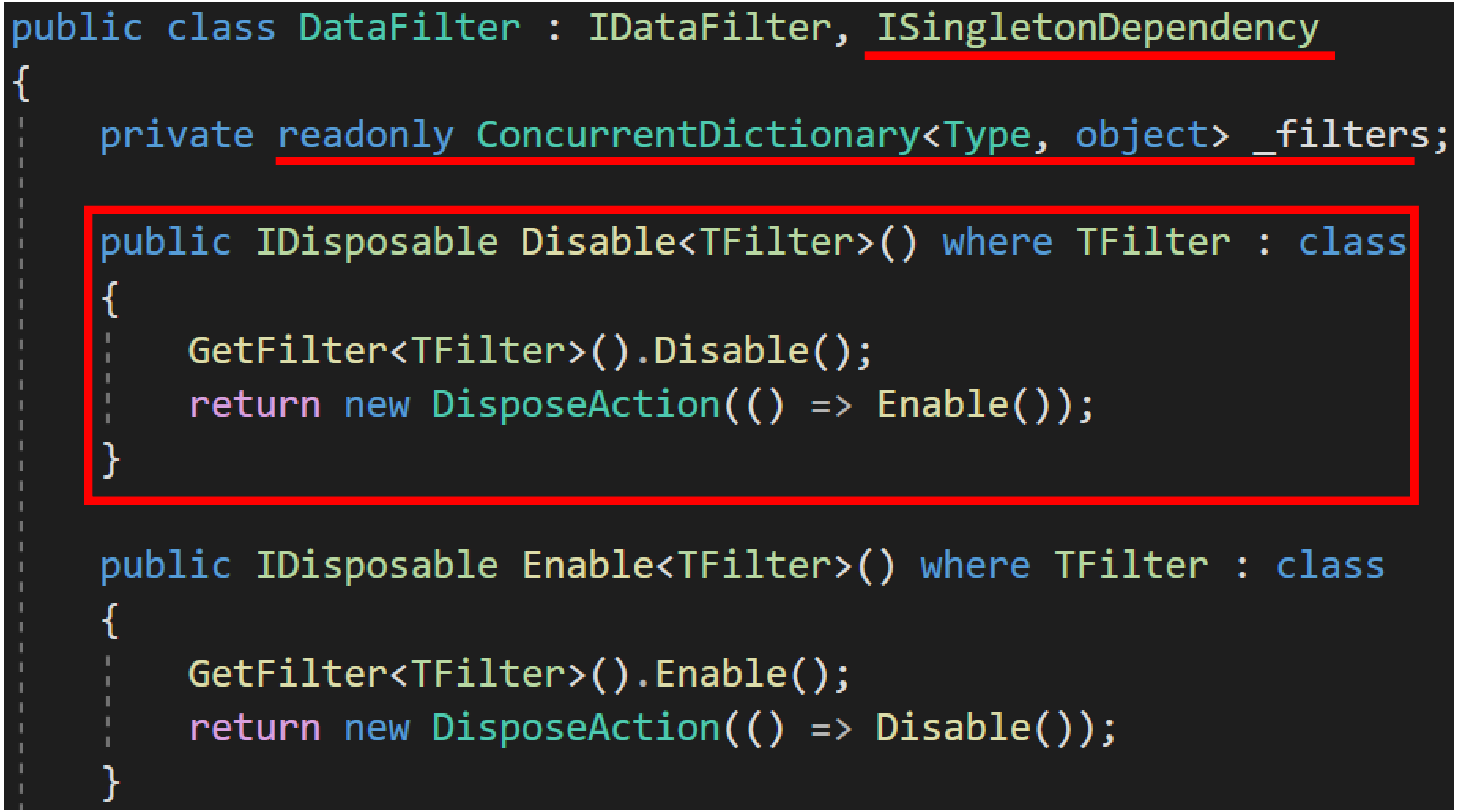
Disabling Multi-Tenancy Filter (Usage)



```
private readonly IDataFilter _filter;
public int GetTotalBookCount()
{
    using (_filter.Disable<IMultiTenant>())
    {
        return _bookRepository.GetCount();
    }
}
```

