Adding More Complex Features to the Site

Gill Cleeren

CTO Xebia Microsoft Services Belgium

@gillcleeren

Overview

Adding in-memory caching

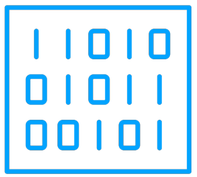
Editing a list of entities

Handling concurrency

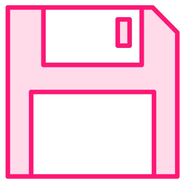
Adding In-memory Caching

Adding Caching

Improve performance



Keep data around for some time



Cached data can go stale

Different Types of Caching in ASP.NET Core

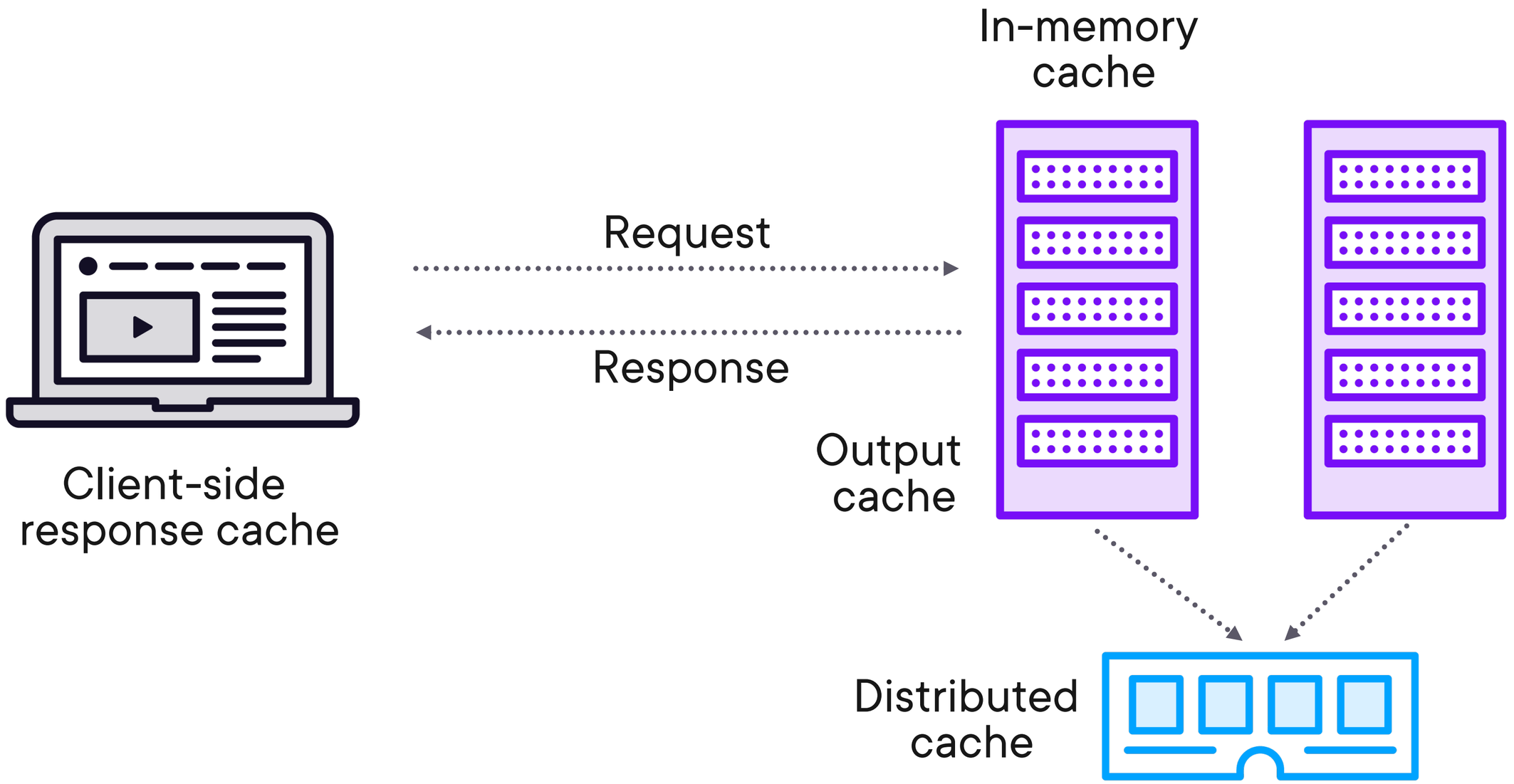
In-memory caching

Distributed cache

Response cache

Output cache

Understanding the Caching Types



Adding In-memory Caching

IMemoryCache interface

Simply stores data in memory of server

Can store all types of data

Store results of database call

Shared between all users by default

```
public class CategoryRepository : ICategoryRepository
{
    private readonly BethanysPieShopDbContext _bethanysPieShopDbContext;
    private IMemoryCache _memoryCache;

    public CategoryRepository
        (BethanysPieShopDbContext bethanysPieShopDbContext, IMemoryCache memoryCache)
    {
        _bethanysPieShopDbContext = bethanysPieShopDbContext;
        _memoryCache = memoryCache;
    }
}
```

