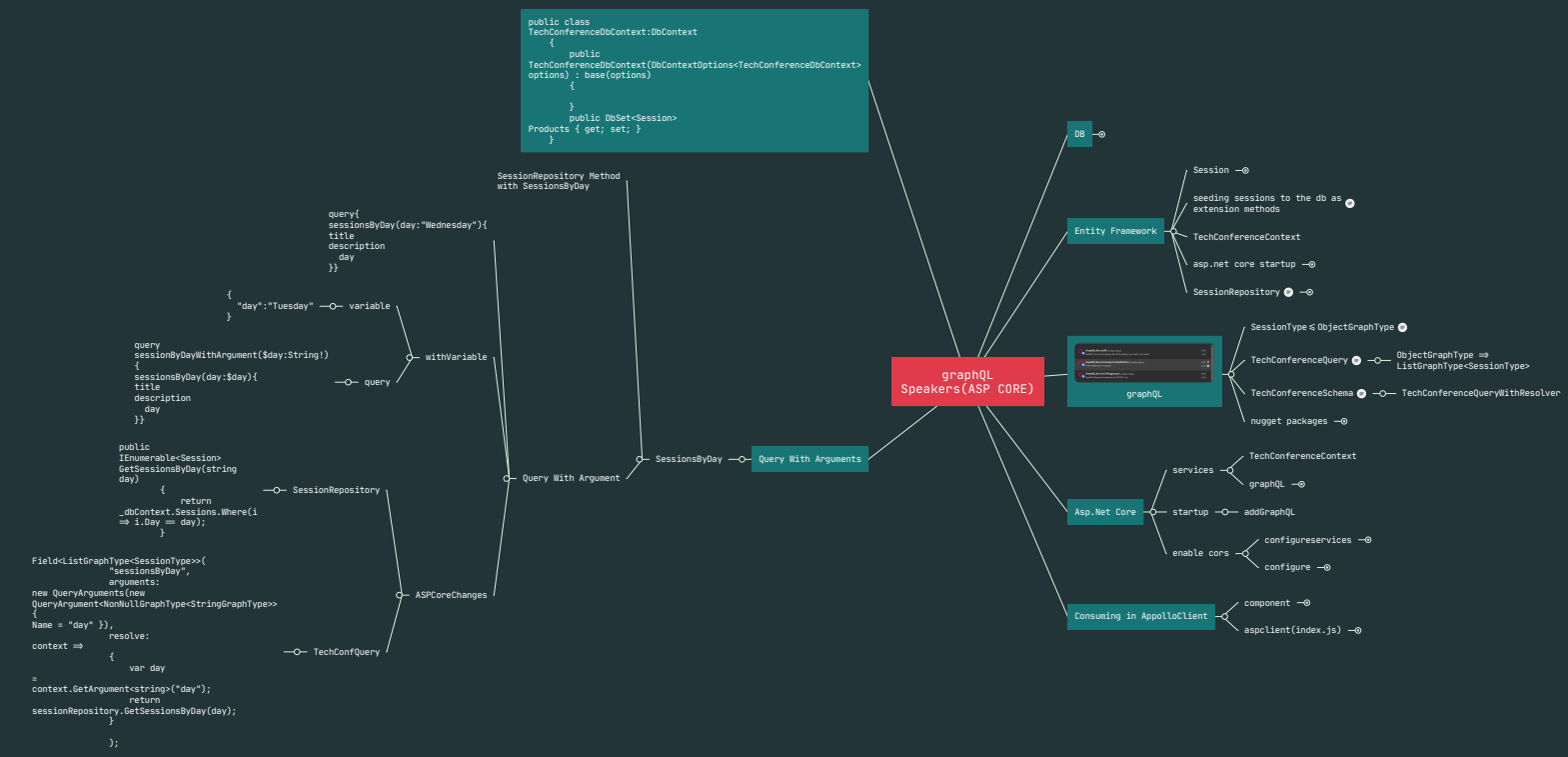
graphQL Speakers(ASP CORE)
DB
Entity Framework
Session
seeding sessions to the db as extension methods
TechConferenceContext
asp.net core startup
SessionRepository
graphQL
SessionType ⩽ ObjectGraphType
TechConferenceQuery
ObjectGraphType ⩾ ListGraphType<SessionType>
TechConferenceSchema
TechConferenceQueryWithResolver
nugget packages
Asp.Net Core
services
TechConferenceContext
graphQL
startup
addGraphQL
enable cors
configureservices
configure
Consuming in ApolloClient
component
aspclient(index.js)
Query With Arguments
SessionsByDay
Query With Argument
SessionRepository Method with SessionsByDay
ASPCoreChanges
SessionRepository
TechConfQuery
withVariable
query
variable
{ "day":"Tuesday" }
query{ sessionsByDay(day:"Wednesday"){ title description day }}
query sessionByDayWithArgument($day:String!) { sessionsByDay(day:$day){ title description day }}
public IEnumerable<Session> GetSessionsByDay(string day) { return _dbContext.Sessions.Where(i ⇒ i.Day == day); }
Field<ListGraphType<SessionType>>("sessionsByDay", arguments: new QueryArguments(new QueryArgument<NonNullGraphType<StringGraphType>> { Name = "day" }), resolve: context ⇒ { var day = context.GetArgument<string>("day"); return sessionRepository.GetSessionsByDay(day); });
public class TechConferenceDbContext:DbContext { public TechConferenceDbContext(DbContextOptions<TechConferenceDbContext> options) : base(options) { } public DbSet<Session> Products { get; set; } }

