Lab 5 Specification

Goal

Extract query logic used when sending notifications into a *Specification*.

Topics Used

Specification

Requirements

The Guestbook currently sends an email to the person who last left a message, in addition to others. This is a bug.

Pull the filtering logic into its own type where it can be reused and tested.

Detailed Steps

- Create `ISpecification<T>` in Core/Interfaces
  - One method: `Expression<Func<T, bool>> Criteria { get; }`
- Extract the querying/filtering logic from *GuestbookNotificationHandler*
  - Put it into a new *GuestbookNotificationPolicy* class
- Have `GuestbookNotificationPolicy` implement `ISpecification<GuestbookEntry>` and implement the `Criteria` property
- Add a `List<GuestbookEntry> ListEntries(Specification<GuestbookEntry> spec)` method to `GuestbookRepository`
- Add the method to a new `IGuestbookRepository` interface that inherits from `IRepository<Guestbook>`
- Add unit tests to confirm the criteria returns any entries within the last day
- Add unit tests to confirm the criteria does not return the entry that triggered the notification
  - You can pass in the `Id` of the entry to omit to the constructor of `GuestbookNotificationPolicy`

Also (not used immediately): - Add a `List(Specification<T> spec)` method to `IRepository<T>` - Implement this new `List` method in Infrastructure/EfRepository

Note: The `.Include()` method doesn't support filtering, which is why in this case we're adding another repository method to get entries separately from `Guestbook`. There are other approaches one could take to achieve this, but this is one of the simplest. Alternately, you could pass in a specification to the `GetById` method when fetching a `Guestbook` and use it to manually fetch its entries and add them to the instance.

Example: *GuestbookNotificationPolicy*

```
public class GuestbookNotificationPolicy : ISpecification<GuestbookEntry>
{
    private readonly int _entryId;

    public GuestbookNotificationPolicy(int entryId)
    {
        _entryId = entryId;
    }

    public Expression<Func<GuestbookEntry, bool>> Criteria {
        get
        {
            return e =>
                e.DateTimeCreated > DateTimeOffset.UtcNow.AddDays(-1)
                && e.Id != _entryId;
        }
    }
}
```

Example: *IGuestbookRepository*

```
public interface IGuestbookRepository : IRepository<Guestbook>
{
    List<GuestbookEntry> ListEntries(ISpecification<GuestbookEntry> spec);
}
```

Example: *GuestbookRepository*

```
public class GuestbookRepository : EfRepository<Guestbook>, IGuestbookRepository
{
    public GuestbookRepository(AppDbContext dbContext) : base(dbContext)
    {
    }

    // Since Guestbook is an Aggregate Root we need it to include its children
    public override Guestbook GetById(int id)
    {
        return _dbContext.Guestbooks
            .Include(g => g.Entries)
            .FirstOrDefault(g => g.Id == id);
    }

    public List<GuestbookEntry> ListEntries(ISpecification<GuestbookEntry> spec)
    {
        return _dbContext.GuestbookEntries
            .Where(spec.Criteria)
            .ToList();
    }
}
```