MIDWEST ARCHITECTURE COMMUNITY COLLABORATION 2020

NOVEMBER 5, 2020

MACC 2020
Microservices Architecture

Practical guide to Microservices and their use in the real world

MACC MISSION

- The Midwest Architecture Community Collaboration's (MACC) purpose is to bring all domains of architecture together to share information and techniques of interest to all of us. It is our shared belief that through collaboration, we can better understand and promote the significance of architecture to business success.

midwest
architecture community
collaboration

midwest
architecture community
collaboration

SOA DESIGN AND DEVELOPMENT

- Service Oriented Architecture (SOA)

- .NET Remoting
- Data Synchronization
- WCF and SOAP protocol

- An then there was Roy Fielding…
- invented an architectural style called:
  "Representational State Transfer"

- …. Or what we all refer to now as.. the "RESTFul" API

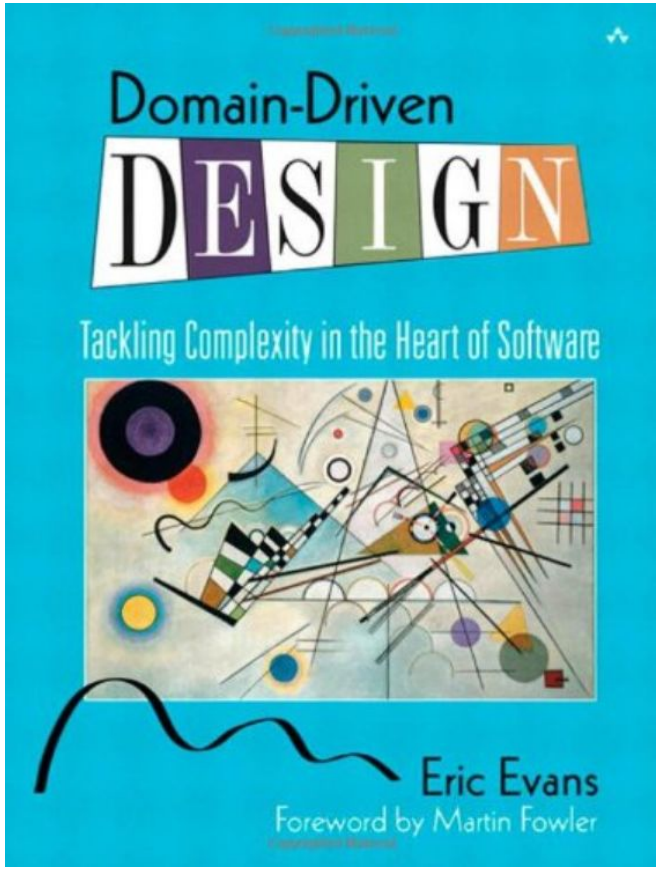
midwest
architecture community
collaboration

SOA AND DDD

DDD???

- Domain Driven Design
- Middle Tier Development First Approach
- Strong understanding of your business domain

- … Eric Evans Book on DDD



midwest architecture community collaboration

MICROSERVICES & DEVOPS

Microservices Architecture and Design

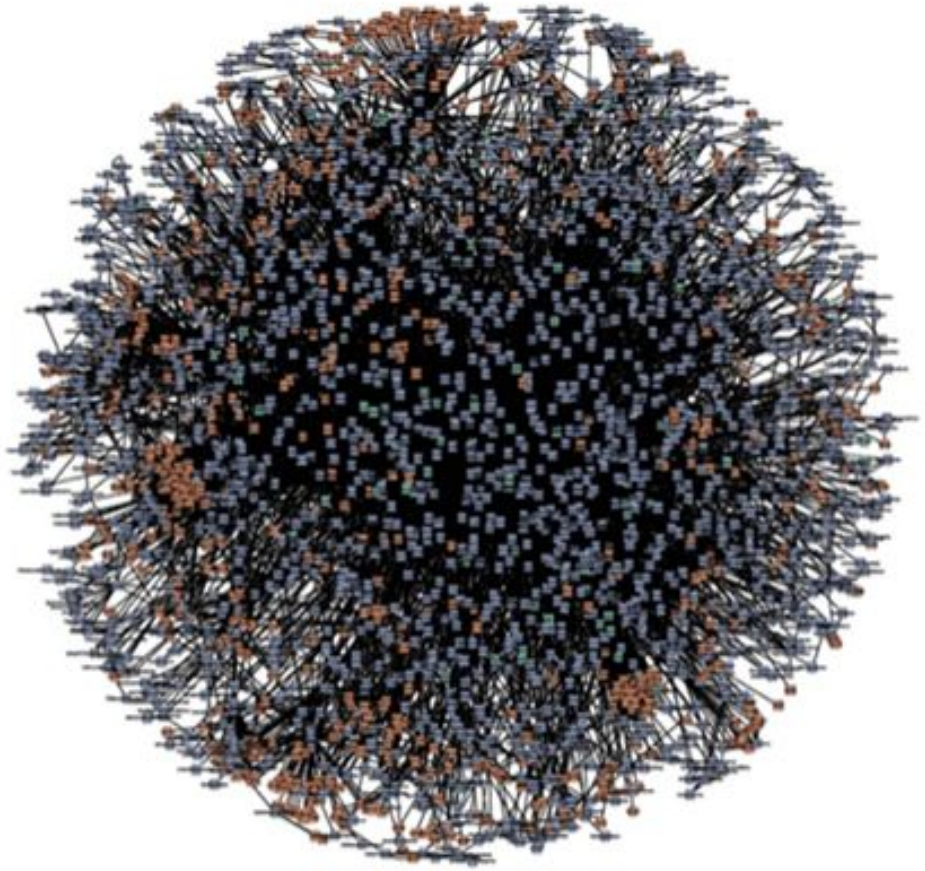
- Small units of work
- Isolation of components, code
- Automated deployments (CI/CD)

midwest
architecture community
collaboration

NETFLIX “DEATH STAR” - +700 MICROSERVICES

midwest
architecture community
collaboration

AMAZON.COM - ???? MICROSERVICES



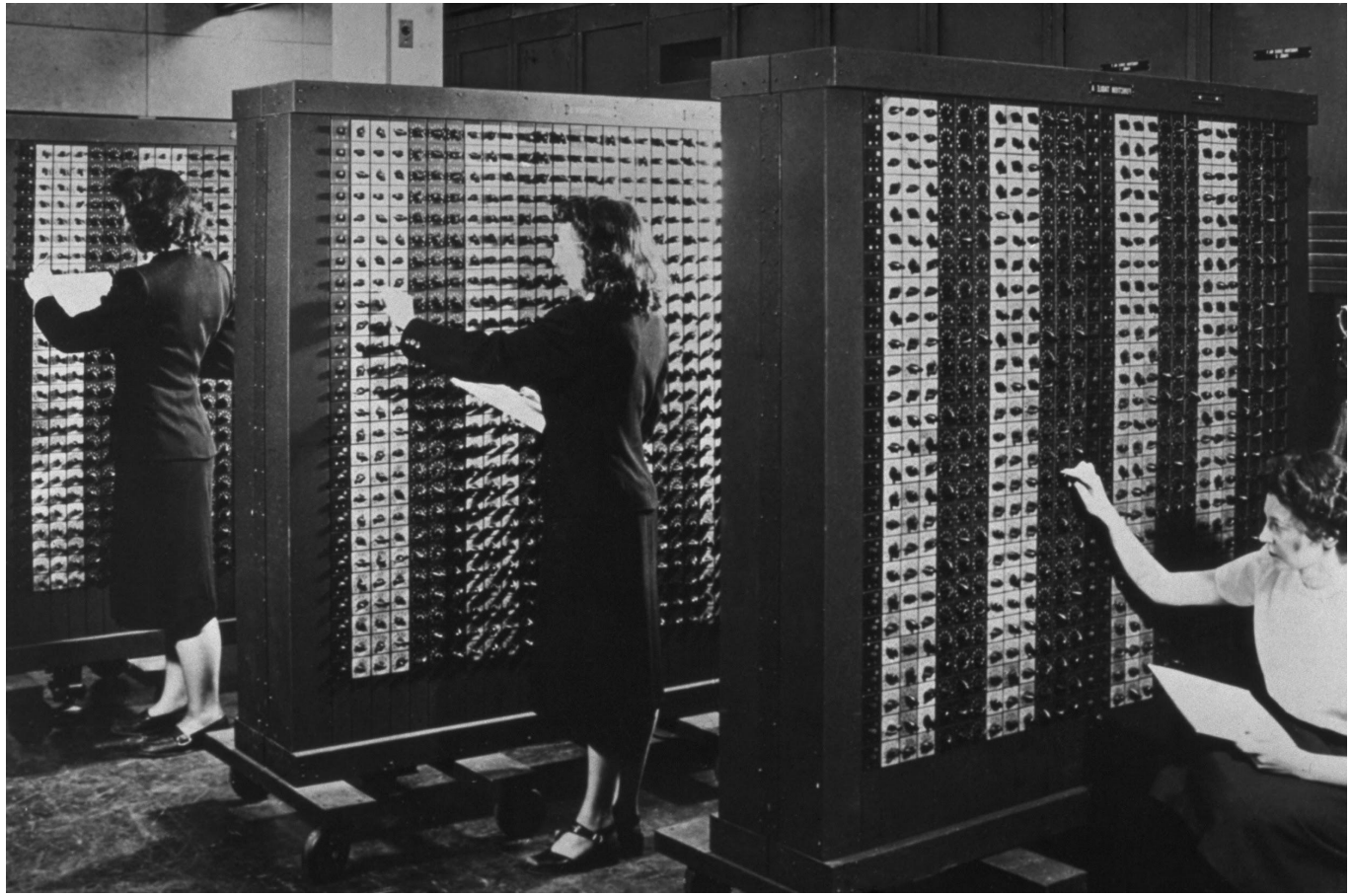
midwest
architecture community
collaboration