Adding IMemoryCache using Dependency Injection

```
if (!_memoryCache.TryGetValue(AllCategoriesCacheName, out allCategories))
{
    allCategories = await
        _bethanysPieShopDbContext.Categories
        .AsNoTracking()
        .OrderBy(c => c.CategoryId).ToListAsync();

    var cacheEntryOptions =
        new MemoryCacheEntryOptions().SetSlidingExpiration(TimeSpan.FromSeconds(60));

    _memoryCache.Set(AllCategoriesCacheName, allCategories, cacheEntryOptions);
}
```

Using IMemoryCache

MemoryCacheEntryOptions can be used to configure how caching works

Demo

Adding IMemoryCache on the repository

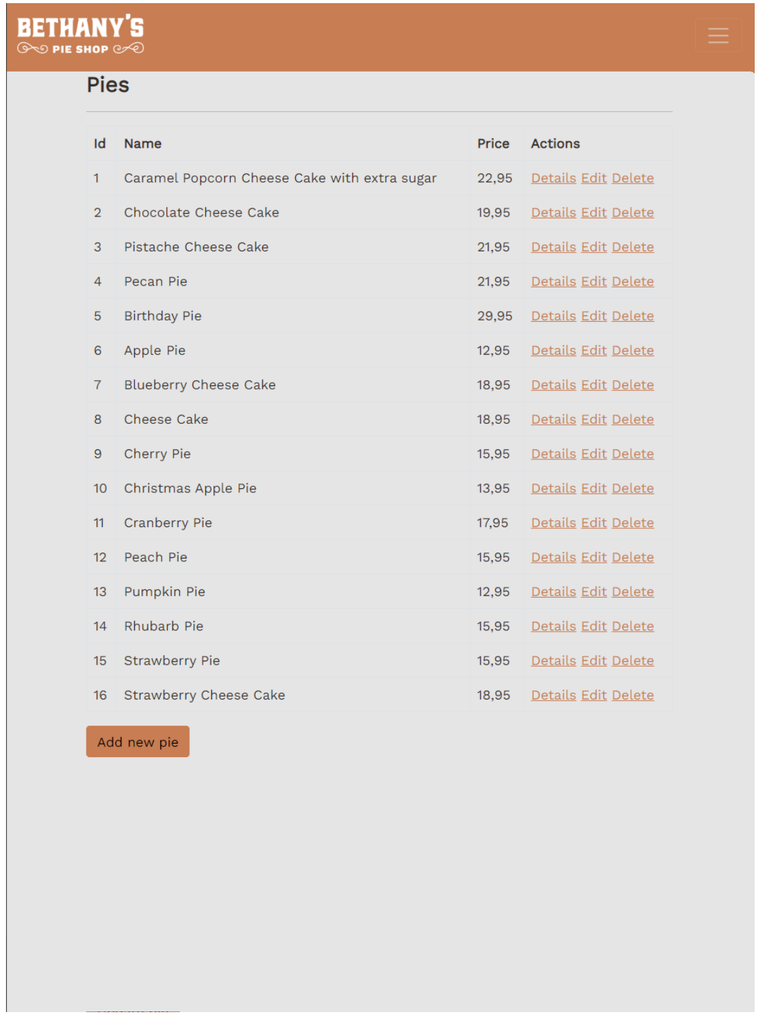
Editing a List of Entities

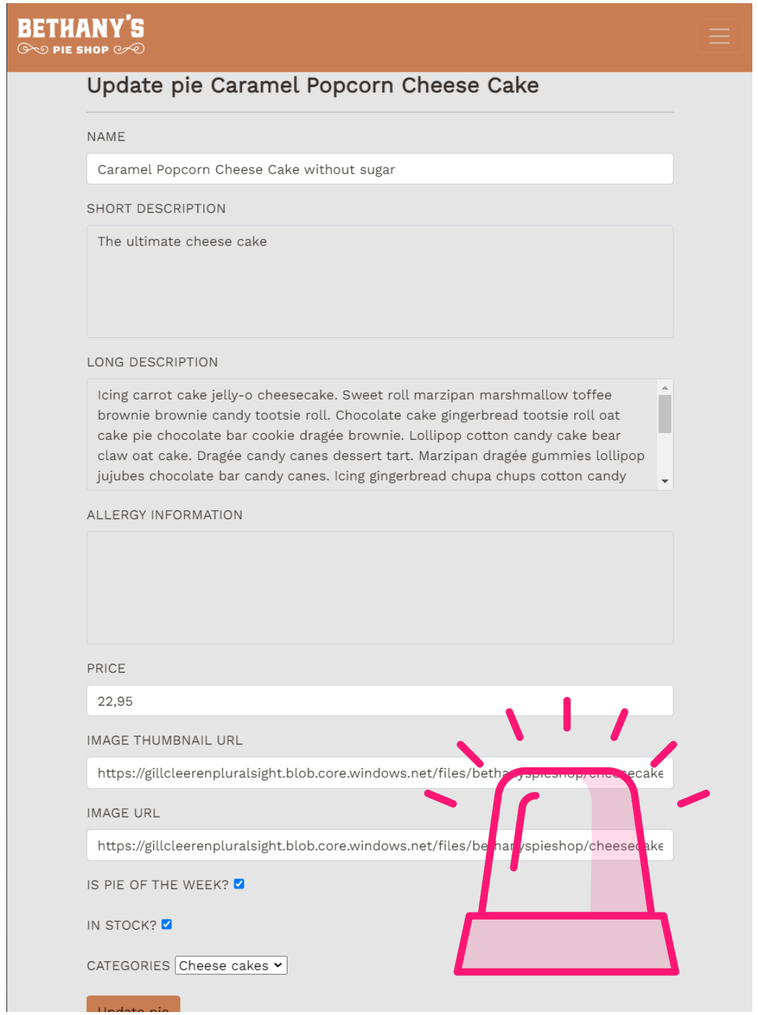
Demo

Editing a list

Handling Concurrency

Understanding the Problem We Face





Possible Approaches

Pessimistic concurrency

Optimistic concurrency

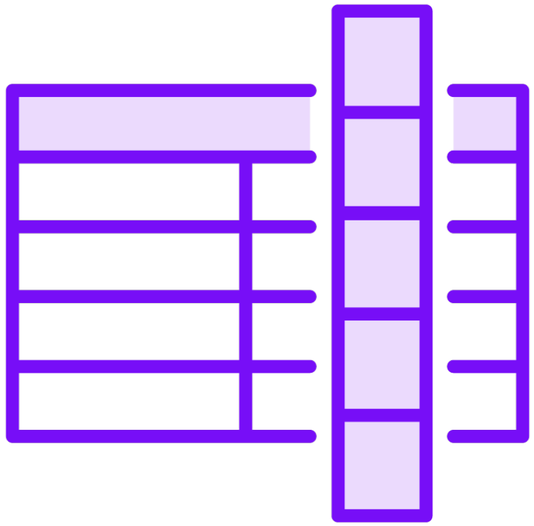
Possible Approaches

