



TABLE OF CONTENTS

1	PROJECT CONFIGURATION	1
1.1	Creating a New Project	1
1.2	launchSettings.json File Configuration	3
1.3	Program.cs Class Explanations.....	4
1.4	Extension Methods and CORS Configuration.....	8
1.5	IIS Configuration	9
1.6	Additional Code in the Program Class.....	11
1.7	Environment-Based Settings	12
1.8	ASP.NET Core Middleware	14
1.8.1	Creating a First Middleware Component.....	17
1.8.2	Working with the Use Method.....	19
1.8.3	Using the Map and MapWhen Methods	21
1.8.4	Using MapWhen Method	22
2	CONFIGURING A LOGGING SERVICE	24
2.1	Creating the Required Projects	24
2.2	Creating the ILoggerManager Interface and Installing NLog	25
2.3	Implementing the Interface and Nlog.Config File.....	27
2.4	Configuring Logger Service for Logging Messages	28
2.5	DI, IoC, and Logger Service Testing	30
3	ONION ARCHITECTURE IMPLEMENTATION	32
3.1	About Onion Architecture	33
3.1.1	Advantages of the Onion Architecture	34



3.1.2	Flow of Dependencies.....	34
3.2	Creating Models	35
3.3	Context Class and the Database Connection.....	37
3.4	Migration and Initial Data Seed.....	40
3.5	Repository Pattern Logic	43
3.6	Repository User Interfaces and Classes	45
3.7	Creating a Repository Manager	46
3.8	Adding a Service Layer.....	49
3.9	Registering RepositoryContext at a Runtime.....	52
4	HANDLING GET REQUESTS	54
4.1	Controllers and Routing in WEB API.....	54
4.2	Naming Our Resources.....	59
4.3	Getting All Companies From the Database	59
4.4	Testing the Result with Postman.....	63
4.5	DTO Classes vs. Entity Model Classes	65
4.6	Using AutoMapper in ASP.NET Core.....	68
5	GLOBAL ERROR HANDLING	72
5.1	Handling Errors Globally with the Built-In Middleware.....	72
5.2	Program Class Modification	74
5.3	Testing the Result	75
6	GETTING ADDITIONAL RESOURCES	77



6.1	Getting a Single Resource From the Database.....	77
6.1.1	Handling Invalid Requests in a Service Layer	79
6.2	Parent/Child Relationships in Web API	82
6.3	Getting a Single Employee for Company	85
7	CONTENT NEGOTIATION	89
7.1	What Do We Get Out of the Box?.....	89
7.2	Changing the Default Configuration of Our Project	90
7.3	Testing Content Negotiation.....	91
7.4	Restricting Media Types	93
7.5	More About Formatters	94
7.6	Implementing a Custom Formatter	95
8	METHOD SAFETY AND METHOD IDEMPOTENCY	98
9	CREATING RESOURCES	100
9.1	Handling POST Requests	100
9.2	Code Explanation	103
9.2.1	Validation from the ApiController Attribute.....	104
9.3	Creating a Child Resource	107
9.4	Creating Children Resources Together with a Parent	110
9.5	Creating a Collection of Resources	111
9.6	Model Binding in API.....	117
10	WORKING WITH DELETE REQUESTS.....	121
10.1	Deleting a Parent Resource with its Children	123



11	WORKING WITH PUT REQUESTS	125
11.1	Updating Employee	125
11.1.1	About the Update Method from the RepositoryBase Class	129
11.2	Inserting Resources while Updating One	129
12	WORKING WITH PATCH REQUESTS	132
12.1	Applying PATCH to the Employee Entity	133
13	VALIDATION	140
13.1	ModelState, Rerun Validation, and Built-in Attributes	140
13.1.1	Rerun Validation	141
13.1.2	Built-in Attributes	142
13.2	Custom Attributes and IValidatableObject	143
13.3	Validation while Creating Resources	145
13.3.1	Validating Int Type	148
13.4	Validation for PUT Requests	149
13.5	Validation for PATCH Requests	151
14	ASYNCHRONOUS CODE	156
14.1	What is Asynchronous Programming?	156
14.2	Async, Await Keywords, and Return Types	158
14.2.1	Return Types of the Asynchronous Methods	160
14.2.2	The IRepositoryBase Interface and the RepositoryBase Class Explanation	161
14.3	Modifying the ICompanyRepository Interface and the CompanyRepository Class	161
14.4	IRepositoryManager and RepositoryManager Changes	162
14.5	Updating the Service layer	163



14.6	Controller Modification	165
14.7	Continuation in Asynchronous Programming	168
14.8	Common Pitfalls	169
15	ACTION FILTERS	171
15.1	Action Filters Implementation.....	171
15.2	The Scope of Action Filters	172
15.3	Order of Invocation	173
15.4	Improving the Code with Action Filters	175
15.5	Validation with Action Filters	175
15.6	Refactoring the Service Layer	178
16	PAGING	183
16.1	What is Paging?	183
16.2	Paging Implementation.....	184
16.3	Concrete Query	187
16.4	Improving the Solution	189
16.4.1	Additional Advice	192
17	FILTERING	194
17.1	What is Filtering?	194
17.2	How is Filtering Different from Searching?.....	195
17.3	How to Implement Filtering in ASP.NET Core Web API	196
17.4	Sending and Testing a Query.....	198



