FIAP

Challenge Plusoft - Finder
Sistema de geolocalização e alerta em edificações

Aline Triñanes Machado - RM 84449 - alinetrim@hotmail.com

Alysson Gustavo Rodrigues Maciel - RM 86484 - alymaciel8@gmail.com

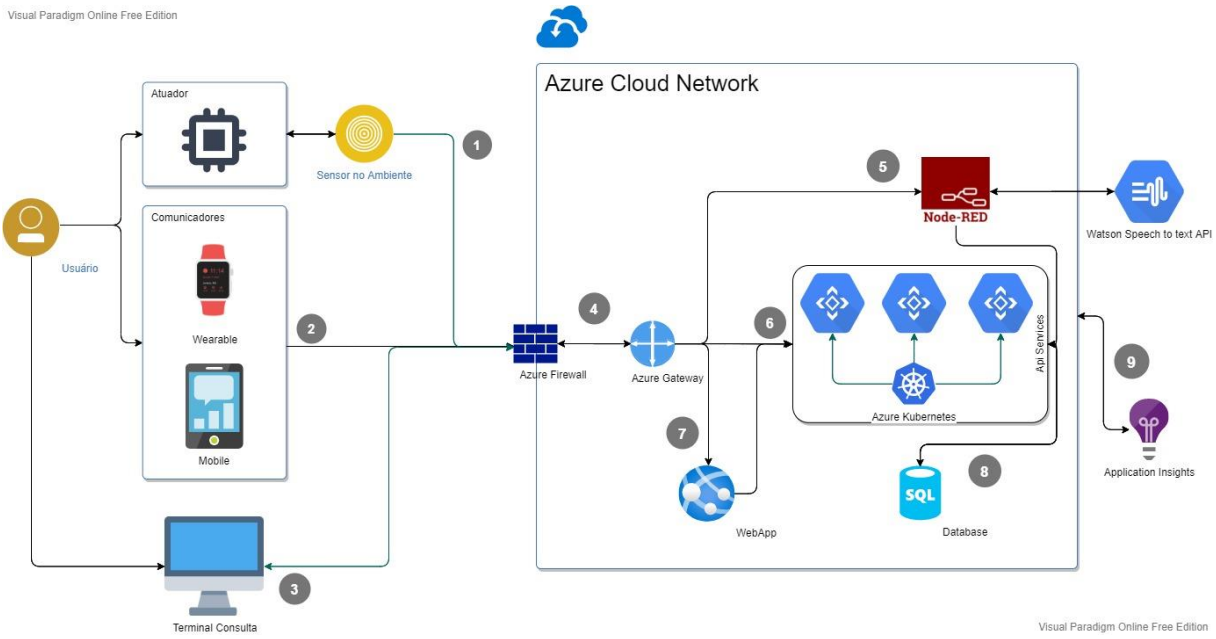
Gabriel Franham - RM 80483 - gabrielfranham@gmail.com

Gabriel Garcia Pereira - RM 86288 - gabriel.garp@outlook.com

Helouíse Cristina de Almeida Itokazo - RM 85110 -helouise.almeida93@gmail.com

Jonas Muniz de Souza - RM 84575 - jonasmzsouza@gmail.com

1. Definição da arquitetura de solução



1.1. Definição

Optamos por utilizar uma arquitetura REST e MICROSERVICES.

Utilizaremos a arquitetura REST porque esta arquitetura nos permite padronizar a interface do back-end que será consumida pelos dispositivos e suas aplicações, facilitando a comunicação, integração e transferência de dados entre eles. Utilizando a arquitetura de microsserviços, separamos as responsabilidades de cada API REST, facilitando sua manutenção, direcionando melhor as requisições de cada interface ou dispositivo e possibilitando uma escalabilidade mais eficiente.

1.2. Explicação

1 – O sensor de ambiente é responsável por captar o sinal de um RFID que o funcionário carrega consigo. Esse RFID carrega a informação do funcionário como usuário no sistema, sendo o papel do sensor de apenas transmitir esses dados para uma Azure Function responsável por atualizar a sua localização atual na base de dados.

2 – Os dispositivos wearable e mobile são responsáveis por realizar e receber solicitações de chamados, através do comando de voz do wearable ou interface mobile, fazendo requisições para os microsserviços responsáveis por esse processamento.

3 – A aplicação desktop tem como papel servir como um terminal de consultas. Suas requisições serão para os microsserviços que buscam históricos de chamados e suas informações.

4 – Através dos serviços Azure Firewall e Gateway de aplicativo de Azure, as requisições serão processadas e enviadas até a sua API de destino.

5 – O node-red será responsável por receber solicitações por voz do wearable e se comunicar com a API Watson Speech to Text, realizando o processamento do pedido por voz e então enviando ao microsserviço que inicia a solicitação de um chamado.