HISTORY

"You need to know the past…

…to understand the future"

~ Dr. Carl Segan



midwest
architecture community
collaboration

CLIENT / SERVER APPROACH

Connecting clients to backend corporate servers

- Solved out data problem – data is in one place
- Did not solve our distribution/deployment problem

THE “BOOM” OF THE .COM

- Webpages or Web-Resources

- You have everything under the sun – the wild west
- ColdFusion development
- PHP development
- Backend processing

- …. And eventually the “Post Back” development

midwest
architecture community
collaboration

API DEVELOPMENT

- Webpages or Web-Resources

- The idea of a stand alone webpage the served up data
- Browser applications could ingest this data

midwest
architecture community
collaboration

EVERYONE LOVES "JAVASCRIPT"

- The invention of SPA (Single Page Applications)

- We needed a better User Experience
- DevExpress / Infragistics / Telerik Controls
- Ruby on Rails
- Microsoft's "Silverlight"… yes, you can breath now

- …. Everyone was struggling to have a better UI

THE MARRIAGE WITH SPA

- The invention of Microservices

- Miniature little APIs
- Monolithic Systems are now in place… What do we do?

midwest
architecture community
collaboration

THE MARRIAGE WITH SPA

- The invention of Microservices

- Miniature little APIs
- We can isolate business logic and ultimately business domains
- Monolithic Systems are now in place… What do we do?

MICROSERVICES

Structure / Layout / Design

midwest
architecture community
collaboration

GOALS – ENTIRE SYSTEM

- Overall goals for the entire set of Microservices

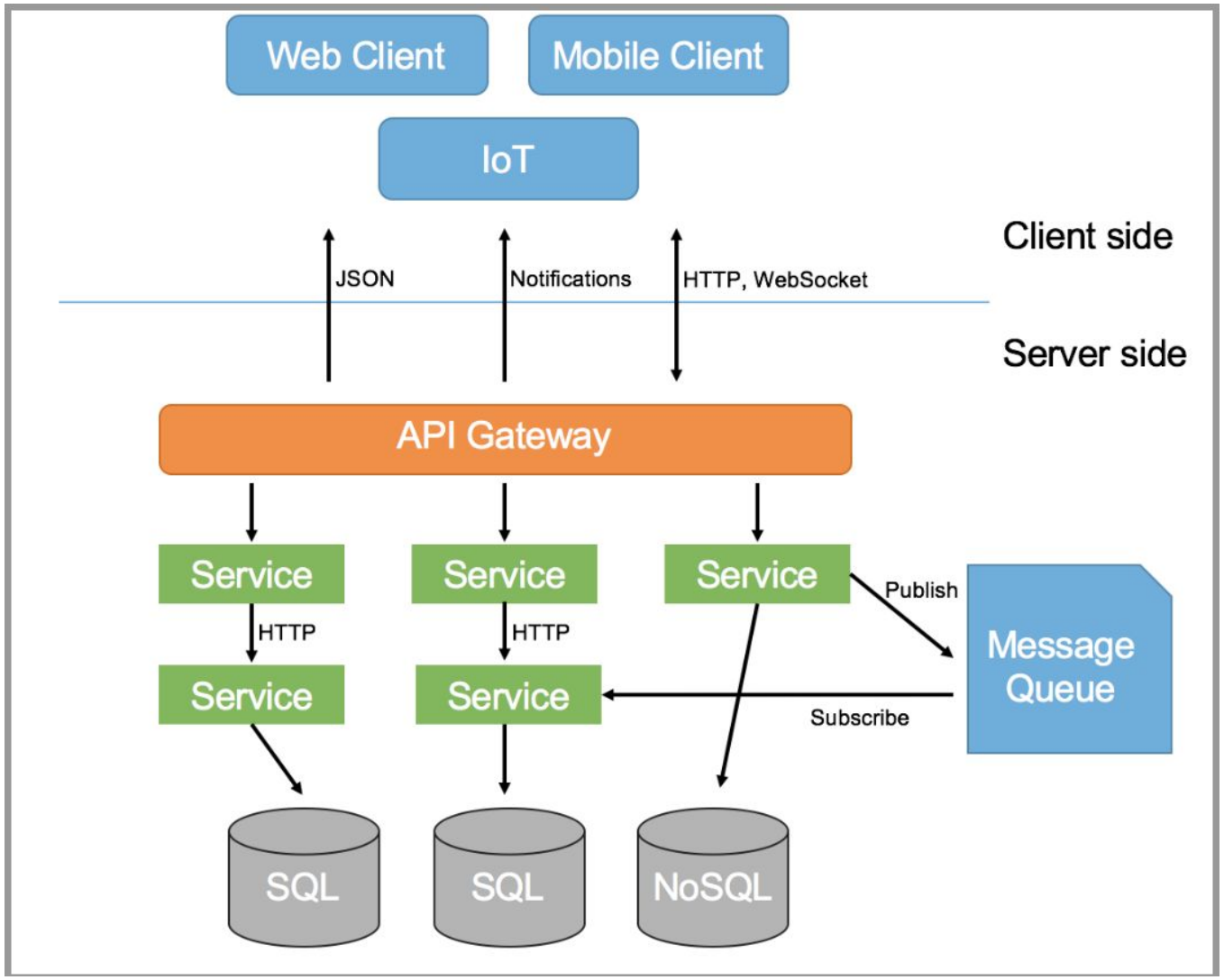
  - Databases are isolated/separated
  - Microservices are isolated/separated
  - Different versions can interact with each other (with consumers)
  - Quick/Automated deployments
  - Consumers can plug and play

