

## 3.4 HD

## Reverse Engineering

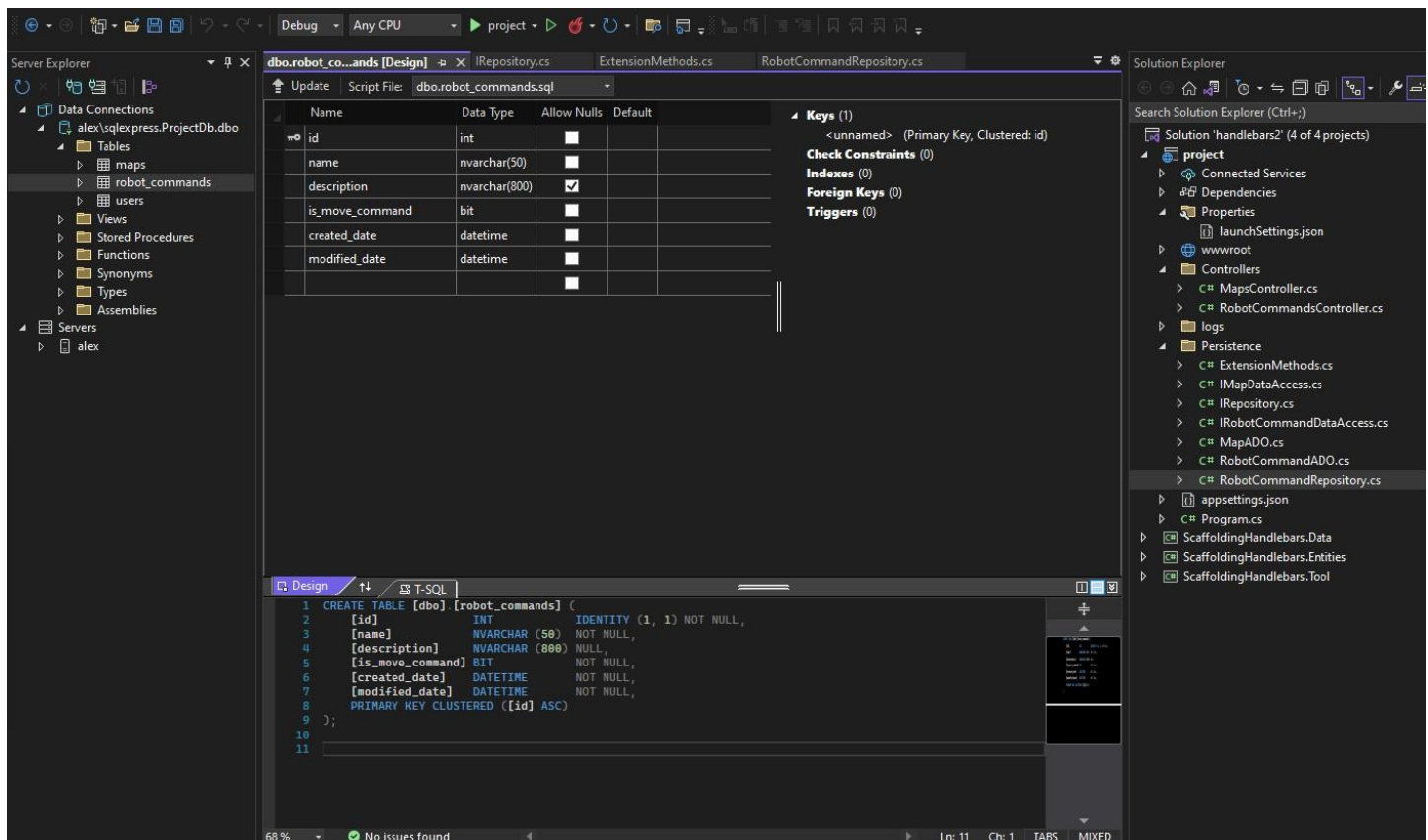
**Youtube:** <https://youtu.be/1sHUj8-4vv0>

