 **INSTITUTO POLITÉCNICO DE BEJA **

**Escola Superior de Tecnologia e Gestão**

**Licenciatura em Engenharia Informática**

**Tópicos de Engenharia Informática**

**ChatBot em React**

Elaborado por:

Alexandre Pereira 17678

Mário Miguel 17693

Rodrigo Fernandes 17845

Docente:

Gonçalo Fontes

10 de abril de 2021

Table of Contents

[Introdução 3](#_Toc68988415)

[Desenvolvimento 3](#_Toc68988416)

[Backend 3](#_Toc68988417)

[“.../api/sendQuestion” 3](#_Toc68988418)

[“.../api/teachBot” 4](#_Toc68988419)

[Frontend 4](#_Toc68988420)

[Conclusão 5](#_Toc68988421)

[Bibliografia 6](#_Toc68988422)

# Introdução

O objetivo deste projeto é criar um chatbot de interface dinâmica na tecnologia React, com ligação ao motor do Wit.ai, potenciando-lhe capacidade de interação e conversão.

O chatbot tem a temática de “Hotel” onde se enquadram as interações com o bot. Por exemplo, é possível fazer reservas, informar sobre preços de quartos e serviços, ou ainda, saber informações dos espaços comuns do hotel.

O bot dá resposta a todos os pedidos relacionados com a temática, aproximando-se a “conversa” o mais possível, à que o utilizador teria com um ser humano.

# Desenvolvimento

Decidimos, primeiro de tudo, qual seria o tema do nosso *bot*, e ficou decidido que seria sobre um Hotel.

Neste trabalho utilizámos o wit.ai para definir os “intents” para essas respostas e quais seriam as “entities” presentes nas mensagens para que este pudesse responder adequadamente.

Para isso, instalamos o NodeJS e dentro da pasta do projeto criámos uma aplicação do Express dentro da diretoria “api”. Esta aplicação é criada ao criar primeiro um ficheiro “server.js” e de seguida “npm init”. Nos passos seguintes, quando é pedido o “*entry point”*, definimos o nosso ficheiro “server.js”.

Dentro do ficherio, apenas precisamos de fazer *require* do Express, obter uma variável da instância do Express e fazer *listen* a um porto.

De forma a adiantar algum trabalho, criámos a aplicação *Frontend* React através do comando “npx create-react-app app” na pasta mãe.

## Backend

Para efetuar a leitura do ficheiro JSON utilizámos a biblioteca “fs” e colocámos o ficheiro que queremos ler na diretoria “data” do *Backend.*

Após analisarmos, decidimos que a API teria apenas dois *enpoints*: um que permitisse enviar e receber uma resposta e um que permitisse ensinar o *bot*.

O JSON é composto por uma mensagem genérica, para quando o *bot* não encontra a pergunta; uma mensagem de erro, uma mensagem de agradecimento, para quando o utilizador está a ensinar o *bot*; e uma lista de perguntas-resposta, onde cada pergunta-resposta contém uma lista de perguntas para a qual existe uma resposta.

### “.../api/sendQuestion”

Este *endpoint* recebe um JSON com um atributo chamado “*question*”, que transporta o *input* do utilizador. Este objeto é obtido, lido e depois é procurado no JSON do *Backend* por uma pergunta que tenha correspondência com, pelo menos, três quartos do *input* do utilizador.

### “.../api/teachBot”

Neste *endpoint*, o utilizador consegue “ensinar” o *bot*. Isto é possível pois é guardado o JSON original e é adicionado à lista de perguntas a pergunta e a resposta que o utilizador definiu. Estes campos são recebidos por JSON, com os atributos “*question*” e “*answer*”.

## Frontend

Para o *Frontend*, decidimos utilizar componentes do Material UI, para estarmos a par com os designs modernos.

De forma a mantermos o aspecto de chat, utilizamos o componente Grid, pois permite manter uma proporção igual em todos os tamanhos de ecrã.

Criámos 3 componentes principais: Header, Body e Footer. Dentro do Header colocámos o nome do *bot*, no Body o *chat* e no Footer um pequeno texto de informação.

No chat, para cada a lista de mensagens foi criado um componente e para cada mensagem foi criado um componente também. O *input* do utilizador é também um componente à parte.

Para termos controlo sobre o estado de cada componente, o estado da lista, o estado dos *inputs*, utilizámos os React Hooks, mais concretamente o UseState e os Props.

# Conclusão

Este trabalho foi ótimo para termos uma interação com o React e o Express simultaneamente, sendo que apenas faltaria uma componente de base de dados para ser um trabalho “Full Stack”. Conseguimos apreender mais conhecimento acerca de Express, pois tivemos alguns problemas com o POST *request* de um formulário, e também aprofundamos mais o nosso conhecimento acerca de CSS, pois tivemos bastante cuidado em termos a certeza que a aplicação ficava apresentável em modo *mobile*.

# Bibliografia

https://material.io/components?platform=web