OVERALL STRUCTURE

- Components – These are your basics

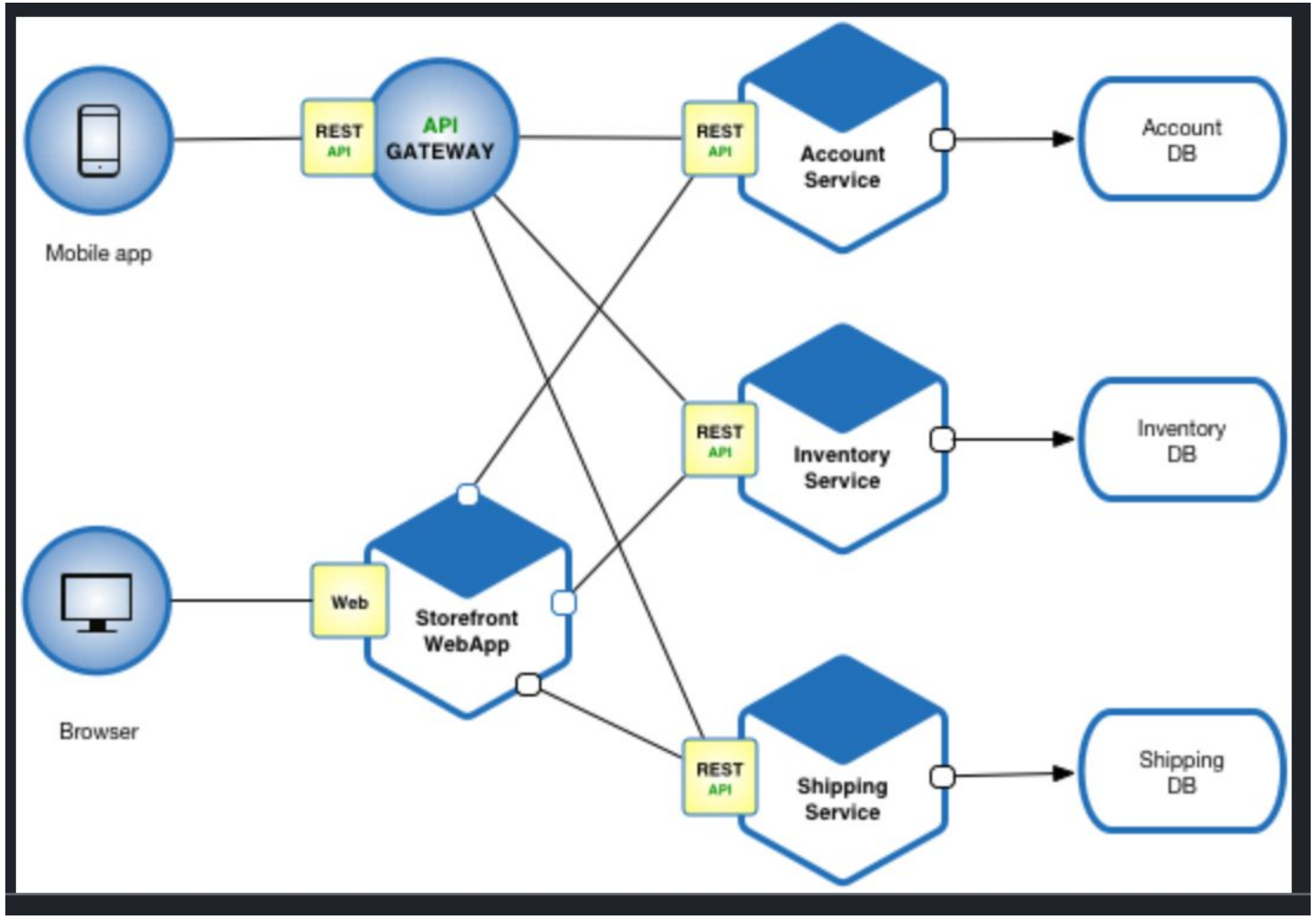
- Microservices
  - “Atomic” and “Composite”
- Single/Isolated data storage system (each Microservice)
- API Gateway (front side caching)
- Data Warehouse (backend data movement)
- Messaging System

OVERALL STRUCTURE



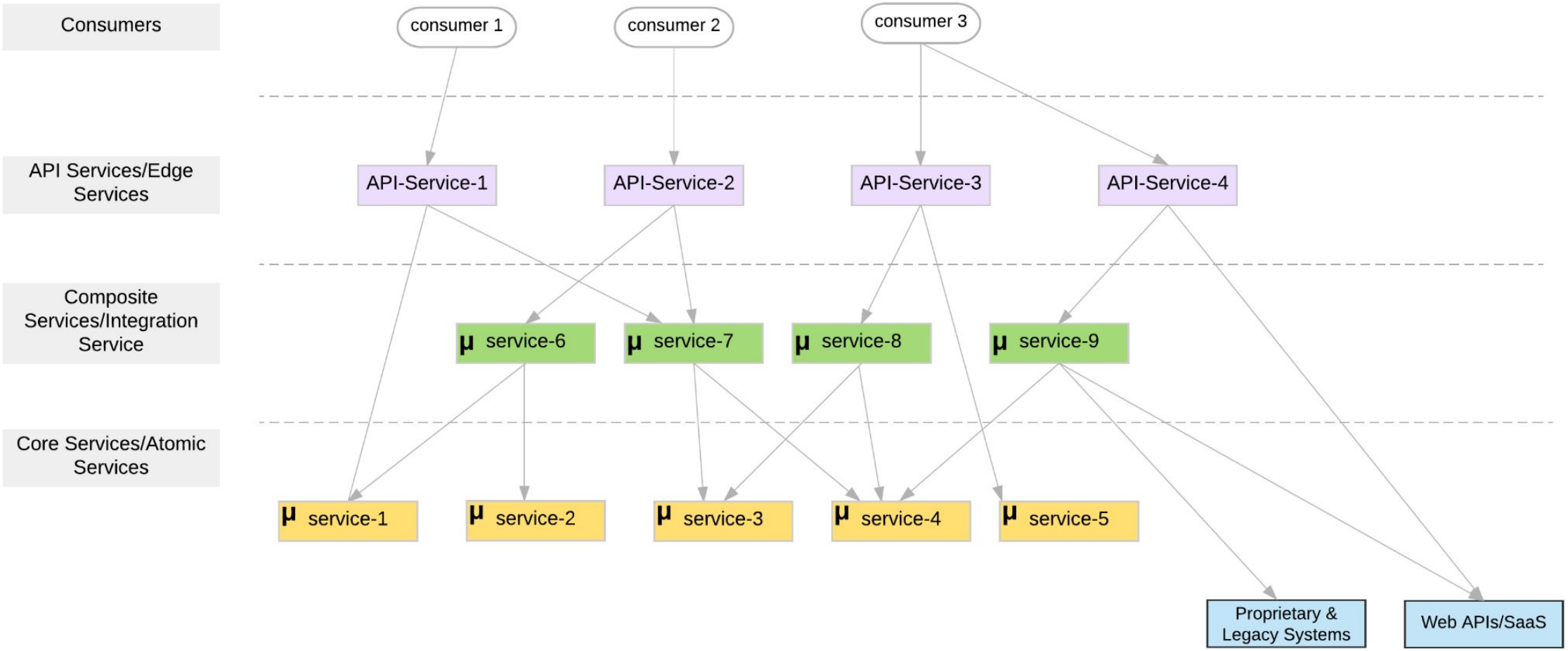
midwest architecture community collaboration