**Github:** sent invite to Mr Slavnenko's deakin email

<https://github.com/alexbaar/SH-3.4-HD>

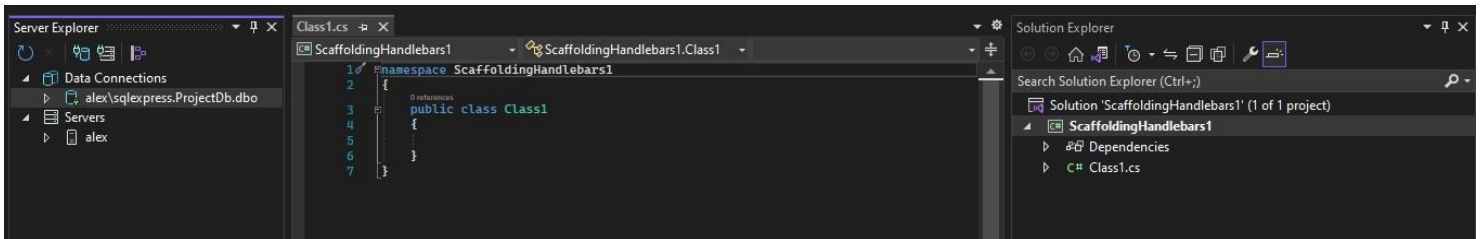
### Steps:

1. Create an empty project in VS 2022
2. Server explorer -> make a database connection (for me I connected to SQLQUERY10)
3. Create a database "ProjectDb"
4. Create tables for robot\_commands and maps (click 'Update') :

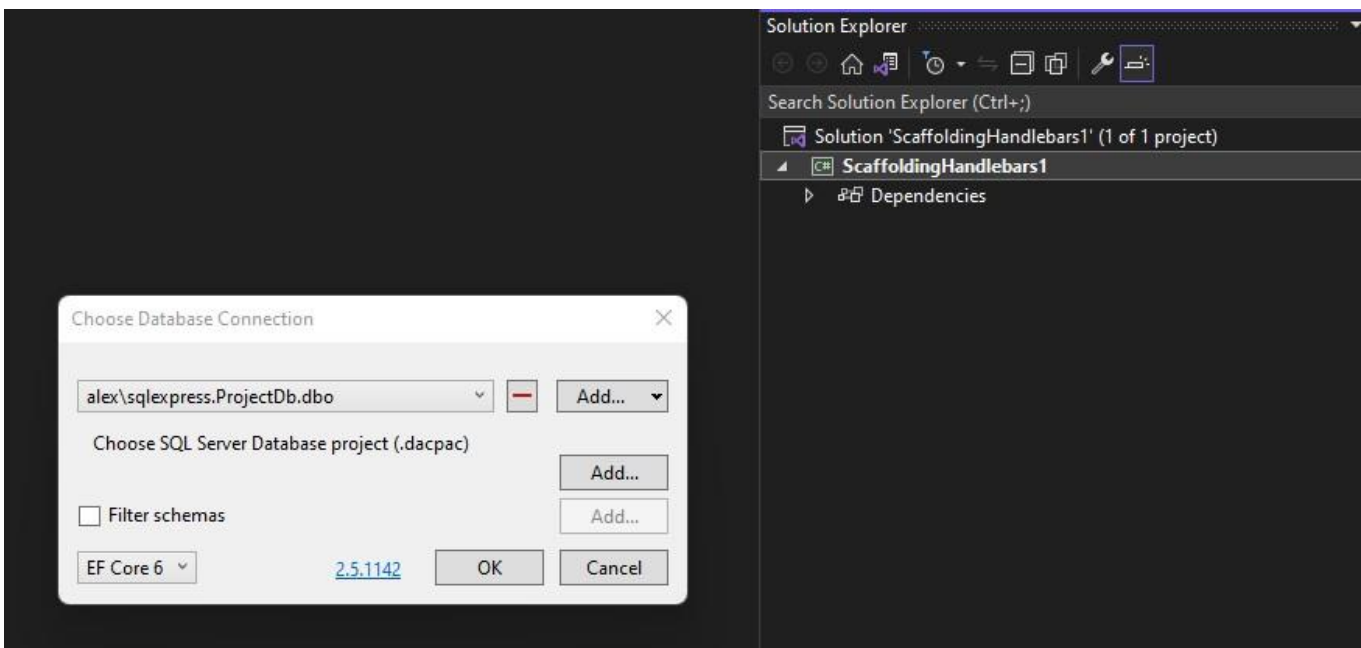


5. Check in SQLQUERY10 if they were created; if yes then
6. Follow the video from time: 24:50 where he creates a blank solution (.net standard class library)