Returns book count without tenantId filter

Disabling Multi-Tenancy Filter (Implementation)



```
public class DataFilter : IDataFilter, ISingletonDependency
{
    private readonly ConcurrentDictionary<Type, object> _filters;

    public IDisposable Disable<TFilter>() where TFilter : class
    {
        GetFilter<TFilter>().Disable();
        return new DisposeAction(() => Enable());
    }

    public IDisposable Enable<TFilter>() where TFilter : class
    {
        GetFilter<TFilter>().Enable();
        return new DisposeAction(() => Disable());
    }
```

✓ Identifying the Active Tenant
✓ Data Isolation
✓ Set TenantId for New Entities
✓ DB Connection String Selection
✓ Changing the Active Tenant
✓ Temporarily Disable Multi-Tenancy

Database Migration

Database Migration

Approach-1: **Make DB migration with a custom tool**

😊 Easy to implement. All tenants are in the same version

😡 May get too long time for big number of tenants and data.

😡 All tenants wait for all upgrade progress

Approach-2: **Run migration on first DB access**

😊 Upgrading is distributed to time. A tenant does not wait for another

😡 First user may wait too much and see timeout exception.

😡 Hard to implement (concurrency problems)!

abp.io

Database Migration – Ideal Way

Approach-3: Make two types application servers.

Upgraded tenants use the new application, other tenants use the old application

😊 Minimum wait time for a tenant
😊 Upgrading can be scheduled for tenants
😊 Run A/B tests and see bugs before anyone else
😡 Requires multiple app servers
😡 Hard to maintain and monitor

✓ Identifying the Active Tenant
✓ Data Isolation
✓ Set TenantId for New Entities
✓ DB Connection String Selection
✓ Changing the Active Tenant
✓ Temporarily Disable Multi-Tenancy
✓ Database Migration

Do You Need Multi-Tenancy?