OVERALL STRUCTURE



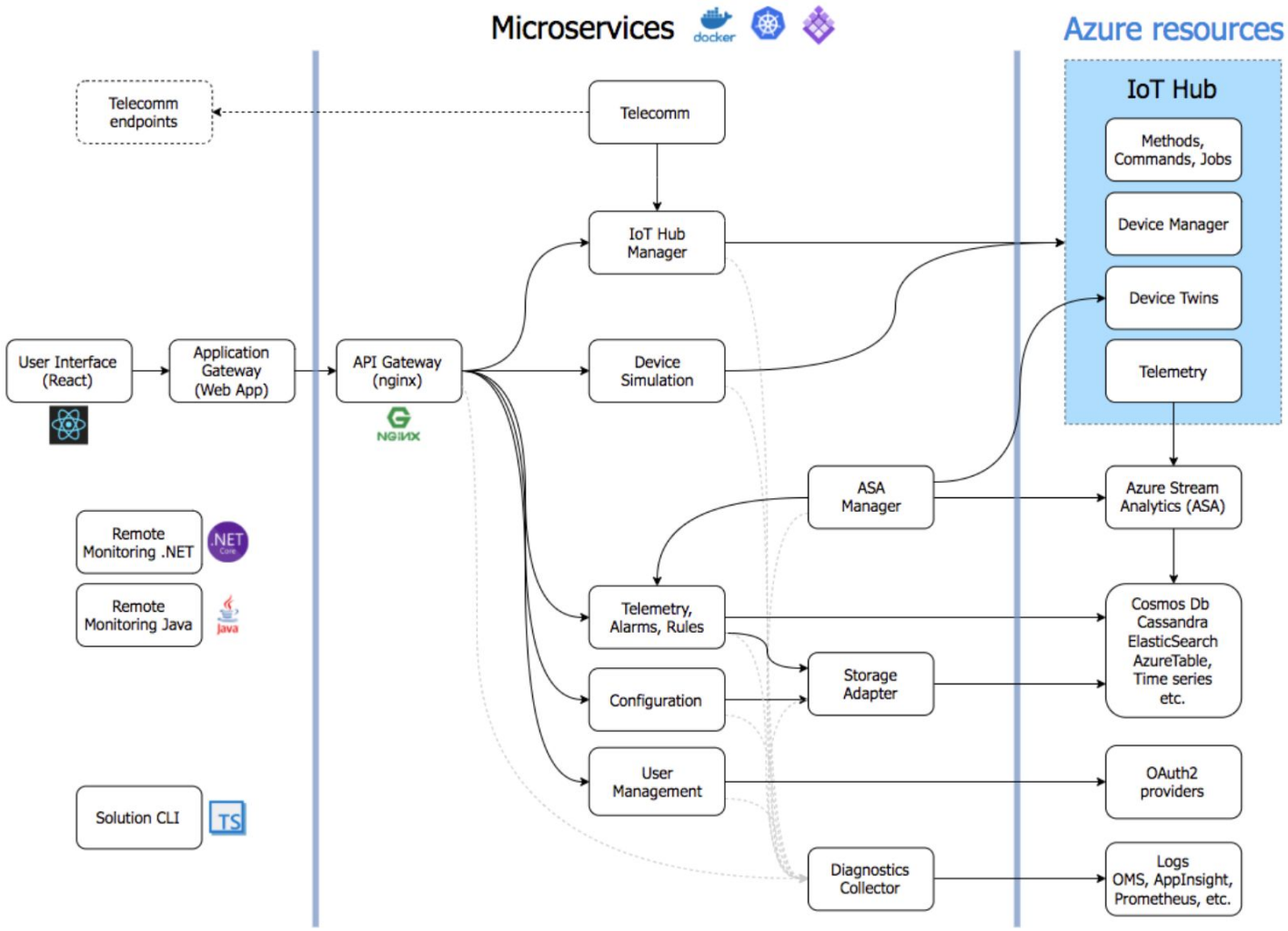
midwest architecture community collaboration

OVERALL STRUCTURE



midwest architecture community collaboration

OVERALL STRUCTURE



midwest architecture community collaboration

GOALS – SINGLE SERVICE

- Overall goals with a single Microservice

  - Built and checked in – 8 hours flat
  - Built / Deployed / Tested in 16 hours flat (to DEV)
  - Database/Data-storage - created automatically
  - Documentation is automatically created (i.e. Swagger)

INTERNAL STRUCTURE

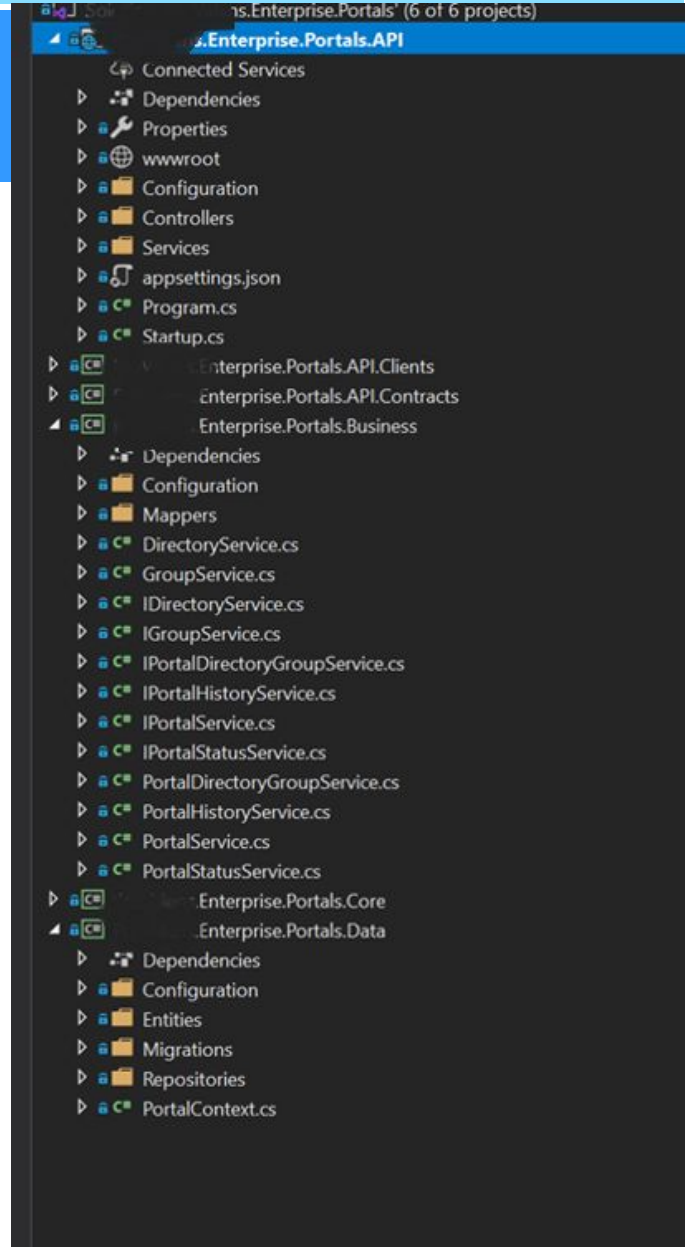
- Microservice Layout for success

  - 3-Tier Architecture
  - Fully RESTful (do not deviate)
  - URL Versioning (backwards compatible)
  - Data storage ORM tool (very helpful)
  - Unit tests (automated)

INTERNAL STRUCTURE

- 3-Tier Architecture

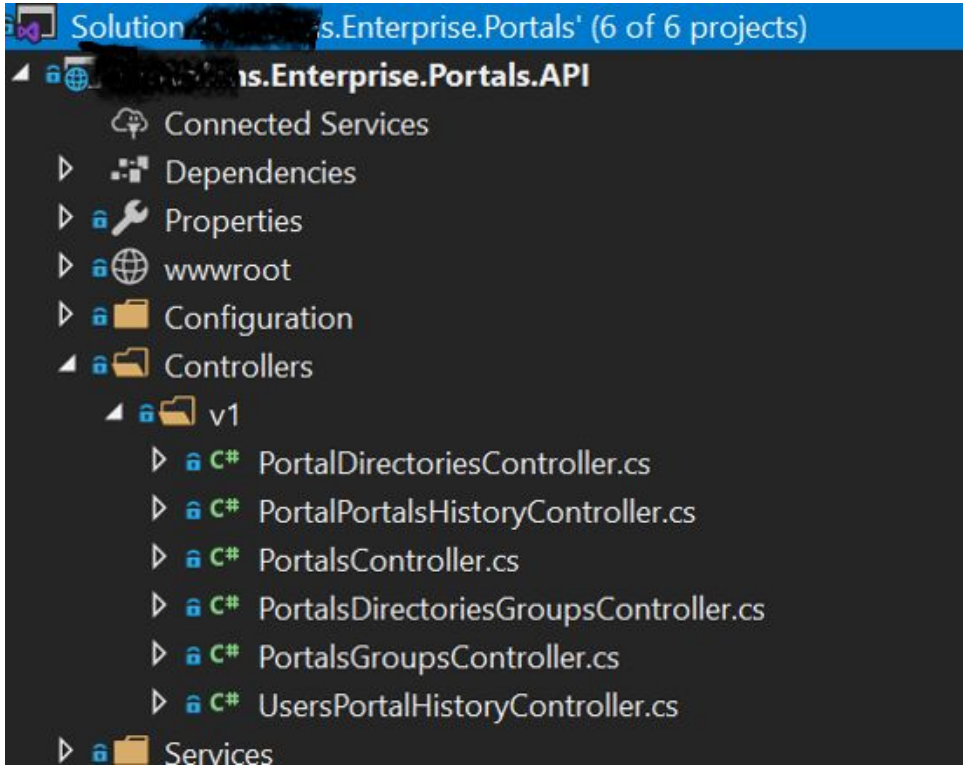
  - Presentation Layer (api endpoints)
  - Business Layer (all logic here)
  - Data Layer (almost fully generated)