Update selection of columns

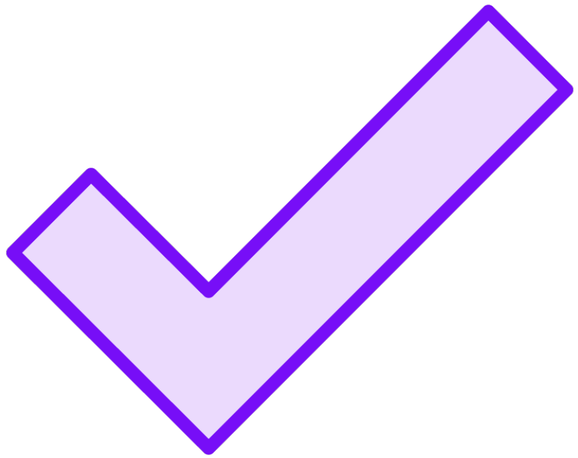
Overwrite data

Display errors or warnings

Detecting Conflicts



Tracking column
Sequential number



ConcurrencyCheck
attribute

```
public class Pie
{

    [Timestamp]
    public byte[]? RowVersion { get; set; }
}
```

Changing the OnModelCreating

Checking for Data Changes

```
try
{
    await _pieRepository.UpdatePieAsync(pie);
    return RedirectToAction(nameof(Index));
}
catch (DbUpdateConcurrencyException ex)
{
    ...
}
```

Demo

Handling concurrency

Summary

Caching data gives us lower load on the database – for free

Model binding works with lists too

Handling concurrency issues is often necessary

“Hey Bethany! The new administration site is ready. You can start using it!”

“Oh, that is awesome!!!

Thank you for the hard work! This will make managing our store so much easier”

Congratulations

on finishing this course!