Versioning and Evolving Microservices in ASP.NET Core

UNDERSTANDING WHY APIS EVOLVE

Mark Heath
CLOUD ARCHITECT

@mark_heath   www.markheath.net

Overview

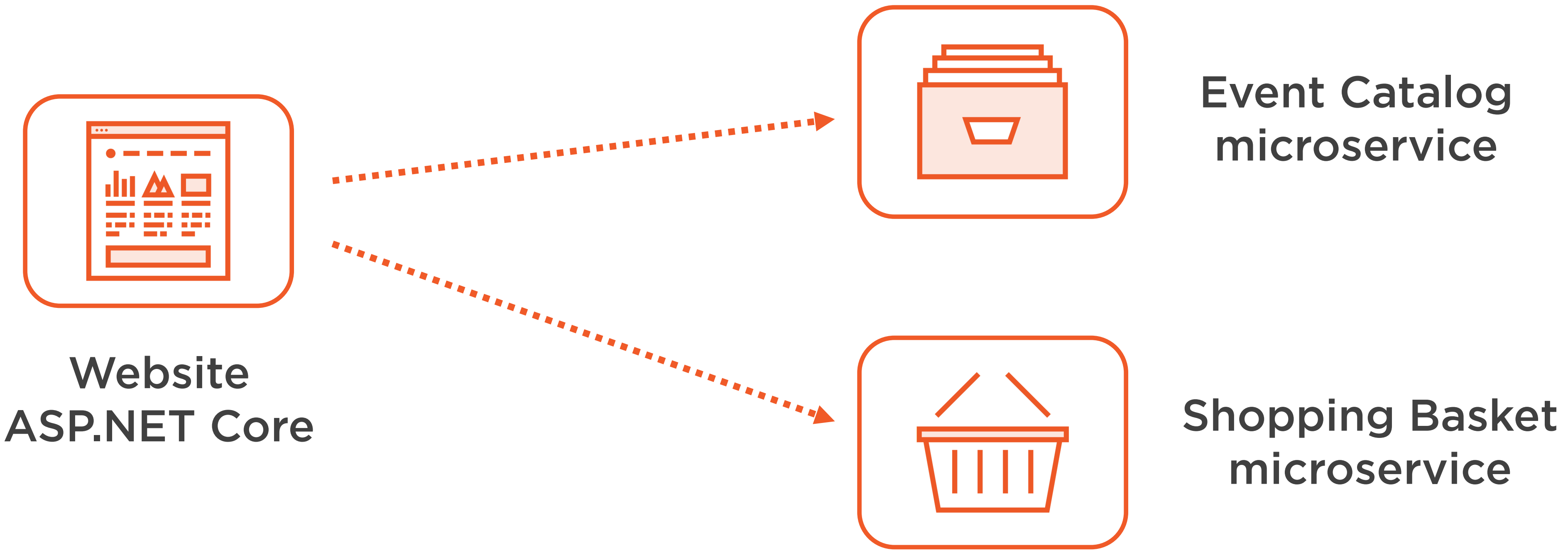
GloboTicket demo application

Why we need to evolve microservice APIs

Introducing GloboTicket

GLOBOTICKET

A Globomantics Company



Building Microservices with ASP.NET Core

This course is part of a learning path on Pluralsight

Microservices: The Big Picture	Getting Started	Microservices Communication
Data Management	Securing Microservices	Versioning
Deploying Microservices	Cross-cutting concerns	Scalability and availability

Demo

Building GloboTicket

- Starter project available in course download materials

Demo

Running Entity Framework Core database migrations

- Microservices should own their own data
- Event Catalog and Shopping Basket microservices both have their own SQL database

Demo

Running GloboTicket from the command line

Demo

Running GloboTicket from Visual Studio

Microservice APIs need to evolve

You’re unlikely to create the perfect API first time

Meeting New Business Requirements

Existing behavior:	Retrieve all events filtered by category
New requirement:	Search for events by keyword
New requirement:	Search for events by location

Better Understanding of Business Domain

Original design:

An event has a single ticket price

```
public class Event
{
   public Guid EventId { get; set; }
   public string Name { get; set; }
   public int Price { get; set; }
   public string Artist { get; set; }
   public DateTime Date { get; set; }
   public string Description { get; set; }
   public string ImageUrl { get; set; }
   public Guid CategoryId { get; set; }
   public string CategoryName { get; set; }
}
```

New requirement:

An event has multiple ticket prices

```
public class Event
{
   // ...
   public Ticket[] Tickets { get; set; }
   // ...
}

public class Ticket
{
   public int Price { get; set; }
   public string Description { get; set; }
}
```

Performance Considerations

Replace chatty communication with efficient APIs

Microservice APIs can evolve for several reasons:

- New Requirements
- Better Domain Understanding
- Improved Performance

We need a good versioning strategy

Summary

GloboTicket sample application

- Build and run with .NET Core

Microservice APIs need to evolve

- Potential to break existing clients

Up next…

Maintaining Backwards Compatibility