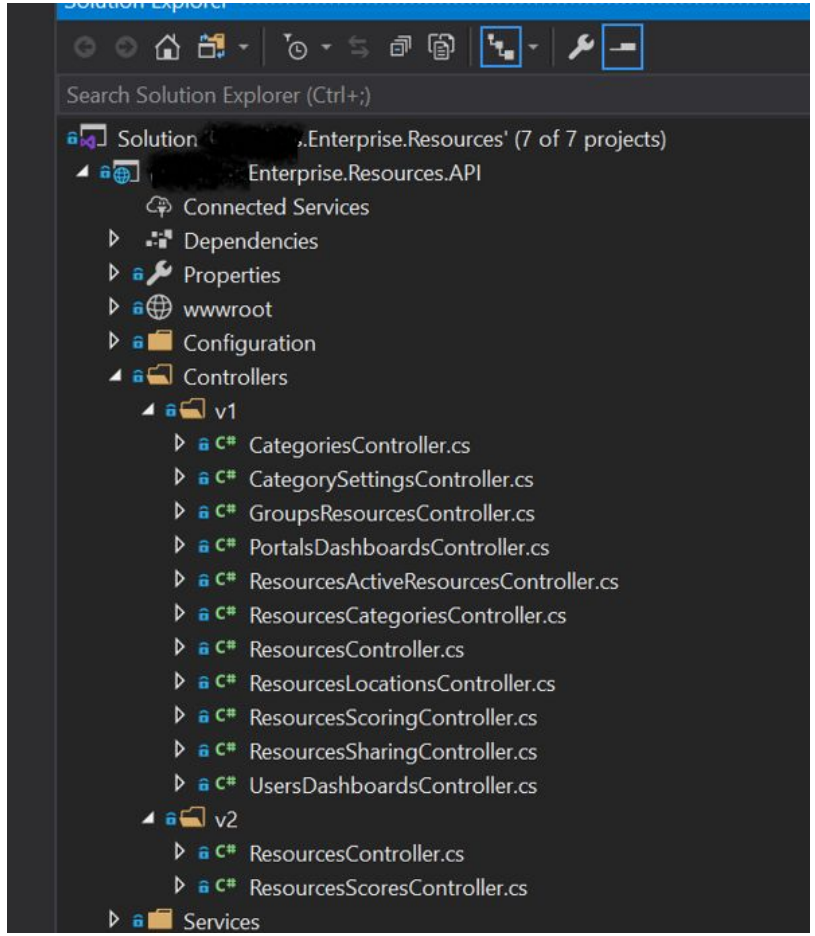


midwest
architecture community
collaboration

INTERNAL STRUCTURE

- Presentation Layer – URL Versioning





midwest architecture community collaboration

INTERNAL STRUCTURE

- Presentation Layer – URL Versioning

```
/// <seealso cref="Microsoft.AspNetCore.Mvc.Controller" />
[Produces( contentType: "application/json")]
[ApiVersion("2.0", Deprecated = false)]
[Route( template: "v{version:apiVersion}")] //Route Prefix
public class ResourcesController : Controller
{
    /// <summary>
    /// RESTful endpoint for retrieving a list of resources
    /// </summary>
    /// <returns></returns>
    [HttpGet, Route( template: "resources"), MapToApiVersion("2.0")]
    public IActionResult Get()
    {
        //just mocked out for now
        var list = new List<Resource>();
        list.Add( item: new Resource());
        return Ok(list);
    }

    /// <summary>
    /// Gets the specified resource identifier.
    /// </summary>
    /// <param name="resourceId">The resource identifier.</param>
    /// <returns></returns>
    [HttpGet, Route( template: "resources/{resourceId}"), MapToApiVersion("2.0")]
    public IActionResult Get(Guid resourceId)
    {
        //just mocked out for now
        return Ok(new Resource());
    }
```

INTERNAL STRUCTURE

- Business Layer
  - All logic resides here

```
public List<Resource> Get(Guid? dashboardId, Guid? portalId, Guid? classificationId, string keywords = null, bool activeOnly = true, string tags = null, float? latitude = 
{

    _logger.LogTrace(message:"Executing GET");
    var resources:IQueryable<Resource> = _repository.Get()
        .Where(x:Resource => x.IsActive && !x.IsDeleted);

    //These are all in seperate methods so that the even the null filters don't get applied if not needed
    resources = ApplyClassificationFilter(resources, classificationId);
    resources = ApplyTagFilters(resources, tags);
    resources = ApplyTitleFilter(resources, title);
    resources = ApplyKeywordFilters(resources, keywords);
    resources = ApplyCategoryFilters(resources, categoryIds);
    resources = ApplyLocationFilter(resources, latitude, longitude, distance, location);

    if (portalId != null && !admin)
    {
        resources = ApplyPortalIdFilter(resources, portalId);
    }

    // Order By
    if (groupId != null)
    {
        resources = ApplyGroupIdFilter(resources, groupId);
        resources = resources.OrderBy(x:Resource => x.GroupResources.First(y:GroupResource => y.IsActive && !y.IsDeleted && y.GroupId == groupId).DisplayOrder);
    }
    else
    {
        resources = ApplyIsActiveInDashboardFilter(resources, dashboardId, activeOnly);
        resources = resources.OrderByDescending(x:Resource => x.ModifiedByDate);
    }

    // Pagination
    if (paging?.RequestCount == true)
    {
        paging.TotalCount = resources.Count();
    }

    int page = paging?.Page ?? 0;
    int pageSize = paging?.PageSize ?? 1000;
    resources = resources.Skip(pageSize * page).Take(pageSize);


    var models:List<Resource> = resources.Select(x:Resource => x.ToResourceModel(dashboardId, _blobService.Configuration)).ToList();
    _logger.LogInformation(message:"Successfully executed GET");
    return models;
}
```

midwest architecture community collaboration

INTERNAL STRUCTURE

- Business Layer – DO NOT EXPOSE DATA ENTITIES!

```
public static Data.Entities.Dashboard ToEntity(this Dashboard model)
{
    var entity = new Data.Entities.Dashboard();
    entity.DashboardId = model.DashboardId ?? Guid.Empty;
    return model.UpdateEntity(entity);
}

public static Data.Entities.Dashboard UpdateEntity(this Dashboard model, Data.Entities.Dashboard entity)
{
    entity.Name = model.Name;
    entity.PortalId = model.PortalId;
    entity.UserId = model.UserId;
    entity.IsDefault = model.IsDefault;
    return entity;
}
```

