1.Install MediatR nuget package.

builder.Services.AddMediatR(config => config.RegisterServicesFromAssemblies(typeof(Program).Assembly));

// Register MediatR services for the current assembly

2. Creating Mockdata MFakeDataStore.cs

builder.Services.AddSingleton<MFakeDataStore>();

// Register the MFakeDataStore as a singleton service

3.Create 3 folders Queries, Commands & Handlers

In Queries folder create a GetMProductQuery class

public record GetMProductQuery:IRequest<IEnumerable<MProduct>>;

In Handlers folder

public class GetMProductsHandler:

IRequestHandler<GetMProductQuery, IEnumerable<MProduct>>

{

private readonly MFakeDataStore \_dataStore;

public GetMProductsHandler(MFakeDataStore dataStore)

{

\_dataStore = dataStore;

}

public async Task<IEnumerable<MProduct>>

Handle(GetMProductQuery request, CancellationToken cancellationToken)

=> await \_dataStore.GetAllMProducts();

}

In Controller class

private readonly ISender \_sender;

public MProductController(ISender sender)

{

\_sender = sender;

}

[HttpGet]

public async Task<IActionResult> GetMProducts()

{

var products = await \_sender.Send(new GetMProductQuery());

return Ok(products);

}

4. We can create as may as classes in the Queries folder for example getproducts by ID

In Queries:

public record GetMProductbyIDQuery(int ID):IRequest<MProduct>;

In Handlers:

public class GetMProductByHandler:

IRequestHandler<GetMProductbyIDQuery, MProduct>

{

private readonly MFakeDataStore \_dataStore;

public GetMProductByHandler(MFakeDataStore dataStore)

{

\_dataStore = dataStore;

}

public async Task<MProduct> Handle(GetMProductbyIDQuery request, CancellationToken cancellationToken)

=> await \_dataStore.GetMProductbyID(request.ID);

}

In Controller:

[HttpGet("{ID}",Name ="productbyId")]

public async Task<IActionResult> GetMProductByID(int ID)

{

var products = await \_sender.Send(new GetMProductbyIDQuery(ID));

return Ok(products);

}