graphQL Speakers(ASP CORE)

1. DB

1.1. Tech Conferernce

2. Entity Framework

2.1. Session

2.1.1. sessions metadata

2.1.1.1. JSON

2.1.1.1.1. { "id": 84473, "title": "reactJS with graphQL", "description": "", "room": "Europa", "day": "Wednesday", "format": "FullDay Workshop", "track": ".NET", "level": "Intermediate" }

2.1.1.2. C# Entity

2.1.1.2.1. public class Session { public int Id { get; set; } [StringLength(100)] public string Title { get; set; } public string Description { get; set; } public string Room { get; set; } public string Day { get; set; } public string Format { get; set; } public string Level { get; set; } }

2.1.2. nuget

2.1.2.1. Microsoft.EntityFrameworkCore

2.1.2.1.1. v3.1.28

2.1.2.1.1.1. from nuget package console

2.1.2.2. Microsoft.EntityFrameworkCore.SqlServer

2.1.2.2.1. v3.1.28

2.1.2.2.1.1. from nuget package console

2.1.3. dbContext

2.1.3.1. references

2.1.3.1.1. using Microsoft.EntityFrameworkCore; using TechConference.Api.Data.Entities;

2.2. seeding sessions to the db as extension methods

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using CarvedRock.Api.Data.Entities;

namespace CarvedRock.Api.Data
{
    public static class InitialData
    {
        public static void Seed(this CarvedRockDbContext dbContext)
        {
            if (!dbContext.Products.Any())
            {
                dbContext.Products.Add(new Product
                {
                    Name = "Mountain Walkers",
                    Description = "Use these sturdy shoes to pass any mountain range with ease.",
                    Price = 219.5m,
                    Rating = 4,
                    Type = ProductType.Boots,
                    Stock = 12,
                    PhotoFileName = "shutterstock_66842440.jpg",
                    IntroducedAt = DateTimeOffset.Now.AddMonths(-1)
                });

                dbContext.Products.Add(new Product
                {
                    Name = "Army Slippers",
                    Description = "For your everyday marches in the army.",
                    Price = 125.9m,
                    Rating = 3,
                    Type = ProductType.Boots,
                    Stock = 56,
                    PhotoFileName = "shutterstock_222721876.jpg",
                    IntroducedAt = DateTimeOffset.Now.AddMonths(-1)
                });

                dbContext.Products.Add(new Product
                {
                    Name = "Backpack Deluxe",
                    Description = "This backpack can survive any tornado.",
                    Price = 199.99m,
                    Rating = 5,
                    Type = ProductType.ClimbingGear,
                    Stock = 66,
                    PhotoFileName = "shutterstock_6170527.jpg",
                    IntroducedAt = DateTimeOffset.Now.AddMonths(-1)
                });
```

```
            dbContext.Products.Add(new Product
            {
                Name = "Climbing Kit",
                Description = "Anything you need to climb the mount Everest.",
                Price = 299.5m,
                Rating = 5,
                Type = ProductType.ClimbingGear,
                Stock = 3,
                PhotoFileName = "shutterstock_48040747.jpg",
                IntroducedAt = DateTimeOffset.Now.AddMonths(-1)
            });

            dbContext.Products.Add(new Product
            {
                Name = "Blue Racer",
                Description = "Simply the fastest kayak on earth and beyond for 2 persons.",
                Price = 350m,
                Rating = 5,
                Type = ProductType.Kayaks,
                Stock = 8,
                PhotoFileName = "shutterstock_441989509.jpg",
                IntroducedAt = DateTimeOffset.Now.AddMonths(-1)
            });

            dbContext.Products.Add(new Product
            {
                Name = "Orange Demon",
                Description = "One person kayak with hyper boost button.",
                Price = 450m,
                Rating = 2,
                Type = ProductType.Kayaks,
                Stock = 1,
                PhotoFileName = "shutterstock_495259978.jpg",
                IntroducedAt = DateTimeOffset.Now.AddMonths(-1)
            });

            dbContext.SaveChanges();
        }
    }
  }
}
```

2.3. TechConferenceContext

2.4. asp.net core startup

2.4.1. in Services

2.4.1.1. register the TechConferenceContext with dbconnection