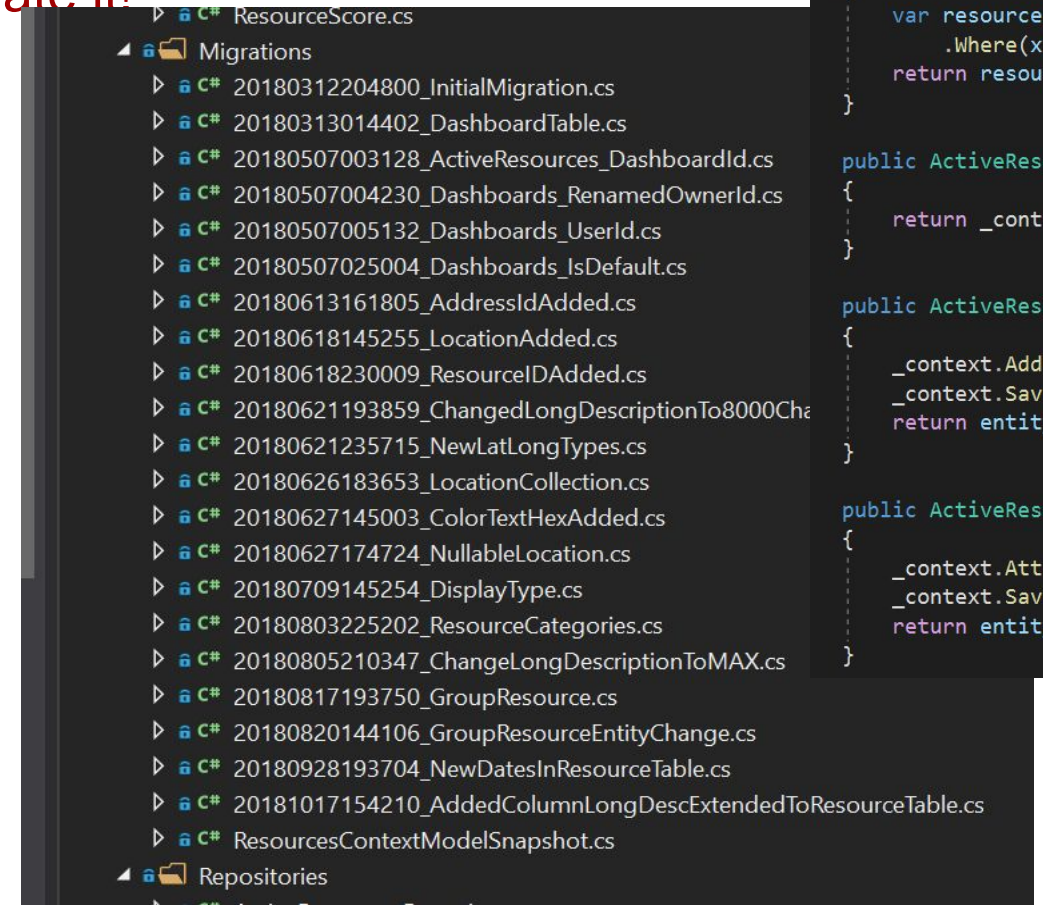
```
public static Dashboard ToModel(this Data.Entities.Dashboard entity)
{
    var model = new Dashboard();
    return entity.UpdateModel(model);
}

public static Dashboard UpdateModel(this Data.Entities.Dashboard entity, Dashboard model)
{
    model.DashboardId = entity.DashboardId;
    model.Name = entity.Name;
    model.PortalId = entity.PortalId;
    model.UserId = entity.UserId;
    model.IsDefault = entity.IsDefault;
    return model;
}
```


midwest architecture community collaboration

INTERNAL STRUCTURE

- Data Layer – Automate it!




```
public IQueryable<ActiveResource> Get()
{
    var resources :IQueryable<ActiveResource> =  _context.ActiveResources
        .Where(x :ActiveResource=>x.IsActive);
    return resources;
}

public ActiveResource Get(Guid resourceId, Guid dashboardId)
{
    return _context.ActiveResources.FirstOrDefault(x :ActiveResource => x
}

public ActiveResource Create(ActiveResource entity)
{
    _context.Add(entity);
    _context.SaveChanges();
    return entity;
}

public ActiveResource Update(ActiveResource entity)
{
    _context.Attach(entity);
    _context.SaveChanges();
    return entity;
}
```

midwest architecture community collaboration

DEVOPS (DEVELOPMENT & OPERATIONS)

Why is this so important?

midwest architecture community collaboration

DEVOPS

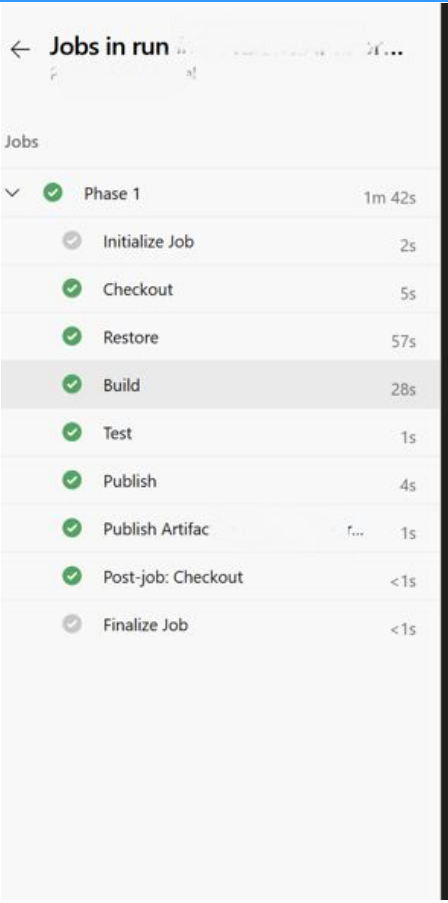
- Development/Deployment Operations

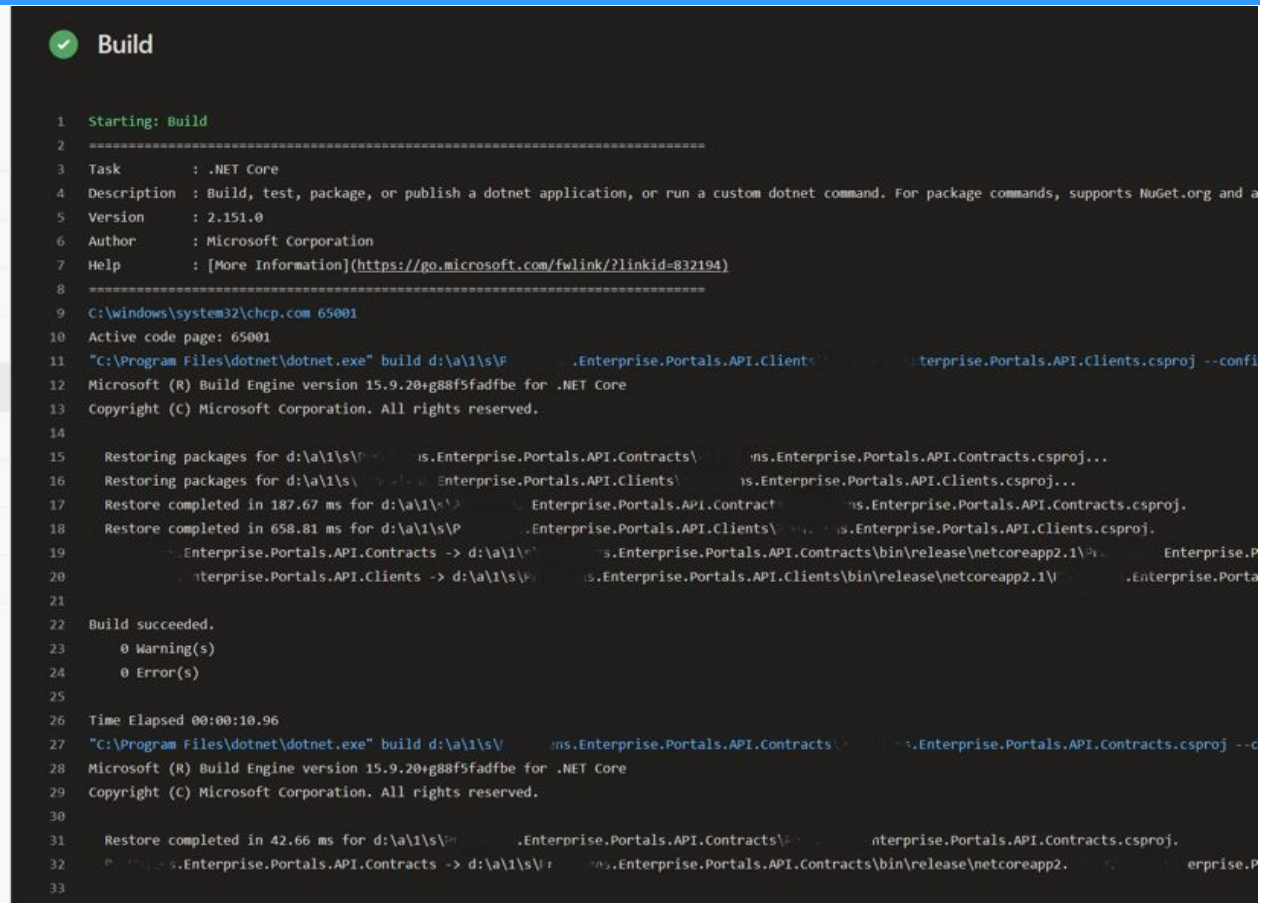
- Need to get your code promoted INSTANTLY
- CI/CD – Continuous Integration / Continuous Deployment
- Automated testing and code coverage

