Production checker

Attribute	Details
Dapr runtime version	v1.10.4
Dapr.NET SDK version	v1.10.0
Dapr CLI version	v1.10.0
Language	C#
Platform	.NET 7 (SDK 7.0.202)
Environment	Self hosted

This repository contains a sample application that simulates a production checker system using Dapr. For this sample I've used a barcode scanner setup. There are two checkpoints where a product's barcode can be scanned and using data from the scanners the system calculates the delay. If the delay is greater than 8 seconds, the application sends a mail to the production manager.

Simulation

- The **Simulation** project is an ASP.NET Web console application that will simulate the production lines and barcode scans.
- The **Signal Process Service** is an ASP.NET Core WebAPI application that offers 2 endpoints: `/entrycheckpoint` and `/exitcheckpoint`.
- The **Notification Service** is an ASP.NET Core WebAPI application that offers 1 endpoint: `/collectdelay` for collecting delay issues.

Run the application in Dapr self-hosted mode

Service	Application Port	Dapr sidecar HTTP port	Dapr sidecar gRPC port
SiganlProcessService	6000	3600	60000
NotificationService	6001	3601	60001

Execute the following steps to run the sample application in self hosted mode:

Start infrastructure components:

1. Make sure you have installed Dapr on your machine in self-hosted mode as described in the Dapr documentation.
2. Open a new command-shell.
3. Change the current folder to the `src/Infrastructure` folder of this repo.
4. Start the infrastructure services by executing `start-all.ps1` or `start-all.sh` script. This script will start Mosquitto (MQTT broker), RabbitMQ (pub/sub broker) and Maildev.

Start the services:

1. Open a new command-shell.
2. Change the current folder to the `src/NotificationService` folder of this repo.
3. Execute the following command (using the Dapr cli) to run the NotificationService:

   ```
   dapr run --app-id notificationservice --app-port 6001 --dapr-http-port 3601
   --dapr-grpc-port 60001 --config ../dapr/config/config.yaml --components-path
   ../dapr/components dotnet run
   ```

4. Open a new command-shell.
5. Change the current folder to the `src/SignalProcessService` folder of this repo.
6. Execute the following command (using the Dapr cli) to run the SignalProcessService:

   ```
   dapr run --app-id signalprocessservice --app-port 6000 --dapr-http-port 3600
   --dapr-grpc-port 60000 --config ../dapr/config/config.yaml --components-path
   ../dapr/components dotnet run
   ```

7. Open a new command-shell.
8. Change the current folder to the `src/Simulation` folder of this repo.
9. Execute the following command to run the Camera Simulation:

   ```
   dotnet run
   ```

To see the barcode scan simulation, open a browser and browse to http://localhost:5215. After the page is opened, click on **Start production** button, then you will see the scanned barcodes.

To see the emails that are sent by the SignalProcessService, open a browser and browse to http://localhost:4000.