DISTRIBUTED SYSTEMS

Assignment 3

Remote Procedure Call

Medication Dispenser

Chișluca Miruna

30243

Project Requirements

Suppose that each patient has an intelligent pill dispenser that can be programmed with the plan defined by the doctor. The pill dispenser will alert the patient when a medication has to be taken and when a patient did not take the medication in the corresponding time interval. The pill dispenser can communicate with the server using RPC.

Develop a client side application for the pill dispenser and a server side application to get/process the medication plan as follows: i) the pill dispenser can download the medication plan from the server; ii) the pill dispenser displays when at the current time a medication has to be taken, and a button for "Medication taken" is displayed. iii) if the button is pressed within the given time interval, the medication is marked as taken and the corresponding message is sent to the server; if the button is not pressed within the given time interval, the medication is marked as not taken and the corresponding message is sent to the server.

1. Conceptual architecture of the distributed system

RPC is a form of inter-process communication that allows executing a procedure in a different address space (on another computer or network) which is coded as if it were a normal procedure call locally. gRPC is a modern high performance RPC. For this project I have used gRPC to implement remote procedure calls.

This assignment contains 2 applications, a client and a server. These 2 applications communicate using remote procedure calls. The server is integrated in the API and the client application is a react web application.

I have added gRPC functionality to the API used for the other assignments by installing the necessary NuGet package to the solution. gRPC uses proto files to define the methods and the objects (messages) that are going to be used across the distributed systems. For implementing the pill dispenser application, my proto file defines 3 methods: GetMedicationPlan which gets a request containing a patient id for whom the medication plans for the current day must be downloaded and sends back a reply containing a list of the medications, with their start and end hours; MedicationTaken that accepts a request containing the medication id, patient id and the exact time at which the patient marked the medication as taken and sends as reply a confirmation message; MedicationMissed which accepts the same request as the other method and responds with a confirmation message. These 3 methods and the messages defined as requests and replies are implemented in a service called PillDispenserService. This service contains the actual implementation for these 3 methods.

GetMedicationPlan destructures the request and gets all the medication plans for the patientId specified in the request. Based on the date, these medication plans are filtered and only the ones that are valid for the current date are processed. If the medication plan is valid for the current date, then the medication list is traversed and new objects that contain the medication name, it's start hour and end hour are created and added to the list that will be sent as a reply.

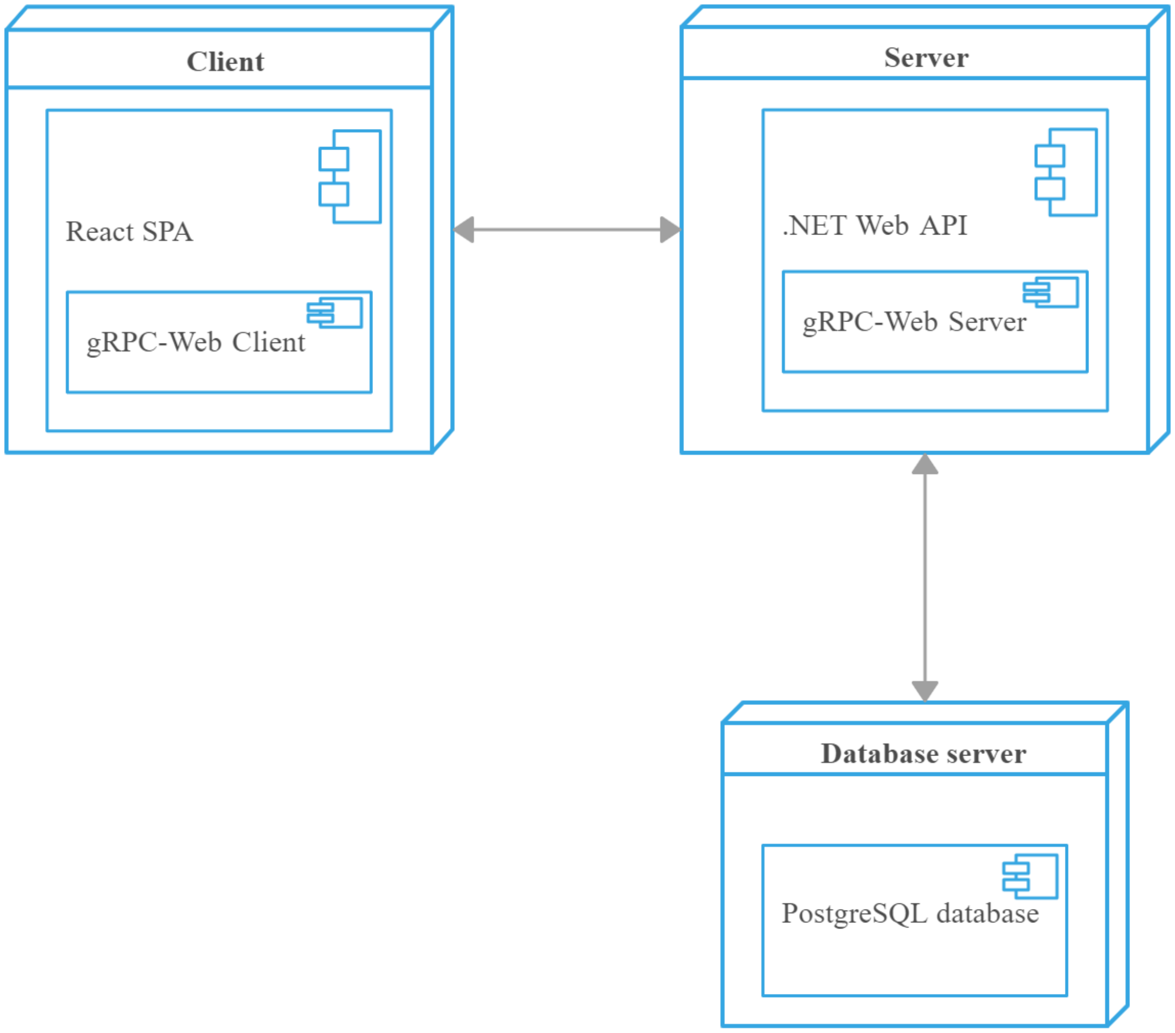
MedicationTaken gets a request containing the patient id, medication id and exact hour and inserts in the database the information provided. As a response, this method sends a response with the message "Medication marked as taken" that will appear on the client application.

MedicationMissed gets a request containing the patient id, medication id and the time that the medication was marked as missed, which corresponds to the end time of the interval in which it was supposed to be taken. The response also contains a message notifying the client that the medication was marked as missed.

The client application contains the same proto file as the server, based on which it generates 2 files containing the description of the messages and the methods. These files are generated by using the command protoc. Because gRPC is not supported in the browser (it uses HTTP2), I have used gRPC-Web in the client application that allows gRPC in the browser. The server is also configured to accept gRPC-Web requests by using a NuGet package that eliminates the need to use an additional proxy to convert gRPC-Web requests into gRPC.

The client application shows the current time and a list of the medication that the patient has to take. The list is refreshed every second. As a result, a new medication is added to this list whenever its start time is equal to the current time. When the medication's end time reached the current time, it is deleted from the list and marked as missed.

2. UML Deployment diagram



3. README

https://pilldispenser.herokuapp.com/