<https://www.youtube.com/watch?v=6Ux7EpgiWXE>



7. Connect to the database created before
8. Install 'ee core power tools' extension that will help to reverse engineer easy way
9. Use that tool to reverse engineer the db (in the video); make sure the box "customize code using Handlebars templates" is ticked



Choose Database Objects



Search



Tables

OK

Cancel

Generate EF Core Model in Project ScaffoldingHandlebars1

Context name

Namespace

EntityType path (f.ex. Models) - optional

EntityType sub-namespace (overrides path) - optional

DbContext path (f.ex. Data) - optional

DbContext sub-namespace (overrides path) - optional

What to generate

Naming

☐ Pluralize or singularize generated object names (English)

☐ Use table and column names directly from the database

☐ Use DataAnnotation attributes to configure the model

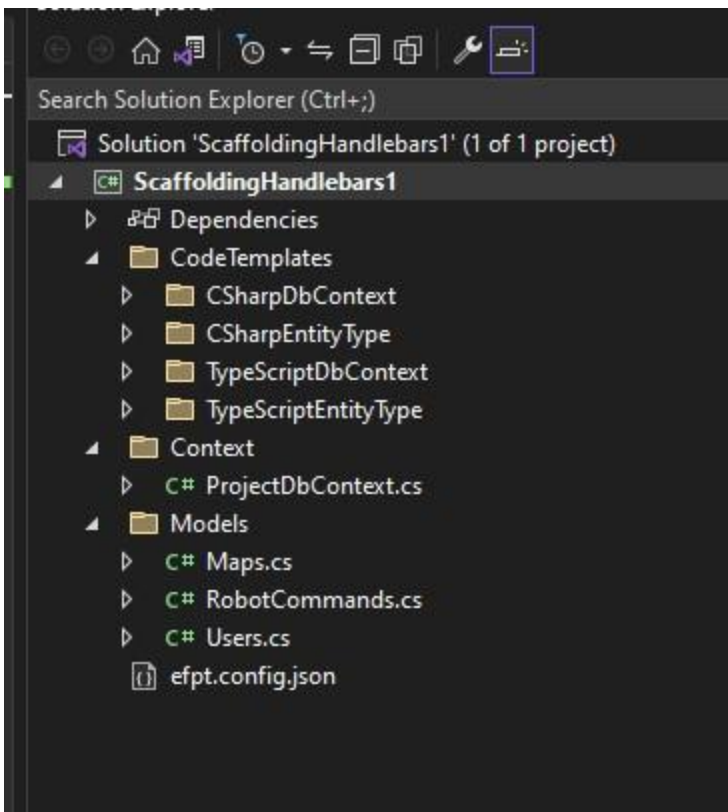
☒ Customize code using templates

☐ Include connection string in generated code

☒ Install the EF Core provider package in the project

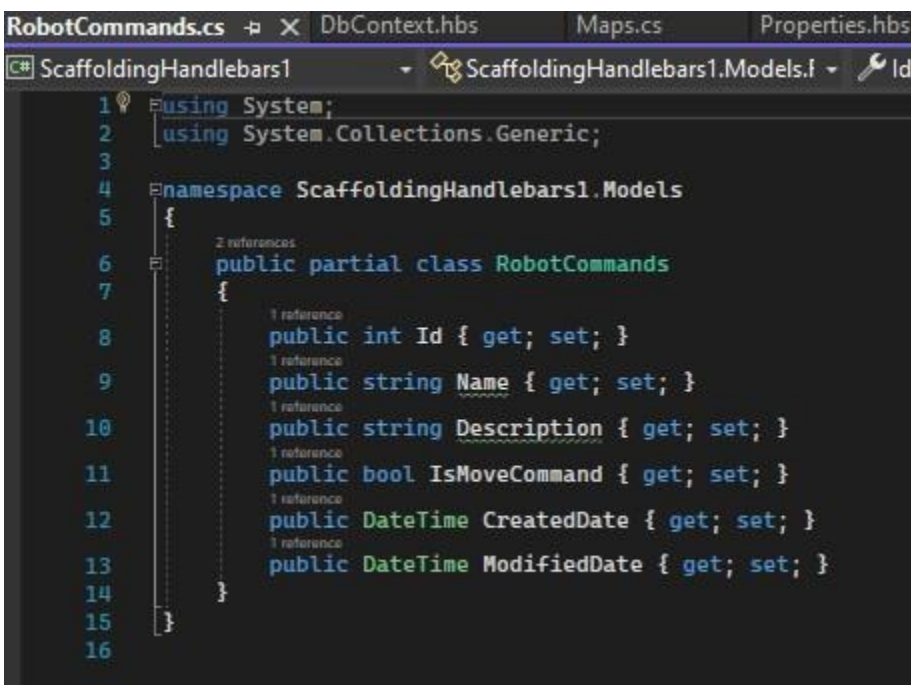
[Help](#) [★ Rate](#)

