```csharp
using Corp.Services.Contracts;
using Corp.Services.DataContracts;
using Grpc.Net.Client;
using Microsoft.AspNetCore.SignalR;
using Microsoft.Extensions.Hosting;
using Microsoft.Extensions.Logging;
using ProtoBuf.Grpc.Client;
using System;
using System.Threading;
using System.Threading.Tasks;
using static Corp.Resources.Infrastructure.Endpoints.Services;

namespace Corp.Services.SignalRHub
{
    public class Worker: BackgroundService
    {
        private readonly ILogger<Worker> logger;
        private readonly IHubContext<FloodingAlerterHub, IFloodingAlerter> floodingAlerterHub;

        public Worker(ILogger<Worker> logger, IHubContext<FloodingAlerterHub, IFloodingAlerter>
          floodingAlerterHub)
        {
            this.logger = logger;
            this.floodingAlerterHub = floodingAlerterHub;
        }

        protected override async Task ExecuteAsync(CancellationToken stoppingToken)
        {
            while(!stoppingToken.IsCancellationRequested)
            {
                logger.LogInformation("Worker starting flooding alert workflow at {Time}",
                  DateTime.Now);

                FloodingAlertWorkflowResponse workflowResult = await GetFloodingAlertData();
                string alert = ConstructAlertMessage(workflowResult);
                await floodingAlerterHub.Clients.All.ShowLatestAlert(alert);
                await Task.Delay(15000);   // 15 sec
            }
        }

        private async Task<FloodingAlertWorkflowResponse> GetFloodingAlertData()
        {
            string localHostAddress = $"http://localhost:{FloodingAlerterWorkflowPort}";
            GrpcChannel channel = GrpcChannel.ForAddress(localHostAddress);
            GrpcClientFactory.AllowUnencryptedHttp2 = true;
            FloodingAlertWorkflowResponse response;
            using(channel)
            {
                IFloodingAlertWorkflowService service =
                  channel.CreateGrpcService<IFloodingAlertWorkflowService>();
                response = await service.StartWorkflow();
            }
            return response;
        }

        private string ConstructAlertMessage(FloodingAlertWorkflowResponse r)
        {
            return $"Vandstanden er {r.WaterLevel} cm og vinden er {r.WindSpeed} m/s.";
        }
    }
}
```