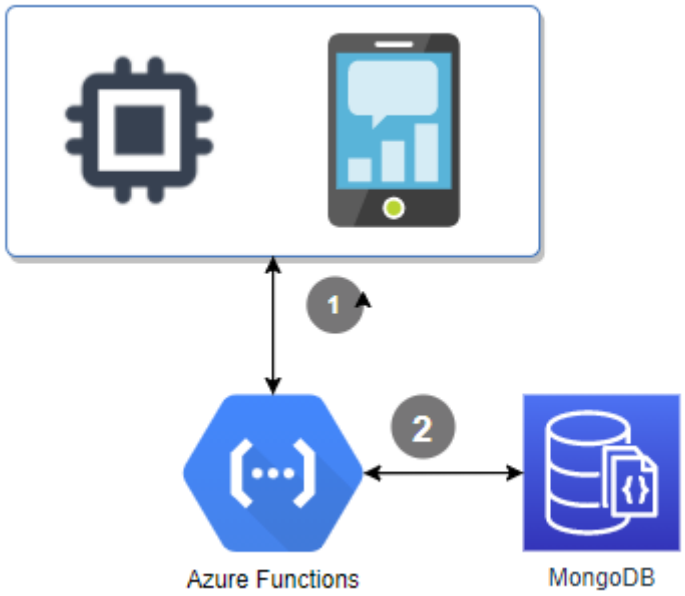
6 – Os serviços(apis) essenciais serão empacotados em contêineres e orquestrados pelo cluster Azure Kubernetes Services.

7 – A aplicação desktop ficará hospedada no serviço Azure WebApp, que fará requisições aos microsserviços de consulta.

8 – Utilizaremos o Azure SQL, serviço que será utilizado pelos microsserviços para a permanência e consulta dos dados da solução.

9 – Através do serviço Application Insights da Azure, será possível monitorar o desempenho de todo o fluxo da aplicação, facilitando a manutenção caso haja necessidade da otimização dos processos.

2 – Definição da arquitetura do mecanismo de localização



2.1 – Definição

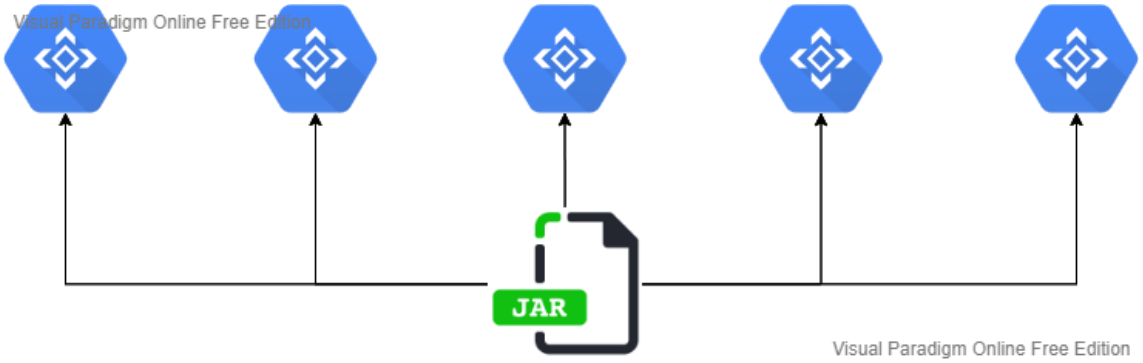
Optamos por utilizar o recurso Azure Functions disponibilizado pela Azure para realizar os procedimentos de busca e procura por oferecer escala flexível e execuções rápidas baseadas em eventos. Como tanto a atualização da localização de uma pessoa e também a busca devem ser realizadas de forma rápida, optamos por utilizar o MongoDB que ficará responsável por guardar e buscar somente essa informação.

2.2 – Explicação

1 – O sensor de RFID, wearable e o dispositivo mobile serão responsáveis por disparar os eventos que acionarão as Azure Functions de atualização de localização e busca por usuário, respectivamente.

2 – As solicitações e dados recebidos de localização serão armazenados e consultados em um banco de dados NoSQL MongoDB, que constantemente será atualizado com novas localizações e receberá requisições de consulta.

2. **Implementação do back-end**



O back-end será implementado através dos microsserviços REST que constam na arquitetura. Cada API REST terá como dependência obrigatória a mesma biblioteca que contempla todas as regras de modelagem do produto. Todas as apis devem obrigatoriamente manter essa biblioteca atualizada em sua última versão.

A biblioteca conta com os domínios principais da modelagem de dados e entidades, configuração de conexão com o banco de dados implantado no Oracle SQL, e contemplará as frameworks Hibernate, JPA e drivers de conexão com o banco.

As principais funcionalidades são:

1. CRUD de usuários;
2. CRUD de setores;
3. CRUD de cargos;
4. CRUD de ambientes;
5. Solicitação de Chamados, com a possibilidade de solicitar por nome de outro funcionário ou função (Azure Function);
6. Sistema de alertas, que notifica um usuário quando uma solicitação é feita para o seu nome, ou se atende as condições de um chamado solicitado (Azure Function);
7. Funcionalidade de consulta de histórico de chamados;
8. Funcionalidade de realizar um chamado através de comandos de voz, processados pelo node-red;
9. Identificação de localização do usuário através do sinal recebido via RFID (Azure Function);

Repositório do github: https://github.com/GarciaGP/PluFinderApi

Link Azure: https://plufinderapi.azurewebsites.net/swagger-ui/

Link Heroku: https://plufinderapi.herokuapp.com/swagger-ui/