# Section 14 Pagination challenge

Your challenge is to implement pagination for the new feature we have just added. These are the steps you can take to implement this kind of functionality.

1. Create a LikesParam.cs in the Helpers folder in the API project that derives from the PagingParams class
2. Update the GetMemberLikes() method in the ILikesRepository and its implementation class to return a PaginatedResultMember> for this method. Tip: Create a variable outside of the switch statement to store the query and the result which you can then use outside of the switch statement e.g:

```
        var query = context.Likes.AsQueryable();
        IQueryable<Member> result;

        switch (likesParams.Predicate)
        {
            case "liked":
                result = query
                    .Where(like => like.SourceMemberId == likesParams.UserId)
                    .Select(like => like.TargetMember);
                break;
```

3. Update the controller method for the GetMemberLikes() action to use the new LikesParams.cs
4. Test this in Postman and ensure you get the pagination metadata returned with the request.
5. Update the likes-service.ts so that we can send up the pagination params as well as the predicate for the getLikes() method.
6. Update the lists.ts component so that we can use the paginated response for the members we return. You will need to create an 'onPageChange()' method for use in the Paginator component we created earlier.
7. Update the lists.html template to include the paginator component and ensure that this works as expected.