midwest architecture community collaboration

DEVOPS

- Automated builds…





midwest architecture community collaboration

DEVOPS

- “Releasing” code
  - Pushing code out

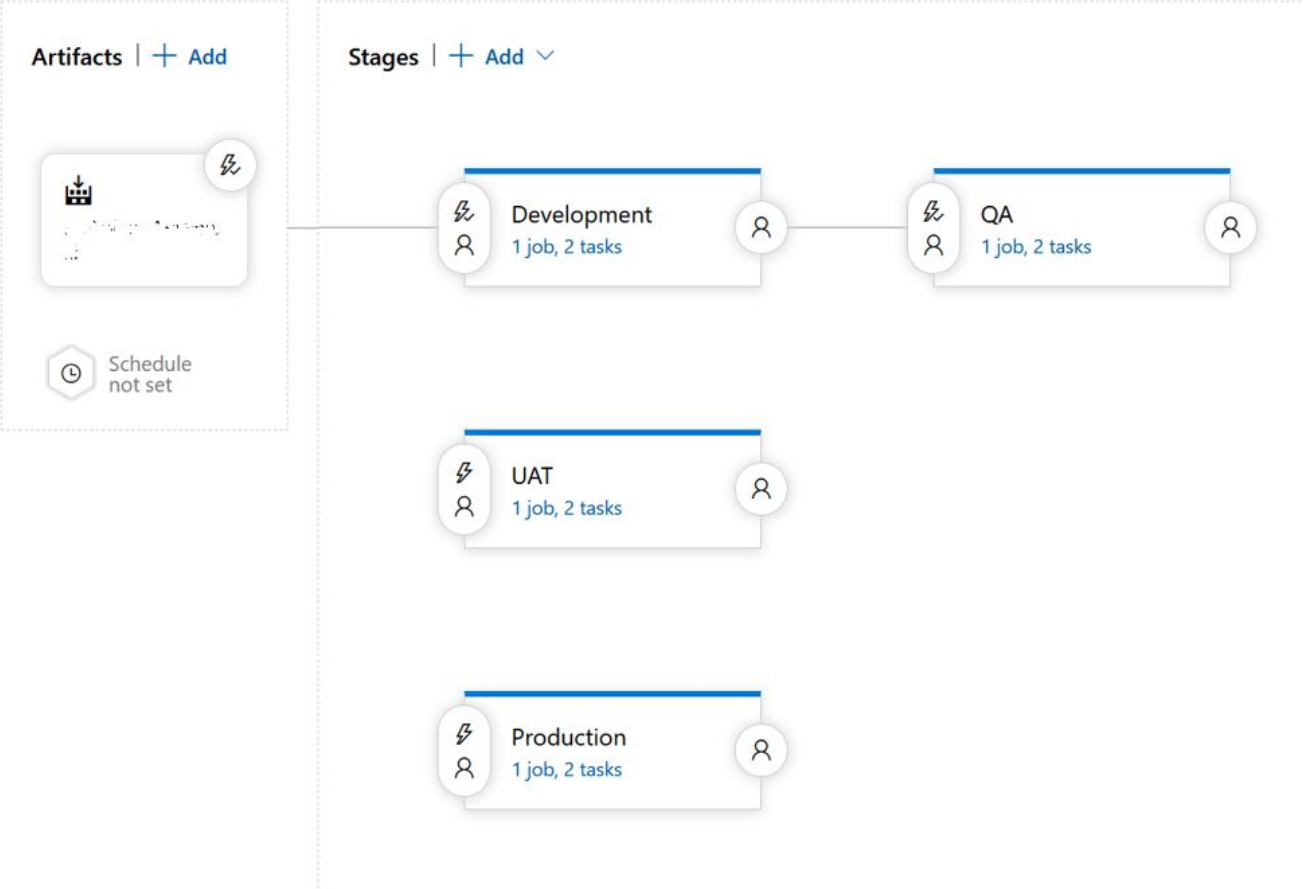
Releases | Deployments | Analytics

Releases	Created	Stages			
M Release-094 for build s.API.Portal-2019.05.19.002 s.API.Portal-2019.05.19.002 master	5/18/2019, 10:00:33 PM	Developm...	QA	UAT	Production
M Release-093 for build s.API.Portal-2019.05.19.001 API.Portal-2019.05.19.001 master	5/18/2019, 8:39:55 PM	Developm...	QA	UAT	Production
M Release-092 for buil s.API.Portal-2019.04.10.002 s.API.Portal-2019.04.10.002 master	4/10/2019, 10:04:51 AM	Developm...	QA	UAT	Production
M Release-088 for build s.API.Portal-2018.11.16.003 s.API.Portal-2018.11.16.003 master	11/16/2018, 11:51:51 AM	Developm...	QA	UAT	Production
MN Release-082 for build s.Enterprise.Portal.API-2018.09.17.001 s.Enterprise.Portal.API-2018.09.17.001 master	9/19/2018, 1:13:00 PM	Developm...	QA	UAT	Production