abp.io

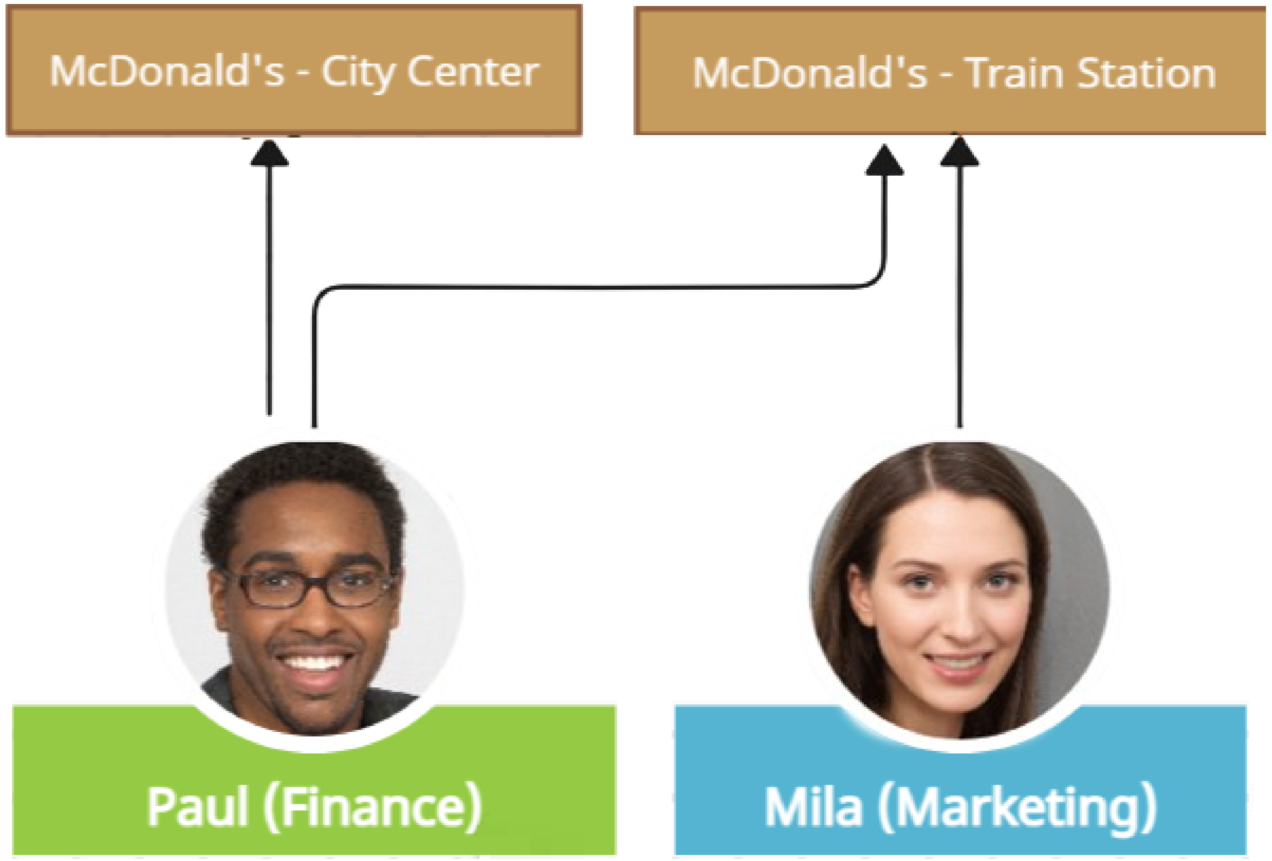
Do You Really Need Multi-Tenancy?

Multi-tenant development is **hard** - Reconsider!

1- Can a user be shared among other tenants?

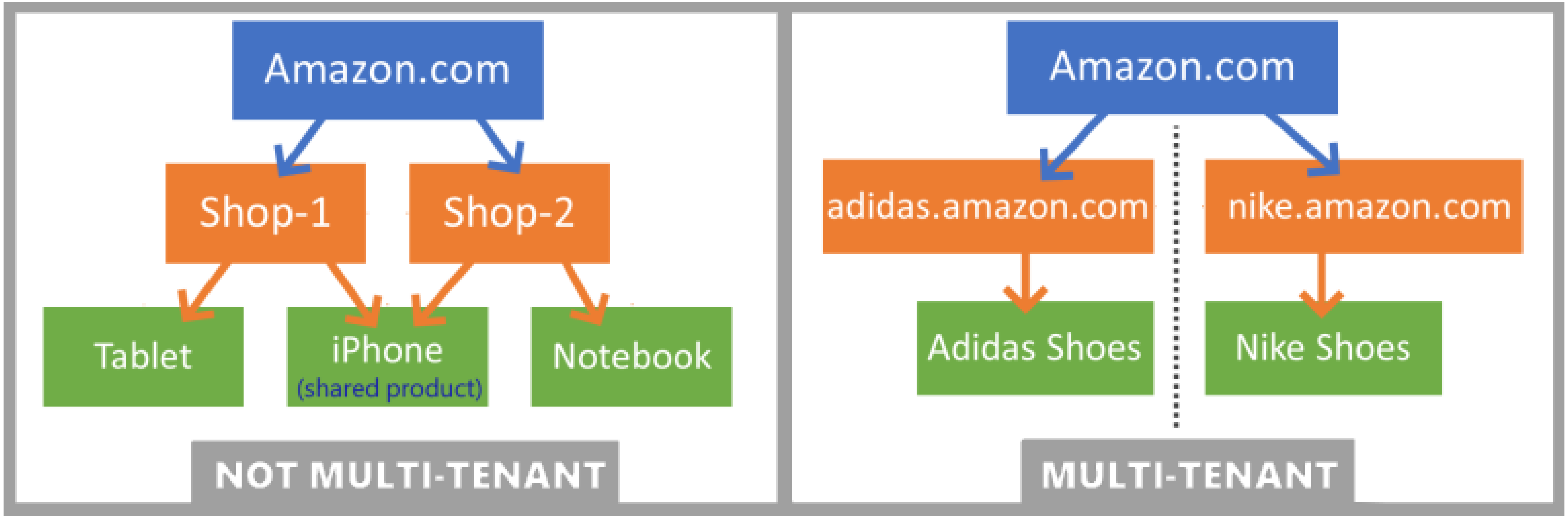
Our customer has branches in different cities, is it multi-tenant?