10.This way we get our dbcontext and models



11.To satisfy the #1 task requirement “ generated classes should not be partial I have modified the code templates ('CSharpEntityType-> Class.hbs and deleted the 'partial' word in line 13)

BEFORE:



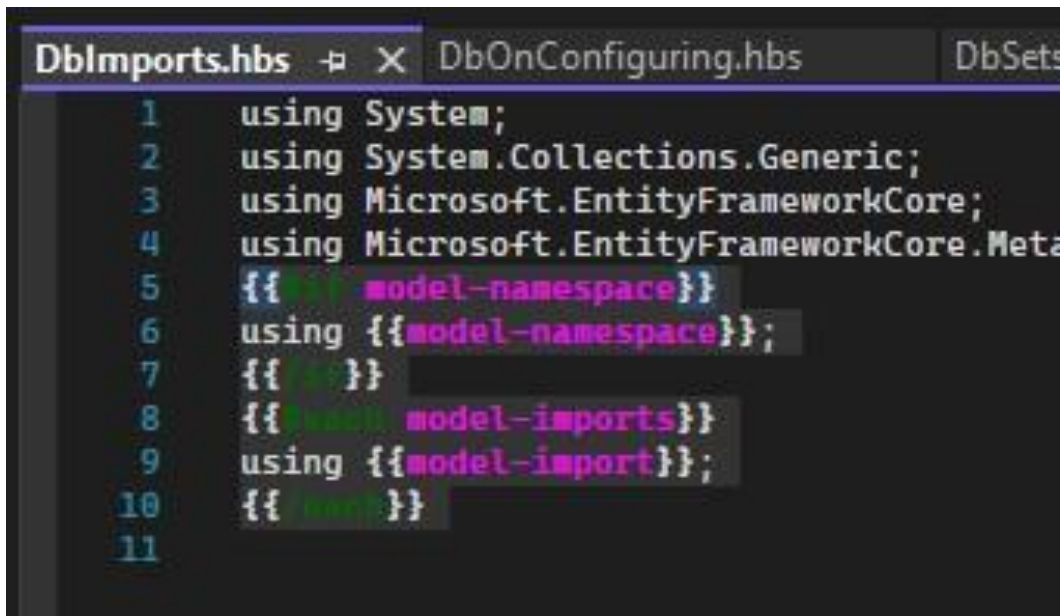
AFTER deleting 'partial' in line 13 and reverse engineer again :

```
Class.hbs  ▢ X
1  {{> imports}}
2
3  namespace {{namespace}}
4  {
5      {{#if comment}}
6      /// <summary>
7      {{comment}}
8      /// </summary>
9      {{/if}}
10     {{#each class-annotations}}
11     {{{class-annotation}}}
12     {{/each}}
13     public partial class {{class}}
14     {
15         {{{> constructor}}}
16         {{{> properties}}}
17     }
18 }
19
```

We get:

```
RobotCommands.cs  ▢ X  ProjectDbContext.cs  PowerToolsReadMe.txt
C:\Users\61424\source\repos\handlebars1\ScaffoldingHandlebars1\Models\RobotCom
1  using System;
2  using System.Collections.Generic;
3
4  namespace ScaffoldingHandlebars1.Models
5  {
6      2 references
7      public class RobotCommands
8      {
9          1 reference
10         public int Id { get; set; }
11         1 reference
12         public string Name { get; set; }
13         1 reference
14         public string Description { get; set; }
15         1 reference
16         public bool IsMoveCommand { get; set; }
17         1 reference
18         public DateTime CreatedDate { get; set; }
19         1 reference
20         public DateTime ModifiedDate { get; set; }
21     }
22 }
```

12.Delete lines 5-10



```
DbImports.hbs  DbOnConfiguring.hbs  DbSets.hbs
1  using System;
2  using System.Collections.Generic;
3  using Microsoft.EntityFrameworkCore;
4  using Microsoft.EntityFrameworkCore.Metadata;
5  {{if model-namespace}}
6  using {{model-namespace}};
7  {{/if}}
8  {{if model-imports}}
9  using {{model-import}};
10 {{/if}}
11
```