18	SEARCHING	201
18.1	What is Searching?.....	201
18.2	Implementing Searching in Our Application	201
18.3	Testing Our Implementation	203
19	SORTING	206
19.1	What is Sorting?.....	206
19.2	How to Implement Sorting in ASP.NET Core Web API	208
19.3	Implementation – Step by Step.....	210
19.4	Testing Our Implementation	212
19.5	Improving the Sorting Functionality	213
20	DATA SHAPING	215
20.1	What is Data Shaping?	215
20.2	How to Implement Data Shaping	216
20.3	Step-by-Step Implementation	218
20.4	Resolving XML Serialization Problems.....	223
21	SUPPORTING HATEOAS	226
21.1	What is HATEOAS and Why is it so Important?.....	226
21.1.1	Typical Response with HATEOAS Implemented	227
21.1.2	What is a Link?.....	227
21.1.3	Pros/Cons of Implementing HATEOAS	228
21.2	Adding Links in the Project	229
21.3	Additional Project Changes	231



21.4	Adding Custom Media Types	232
21.4.1	Registering Custom Media Types	233
21.4.2	Implementing a Media Type Validation Filter	234
21.5	Implementing HATEOAS	235
22	WORKING WITH OPTIONS AND HEAD REQUESTS	243
22.1	OPTIONS HTTP Request	243
22.2	OPTIONS Implementation	243
22.3	Head HTTP Request	245
22.4	HEAD Implementation	245
23	ROOT DOCUMENT	247
23.1	Root Document Implementation	247
24	VERSIONING APIS	252
24.1	Required Package Installation and Configuration	252
24.2	Versioning Examples	253
24.2.1	Using Query String	255
24.2.2	Using URL Versioning	256
24.2.3	HTTP Header Versioning	257
24.2.4	Deprecating Versions	258
24.2.5	Using Conventions	259
25	CACHING	260
25.1	About Caching	260
25.1.1	Cache Types	260
25.1.2	Response Cache Attribute.....	261
25.2	Adding Cache Headers	261
25.3	Adding Cache-Store	263



25.4	Output Caching	266
25.4.1	Differences Between Output and Response Caching	266
25.5	Using Output Caching In Our App.....	267
25.5.1	Using Policies With Output Caching	269
25.5.2	Output Cache Keys	271
25.5.3	Caching Revalidation.....	272
26	RATE LIMITING AND THROTTLING	276
26.1	Implementing Rate Limiting.....	277
26.1.1	Rejection Configuration	278
26.1.2	Rate Limiter Queues	279
26.1.3	Policies With Rate Limiting	280
27	JWT, IDENTITY, AND REFRESH TOKEN	283
27.1	Implementing Identity in ASP.NET Core Project.....	283
27.2	Creating Tables and Inserting Roles.....	286
27.3	User Creation	287
27.4	Big Picture	293
27.5	About JWT.....	293
27.6	JWT Configuration.....	295
27.7	Protecting Endpoints.....	297
27.8	Implementing Authentication	298
27.9	Role-Based Authorization.....	303
28	REFRESH TOKEN.....	306
28.1	Why Do We Need a Refresh Token	308
28.2	Refresh Token Implementation	309



28.3	Token Controller Implementation	313
29	BINDING CONFIGURATION AND OPTIONS PATTERN.....	317
29.1	Binding Configuration	318
29.2	Options Pattern	320
29.2.1	Using IOption	321
29.2.2	IOptionSnapshot and IOptionMonitor.....	323
30	DOCUMENTING API WITH SWAGGER	326
30.1	About Swagger.....	326
30.2	Swagger Integration Into Our Project.....	327
30.3	Adding Authorization Support	331
30.4	Extending Swagger Configuration	334
31	DEPLOYMENT TO IIS	338
31.1	Creating Publish Files.....	338
31.2	Windows Server Hosting Bundle	340
31.3	Installing IIS.....	341
31.4	Configuring Environment File	344
31.5	Testing Deployed Application	345
32	BONUS 1 - RESPONSE PERFORMANCE IMPROVEMENTS.....	349
32.1	Adding Response Classes to the Project.....	349
32.2	Service Layer Modification	351
32.3	Controller Modification	353
32.4	Testing the API Response Flow	355



33	BONUS 2 - INTRODUCTION TO CQRS AND MEDIATR WITH ASP.NET CORE WEB API	358
33.1	About CQRS and Mediator Pattern.....	358
33.1.1	CQRS.....	358
33.1.2	Advantages and Disadvantages of CQRS	360
33.1.3	Mediator Pattern.....	361
33.2	How MediatR facilitates CQRS and Mediator Patterns	362
33.3	Adding Application Project and Initial Configuration.....	362
33.4	Requests with MediatR.....	365
33.5	Commands with MediatR.....	371
33.5.1	Update Command.....	373
33.5.2	Delete Command.....	375
33.6	MediatR Notifications	376
33.7	MediatR Behaviors	379
33.7.1	Adding Fluent Validation	380
33.7.2	Creating Decorators with MediatR PipelineBehavior	381
33.7.3	Validating null Object	386