Our university has different faculties; should I make each faculty a different tenant?



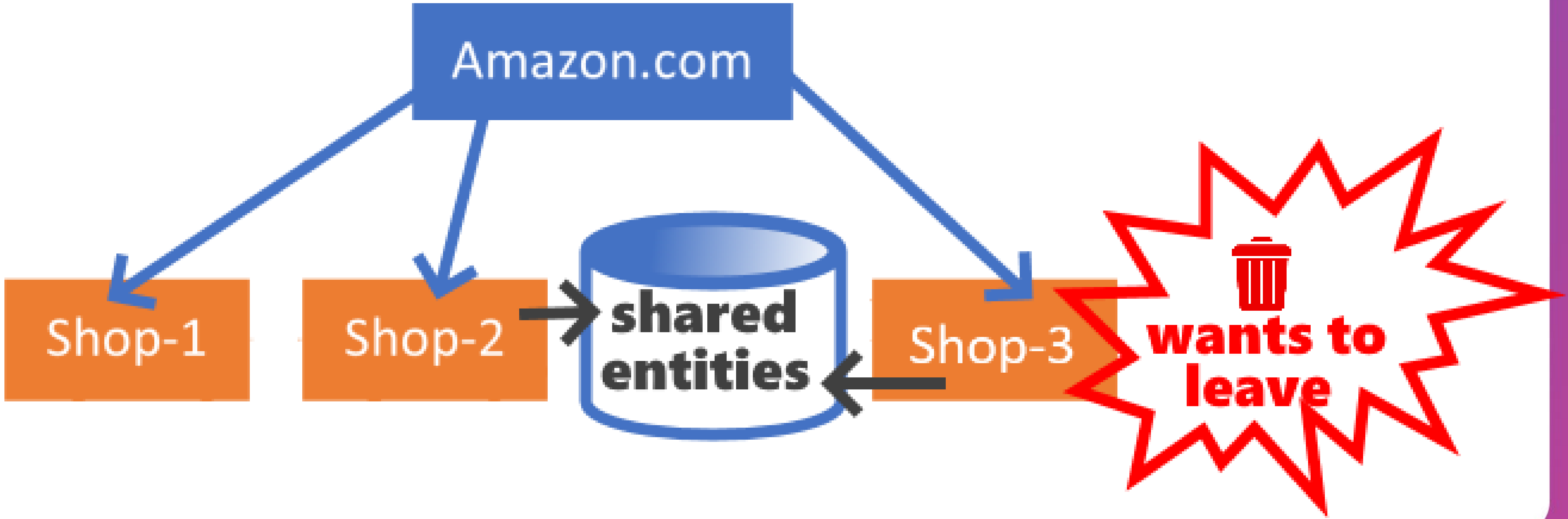
Do You Really Need Multi-Tenancy?

2- Any tenant needs to see the other tenant's data?