13.Snake\_case



```

33     entity.Property(e => e.Description)
34         .HasMaxLength(800)
35         .HasColumnName("description");
36
37     entity.Property(e => e.ModifiedDate)
38         .HasColumnType("datetime")
39         .HasColumnName("modified_date");
40
41     entity.Property(e => e.Name)
42         .IsRequired()
43         .HasMaxLength(50)
44         .HasColumnName("name");
45
46     entity.Property(e => e.Rows).HasColumnName("rows");
47 };
48
49 modelBuilder.Entity<RobotCommands>(entity =>
50 {
51     entity.ToTable("robot_commands");
52
53     entity.Property(e => e.Id).HasColumnName("id");
54
55     entity.Property(e => e.CreatedDate)
56         .HasColumnType("datetime")
57         .HasColumnName("created_date");
58
59     entity.Property(e => e.Description)
60         .HasMaxLength(800)
61         .HasColumnName("description");
62
63     entity.Property(e => e.IsMoveCommand).HasColumnName("is_move_command")
64
65     entity.Property(e => e.ModifiedDate)
66         .HasColumnType("datetime")
67         .HasColumnName("modified_date");
68

```

14. Followed the video to create :

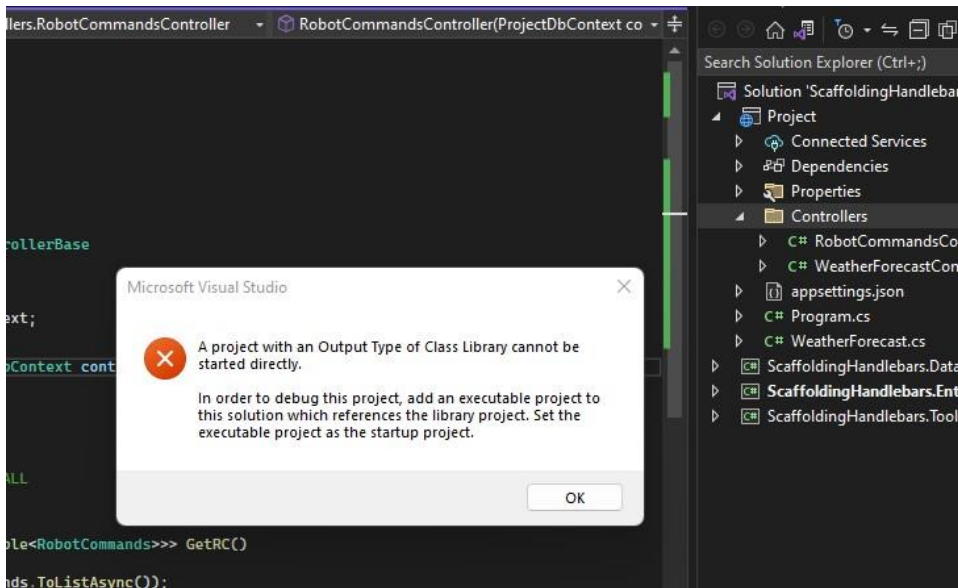
- ScaffoldingHandlebars.Data
- ScaffoldingHandlebars.Tool

15. In the same solution I added a new project 'ASP.NET Core Web API'

And added the controllers. Before I have added my Persistence layer I have used DbContext directly in my controller to see if it works.

16. Start up project error: rightclick 'Project' and set as startup project





<See Project “ScaffoldingHandlebars1” >

The below is <Project “handlebars2” >

17. Once I had that working I have fixed the Fast Member and ADO persistence layer implementations (also changed the IRepository.cs and ExtensionMethods.cs to allow SqlParameter, not Npgsql ones, and updated the connection string since I was not using postgres)

```

namespace project.Persistence
{
    2 references
    public interface IRepository
    {
        private const string CONNECTION_STRING = "Server=localhost\\sqlexpress;Database=ProjectDb;Trusted_connection=true;MultipleActiveResultSets=true";

        2 references
        public List<T> ExecuteReader<T>(string sqlCommand, SqlParameter[] dbParams = null) where T : class, new()
        {
            var entities = new List<T>();
            using var conn = new SqlConnection(CONNECTION_STRING);
            conn.Open();
            using var cmd = new SqlCommand(sqlCommand, conn);
            // Some of our SQL commands might have SQL parameters we will need to pass to DB.MS
            if (dbParams is not null)
            {
                // CommandType is unnecessary for PostgreSQL but can be used in other DB engines like Oracle or SQL Server. MS
                cmd.CommandType = CommandType.Text;
                cmd.Parameters.AddRange(dbParams.Where(x => x.Value is not null).ToArray());
            }
            using var dr = cmd.ExecuteReader();
            while (dr.Read())
            {
                var entity = new T();
                dr.MapTo(entity); // AUTOMATIC ORM WILL HAPPEN HERE. MapTo does not exist yet.MS; use .NET Extension Methods
                entities.Add(entity);
            }
            return entities;
        }
    }
}