```
services.AddDbContext(options =>
        options.UseSqlServer(Configuration["ConnectionStrings:TechConference"]));
```

2.4.2. in configure

2.4.2.1. call initial db

2.5. SessionRepository

```
public class SessionRepository
    {
        public TechConferenceDbContext _dbContext;
        public SessionRepository(TechConferenceDbContext dbContext)
        {
            _dbContext = dbContext;
        }

        public IEnumerable GetSessions()
        {
            return _dbContext.Sessions;
        }

    }
```

2.5.1. GetAllSession

2.5.1.1. TechConferenceContext IOC

3. graphQL

3.1. SessionType<=ObjectGraphType

```
public class SessionType:ObjectGraphType
    {
        public SessionType()
        {
            Field(t => t.Id);
            Field(t => t.Title);
            Field(t => t.Description);
        }
    }
```

3.2. TechConferenceQuery

```
public class TechConferenceQuery:ObjectGraphType
    {
        public TechConferenceQuery(SessionRepository sessionRepository)
        {
            Field>(
                "sessions",
                resolve: context => sessionRepository.GetSessions()
                );
        }
    }
```

3.2.1. ObjectGraphType => ListGraphType<SessionType>

3.3. TechConferenceSchema

```
public class TechConferenceSchema :Schema
    {
        public TechConferenceSchema(IServiceProvider resolver): base(resolver)
        {
            Query = (IObjectGraphType)resolver.GetService(typeof(TechConferenceQuery));
        }
    }
```

3.3.1. TechConferenceQueryWithResolver

3.4. nugget packages

3.4.1. GraphQL.Server.All

3.4.1.1. v5.0.0

3.4.2. Graphql.Server.Transport.AspNetCore

3.4.2.1. v5.0.0

3.4.3. Graphql.Server.UI.Playground

3.4.3.1. v5.0.0

4. Asp.Net Core

4.1. services

4.1.1. TechConferenceContext

4.1.2. graphQL

4.1.2.1. IDependencyResolver

4.2. startup

4.2.1. addGraphQL

4.3. enable cors

4.3.1. configureservices

4.3.1.1. services.AddCors();

4.3.2. configure

4.3.2.1. app.UseCors(builder => builder.AllowAnyOrigin().AllowAnyHeader().AllowAnyMethod());

5. Consuming in AppolloClient

5.1. component

5.1.1. Sessions

```
import React from 'react'
import {gql,useQuery} from '@apollo/client'

const GET_SESSIONS = gql`
query GetSessions{
    sessions{
                id
    title
    description
  }
}
`
export default function Sessions() {
    const {data,error,loading} = useQuery(GET_SESSIONS);

welcome to appollo

    if(loading) return

loading...

     if(error) return

error

  return data.sessions.map(({id,title,description})=>(




{id}



{title}



    About this session:


{description}







  ))
}
```

5.1.1.1. Query

5.1.1.1.1. const GET_SESSIONS = gql` query GetSessions{ sessions{ id title description } } `

5.2. aspclient(index.js)

5.2.1. const aspclient = new ApolloClient({ uri: 'https://localhost:5001/graphql ', cache: new InMemoryCache(), });

6. Query With Arguments

6.1. SessionsByDay

6.1.1. SessionRepository Method with SessionsByDay

6.1.2. Query With Argument

6.1.2.1. query{ sessionsByDay(day:"Wednesday"){ title description day }}

6.1.2.2. withVariable

6.1.2.2.1. variable

6.1.2.2.1.1. { "day":"Tuesday" }

6.1.2.2.2. query

6.1.2.2.2.1. query sessionByDayWithArgument($day:String!) { sessionsByDay(day:$day){ title description day }}

6.1.2.3. ASPCoreChanges

6.1.2.3.1. SessionRepository

6.1.2.3.1.1. public IEnumerable<Session> GetSessionsByDay(string day) { return _dbContext.Sessions.Where(i => i.Day == day); }

6.1.2.3.2. TechConfQuery

6.1.2.3.2.1. Field<ListGraphType<SessionType>>("sessionsByDay", arguments: new QueryArguments(new QueryArgument<NonNullGraphType<StringGraphType>> { Name = "day" }), resolve: context => { var day = context.GetArgument<string>("day"); return sessionRepository.GetSessionsByDay(day); });

7. public class TechConferenceDbContext:DbContext { public TechConferenceDbContext(DbContextOptions<TechConferenceDbContext> options) : base(options) { } public DbSet<Session> Products { get; set; } }