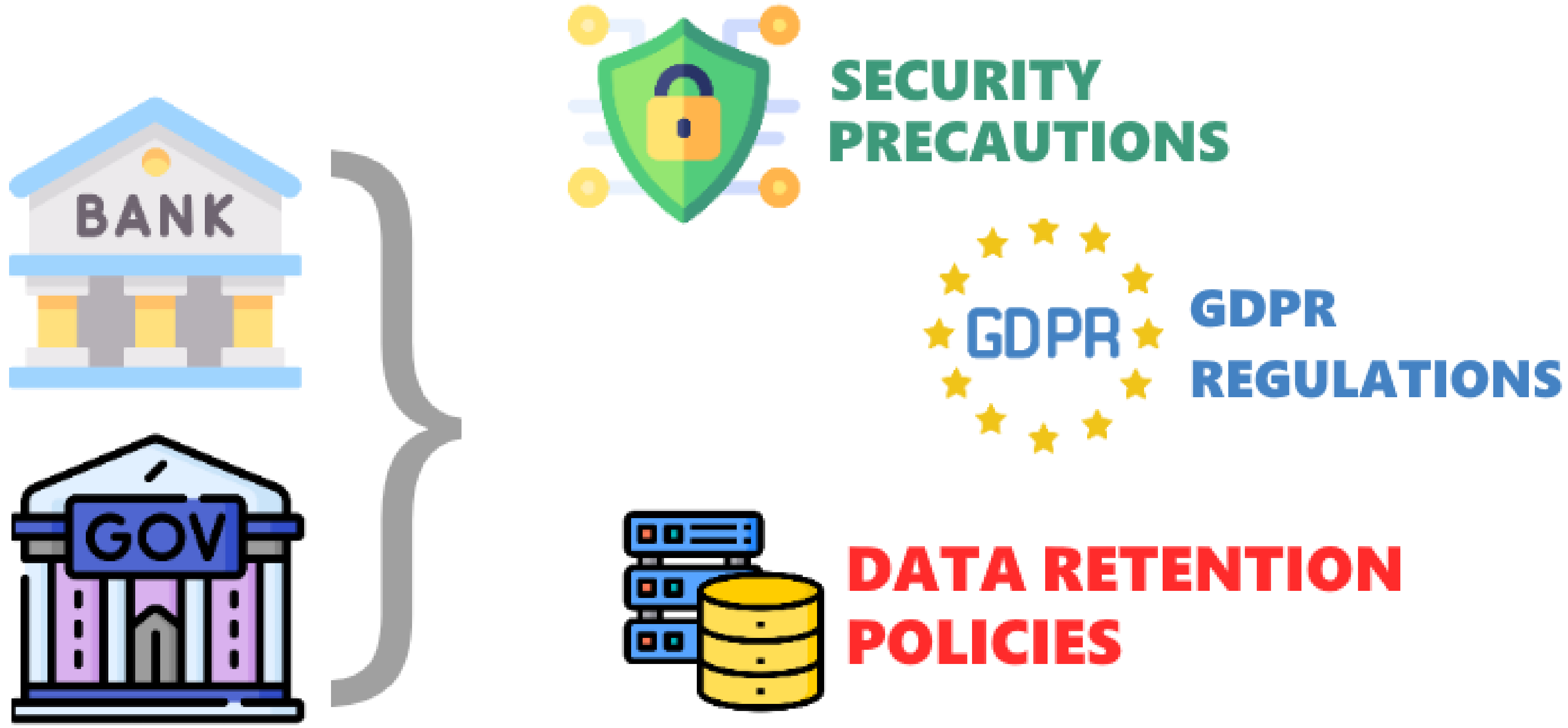
Do You Really Need Multi-Tenancy?

3- Does your application still work if you physically move one of the tenants physically?



Do You Really Need Multi-Tenancy?

4- Do your customers need higher security and data protection rules?



Thank you for joining ☺

https://twitter.com/**alperebicoglu**

https://github.com/**ebicoglu**

https://medium.com/**@alperonline**

Download this presentation:
https://github.com/ebicoglu/presentations

open-source web application framework
https://abp.io