```

```

1 using FastMember;
2 using Microsoft.Data.SqlClient;
3 using Npgsql;
4 namespace project.Persistence
5 {
    0 references
    6 public static class ExtensionMethods
    7 {
    8     // we create our own MapTo Method
    9     1 reference
    10     public static void MapTo<T>(this SqlDataReader dr, T entity)
    11     {
    12         if (entity == null) throw new
    13         ArgumentNullException(nameof(entity));
    14         var fastMember = TypeAccessor.Create(entity.GetType());
    15         var props = fastMember.GetMembers().Select(x =>
    16         x.Name).ToHashSet(StringComparer.OrdinalIgnoreCase);
    17         for (int i = 0; i < dr.FieldCount; i++)
    18         {
    19             var prop = props.FirstOrDefault(x =>
    20             x.Equals(dr.GetName(i), StringComparison.OrdinalIgnoreCase));
    21             if (!string.IsNullOrEmpty(prop))
    22                 fastMember[entity, prop] = dr.IsDBNull(i) ? null :
    23                 dr.GetValue(i);
    24         }
    25     }
    26 }
    27

```

18.The MapADO.cs and RobotCommandADO.cs were easy to implement with dbContext.

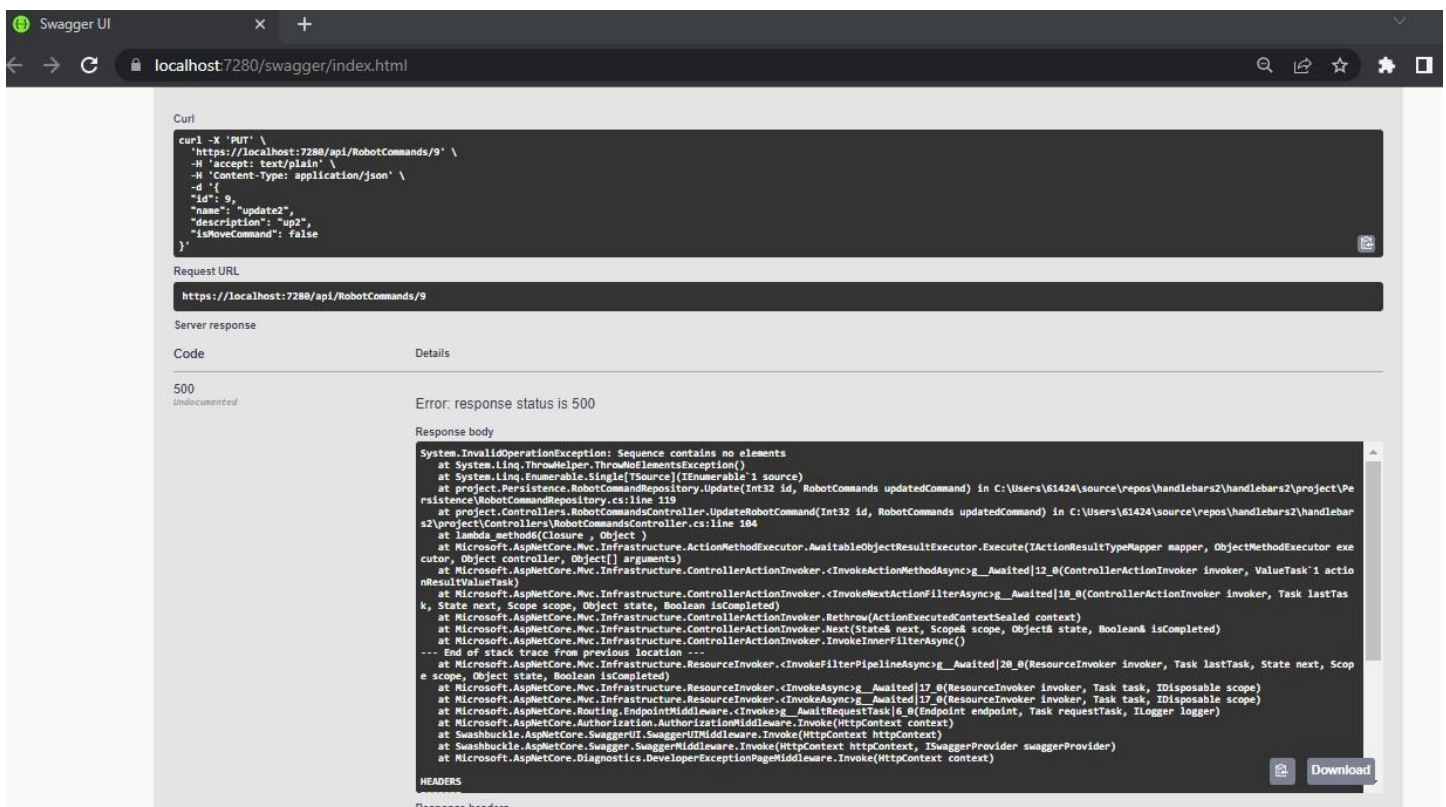
19.The RobotCommandRepository.cs was harder to adjust. Once I adjusted the Fast Member mapping, the challenges were:

- The Get All endpoint was easy :

```
// GET ALL
2 references
public List<RobotCommands> GetRobotCommands()
{
    var commands = _repo.ExecuteReader<RobotCommands>("SELECT * FROM robot_commands");

    return _context.RobotCommands.ToList();
}
```

- The update was bit harder as had to fix the connection, and then I would get an error “Sequence contains no elements”. The function still worked, meaning it did update the element, but the error was there:



But when changed from Single() to SingleOrDefault() the problem disappeared:

```

117         _context.SaveChanges();
118
119         var result = _repo.ExecuteReader<RobotCommands>(
120             commandText,
121             sqlParams)
122         // .Single()
123         .SingleOrDefault();
124         // changed from Single(); otherwise we get error: Sequence contains no elements
125
126         conn.Close();
127     }
128     return obj;
129
130

```

20. **Unresolved error** -> when trying to add method descriptions in Swagger UI

There is a path problem on the ScaffoldingHandlebars side that duplicates the path :

```

27 // Configure Swashbuckle to incorporate the XML comments on file into the generated Swagger JSON
28 options.SwaggerDoc("v1", new OpenApiInfo
29 {
30     Title = "Robot Controller API",
31     Description = "New backend service that provides resources for the Moon robot",
32     Contact = new OpenApiContact
33     {
34         Name = "Aleksandra Bartosiak",
35         Email = "abartosiak@deakin.edu.au"
36     },
37 });
38
39 // generate XML comments file and pass it into OPENAPI generator
40 var xmlFilename = $"{Assembly.GetExecutingAssembly().GetName().Name}.xml";
41 options.IncludeXmlComments(Path.Combine(AppContext.BaseDirectory, xmlFilename));
42

```

Exception User-Unhandled

System.IO.FileNotFoundException: 'Could not find file 'C:\Users\61424\source\repos\handlebars2\handlebars2\project\bin\Debug\net6.0\project.xml'.'

Exception was originally thrown at this call stack:

am.<Main>\$.AnonymousMethod\_0\_1(Swashbuckle.AspNetCore.SwaggerGen.Code) [Project: project.dll]

View Details | Copy Details | Start Live Share session...

Exception Settings

- ☒ Break when this exception type is user-unhandled
- Except when thrown from:
  - ☐ project.dll

Open Exception Settings | Edit Conditions

You can see '...\handlebars2\handlebars2\...' should only be 1

Exception User-Unhandled

System.IO.FileNotFoundException: 'Could not find file 'C:\Users\61424\source\repos\handlebars2\handlebars2\project\bin\Debug\net6.0\project.xml'.'

21. Video Links

- Scaff Handlebars Community

<https://www.youtube.com/watch?v=6Ux7EpgiWXE&t=1501s>

- Helpful Database 1<sup>st</sup> approach with Entity Framework Scaffolding

<https://www.youtube.com/watch?v=-yUsrUAUfHs&t=1999s>