midwest architecture community collaboration

DEVOPS

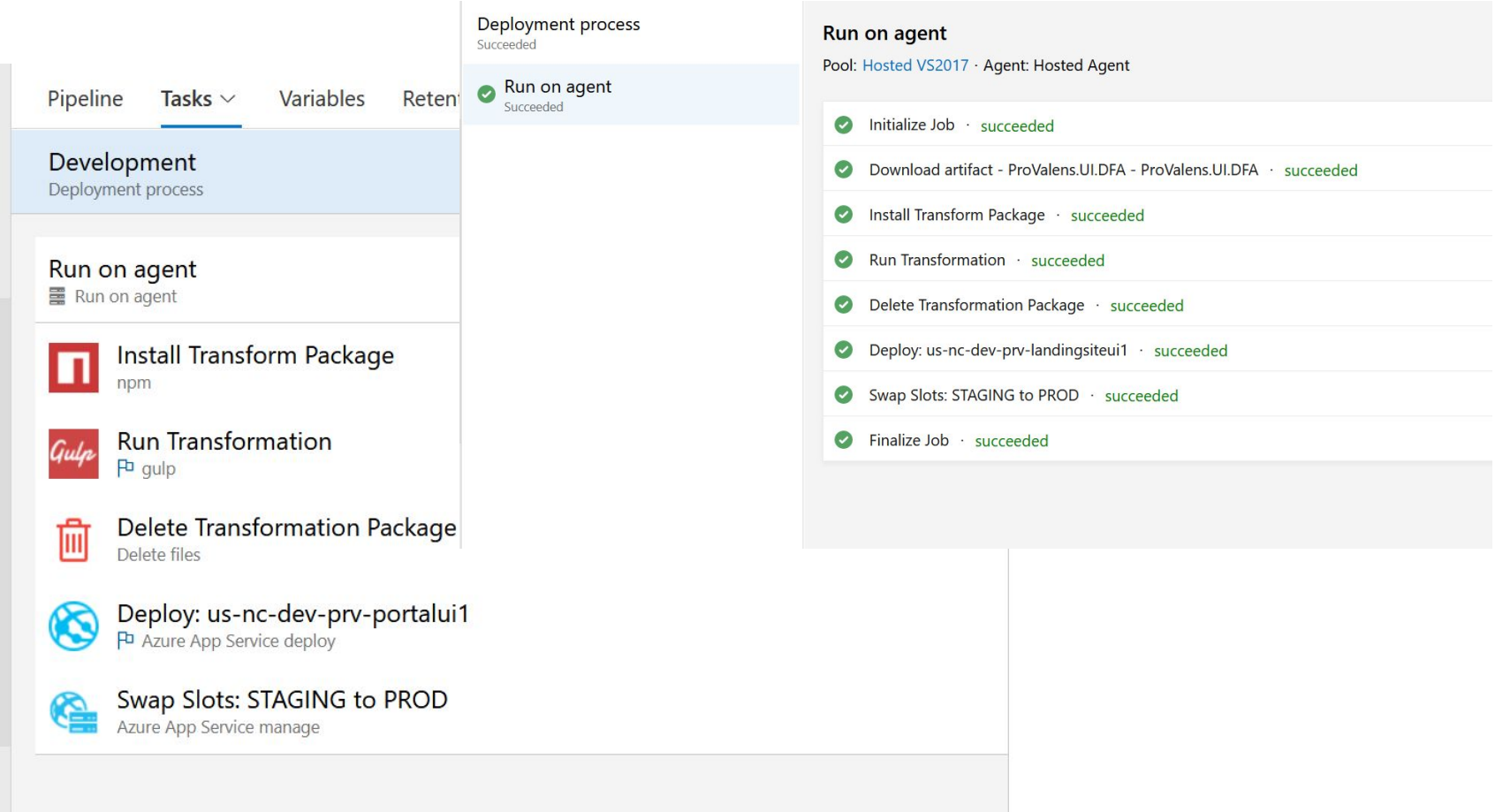
- “Releasing” code
  - Pushing code out



midwest architecture community collaboration

DEVOPS

- “Releasing” code
  - Pushing code out



midwest architecture community collaboration

DEVOPS

- Creating components in the cloud automatically

us-nc-prod-portalui2 Web App Running	us-nc-prod-registrati... Web App Running	us-nc-prod-userui Web App Running	us-nc-prod-adminui2 Web App Running	us-nc-prod-landings... Web App Running
us-nc-prod-portalapi API app Running	us-nc-prod-organiza... API app Running	us-nc-prod-resource... API app Running	us-nc-prod-securitie... API app Running	us-nc-prod-userprofi... API app Running
us-nc-prod-searchapi API app Running	us-nc-prod-imageapi API app Running	us-nc-prod-notificati... API app Running	us-nc-prod-categori... API app Running	us-nc-prod-location... API app Running
		us-nc-prod-provalens SQL server Available		
us-nc-prod-portals SQL database Online	us-nc-prod-resources SQL database Online	us-nc-prod-organiza... SQL database Online	us-nc-prod-security SQL database Online	us-nc-prod-userprofi... SQL database Online
	us-nc-prod-images SQL database Online	us-nc-prod-notificati... SQL database Online	us-nc-prod-categori... SQL database Online	us-nc-prod-locations SQL database Online

midwest architecture community collaboration

BENEFITS - VS - CONSTRAINTS

midwest
architecture community
collaboration

BENEFITS

- Retiring the monolithic systems

  - We can now have small isolated components
  - Physical separation
  - You won't break my stuff – I won't break yours
  - Piece components together that you need

midwest architecture community collaboration

CONSTRAINTS

- Bringing everything together

  - How do you perform complex queries
  - How do you report out large data graphs
  - Separation is problematic – when everything is related

WHAT NOT TO DO

midwest
architecture community
collaboration

WHAT WE DID WRONG

- Our first Microservice architecture

  - Atomics only – No coupling
  - Skip the API Gateway – over kill
  - Force consumers to merry up data

WHAT WE DID WRONG

- Let's try it again – Round two

  - Too many microservices.. Too granular
  - Mass Data import was (almost) impossible
  - No messages or data movement

  - …. Failed to explain the benefits of the architecture

midwest
architecture community
collaboration

NOW WHERE TO?

- The world is changing again!

- Serverless Technology
  - AWS Lambda Functions
  - Azure Functions
  - Serverless Templates (SAM)

- Data Analytics
  - IoT – data collection component
  - Data Lake – not your average data warehouse

OTHER "OUTLIERS"

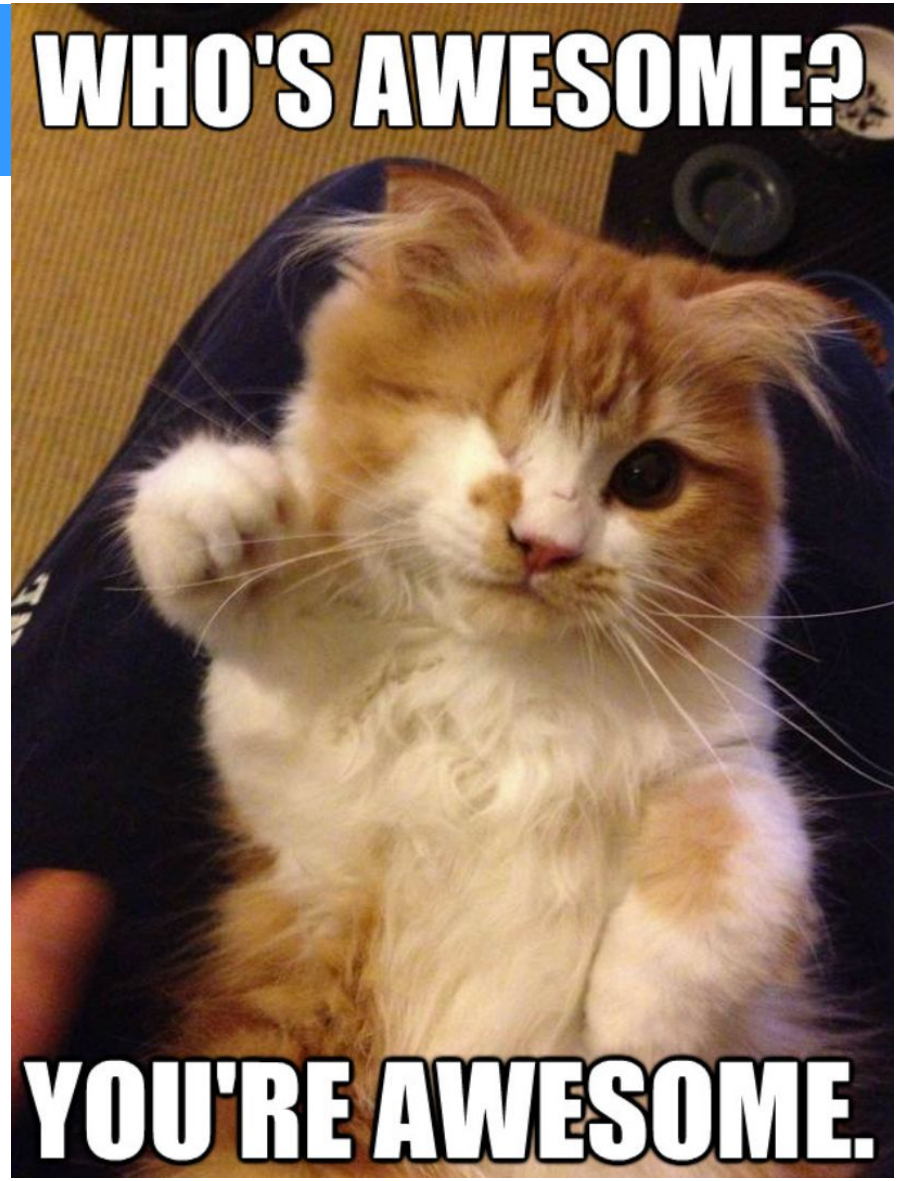
- Spin-Offs of Microservices

- GraphQL
  - The gang at Facebook

- Graph Databases and presentation
  - How do we use this in a Microservice?

THANK YOU

- Please reach out – I'm happy to help



midwest
architecture community
collaboration