Generowanie dokumentów (Document Service) z RabbitMQ

Docker

1. **Pobranie obrazu**

```
docker pull rabbitmq:management
```

2. **Uruchomienie kontenera**

```
docker run --name rabbitmq-shopper -d -p 5672:5672 -p 15672:15672
rabbitmq:management
```

> Port `5672` — komunikacja aplikacji z RabbitMQ
> Port `15672` — panel zarządzania (UI)

3. **Dostęp do panelu zarządzania**
   Przejdź do http://localhost:15672
   Domyślne dane logowania:

- **login:** `guest`
- **hasło:** `guest`

4. **Sprawdzenie statusu (opcjonalnie)**

```
docker exec -it rabbitmq-shopper rabbitmqctl status
```

Docker Compose

1. **Utwórz plik *docker-compose.yaml***

```
services:
  rabbitmq:
    container_name: rabbitmq-shopper
    image: rabbitmq:management
```

```
    ports:
      - "5672:5672"     # komunikacja z aplikacjami
      - "15672:15672"   # panel webowy
```

2. **Uruchom**
   ```bash
   docker-compose up -d

```
3. **Zatrzymaj**
```bash
docker-compose down
```

Shared.Contracts

```
// Shared.Contracts.csproj
<Project Sdk="Microsoft.NET.Sdk">

  <PropertyGroup>
    <TargetFramework>net8.0</TargetFramework>
  </PropertyGroup>

</Project>

// OrderPlaced.cs
namespace Shared.Contracts;

public interface OrderPlaced
{
    string OrderId { get; }
    string UserId { get; }
    decimal TotalAmount { get; }
    DateTime PlacedAt { get; }
}
```

OrderingService

```
// OrderingService.csproj
<Project Sdk="Microsoft.NET.Sdk">
```

```
  <PropertyGroup>
    <OutputType>Exe</OutputType>
    <TargetFramework>net8.0</TargetFramework>
  </PropertyGroup>

  <ItemGroup>
    <PackageReference Include="MassTransit" Version="8.0.0" />
    <PackageReference Include="MassTransit.RabbitMQ" Version="8.0.0" />
    <PackageReference Include="Microsoft.AspNetCore" Version="2.2.0" />
    <PackageReference Include="Microsoft.AspNetCore.Mvc.Core" Version="2.2.5"
/>
    <PackageReference Include="Microsoft.AspNetCore.Hosting" Version="2.2.7"
/>
  </ItemGroup>

  <ItemGroup>
    <ProjectReference Include="..\Shared.Contracts\Shared.Contracts.csproj" />
  </ItemGroup>

</Project>
```

```
// Program.cs
using MassTransit;
using Shared.Contracts;

var builder = WebApplication.CreateBuilder(args);

builder.Services.AddMassTransit(x =>
{
    x.UsingRabbitMq((context, cfg) =>
    {
        cfg.Host("rabbitmq", "/", h =>
        {
            h.Username("guest");
            h.Password("guest");
        });
    });
});

builder.Services.AddEndpointsApiExplorer();

var app = builder.Build();

app.MapPost("/orders", async (IBus bus, OrderRequestDto dto) =>
```

```
{
    var orderId = Guid.NewGuid().ToString();

    await bus.Publish<OrderPlaced>(new
    {
        OrderId = orderId,
        UserId = dto.UserId,
        TotalAmount = dto.TotalAmount,
        PlacedAt = DateTime.UtcNow
    });

    return Results.Ok(new { OrderId = orderId });
});

app.Run();

// OrderRequestDto.cs
namespace OrderingService;

public class OrderRequestDto
{
    public string UserId { get; set; } = default!;
    public decimal TotalAmount { get; set; }
}
```

Document Service

```
// DocumentService.csproj
<Project Sdk="Microsoft.NET.Sdk">
  <PropertyGroup>
    <TargetFramework>net8.0</TargetFramework>
  </PropertyGroup>
  <ItemGroup>
    <PackageReference Include="MassTransit" Version="8.0.0" />
    <PackageReference Include="MassTransit.RabbitMQ" Version="8.0.0" />
    <PackageReference Include="QuestPDF" Version="2024.2.0" />
  </ItemGroup>
</Project>

// Contracts/OrderPlaced.cs
namespace DocumentService.Contracts;

public interface OrderPlaced
{
```

```
    string OrderId { get; }
    string UserId { get; }
    decimal TotalAmount { get; }
    DateTime PlacedAt { get; }
}

// Pdf/PdfGenerator.cs
using QuestPDF.Fluent;
using QuestPDF.Helpers;
using QuestPDF.Infrastructure;

namespace DocumentService.Pdf;

public static class PdfGenerator
{
    public static Task GenerateInvoiceAsync(string orderId)
    {
        var fileName = $"invoice_{orderId}.pdf";

        Document.Create(container =>
        {
            container.Page(page =>
            {
                page.Margin(50);
                page.Content().Column(col =>
                {
                    col.Item().Text($"Invoice for Order: {orderId}")
                              .FontSize(24)
                              .Bold()
                              .FontColor(Colors.Blue.Medium);
                });
            });
        }).GeneratePdf(fileName);

        Console.WriteLine($"PDF generated: {fileName}");
        return Task.CompletedTask;
    }
}

// Consumers/OrderPlacedConsumer.cs
using DocumentService.Contracts;
using DocumentService.Pdf;
using MassTransit;

namespace DocumentService.Consumers;
```

```
public class OrderPlacedConsumer : IConsumer<OrderPlaced>
{
    public async Task Consume(ConsumeContext<OrderPlaced> context)
    {
        var message = context.Message;
        Console.WriteLine($"Received OrderPlaced for OrderId:
{message.OrderId}");

        await PdfGenerator.GenerateInvoiceAsync(message.OrderId);
    }
}

// Program.cs
using DocumentService.Consumers;
using MassTransit;

var builder = Host.CreateApplicationBuilder(args);

builder.Services.AddMassTransit(x =>
{
    x.AddConsumer<OrderPlacedConsumer>();

    x.UsingRabbitMq((ctx, cfg) =>
    {
        cfg.Host("rabbitmq", "/", h =>
        {
            h.Username("guest");
            h.Password("guest");
        });

        cfg.ReceiveEndpoint("document-service-orderplaced", e =>
        {
            e.ConfigureConsumer<OrderPlacedConsumer>(ctx);
        });
    });
});

var app = builder.Build();
await app.RunAsync();
```

Dockerfile

```
FROM mcr.microsoft.com/dotnet/aspnet:8.0 AS base
WORKDIR /app
```

```
FROM mcr.microsoft.com/dotnet/sdk:8.0 AS build
WORKDIR /src
COPY . .
RUN dotnet restore
RUN dotnet publish -c Release -o /app/publish

FROM base AS final
WORKDIR /app
COPY --from=build /app/publish .
ENTRYPOINT ["dotnet", "DocumentService.dll"]
```

```
// docker-compose.yml
version: "3.9"

services:
  rabbitmq:
    image: rabbitmq:3-management
    ports:
      - "5672:5672"
      - "15672:15672"
    environment:
      RABBITMQ_DEFAULT_USER: guest
      RABBITMQ_DEFAULT_PASS: guest

  document-service:
    build:
      context: ./DocumentService
    depends_on:
      - rabbitmq
    environment:
      - DOTNET_ENVIRONMENT=Production
    volumes:
      - ./generated-pdfs:/app
```