Uma imagem com desenho

Descrição gerada automaticamente

**Instituto Politécnico de Beja**

**Escola Superior de Tecnologia e Gestão**

**Licenciatura de Engenharia Informática**

**Tópicos de Engenharia Informática**

**ChatBot com React e NPL**

**Alexandre Pereira 17678**

**Mário Miguel 17693**

**Rodrigo Fernandes 17845**

**Beja**

**2020/2021**

**Instituto Politécnico de Beja**

**Escola Superior de Tecnologia e Gestão**

**Licenciatura de Engenharia Informática**

**Tópicos de Engenharia Informática**

**ChatBot com React e NPL**

**Elaborado por:**

Alexandre Pereira 17678

Mário Miguel 17693

Rodrigo Fernandes 17845

**Orientado por:**

**Prof. Gonçalo Fontes**

**Beja**

**2020/2021**

**Conteúdo**

[**Capítulo I – Introdução** 4](#_Toc72079229)

[**Capítulo II – Front-End** 5](#_Toc72079230)

[**Capítulo III – Back-End** 6](#_Toc72079231)

[**III.1** **Wit.ai** 6](#_Toc72079232)

[**III.1.1**  **Intents e Entities do Wit.ai** 6](#_Toc72079233)

[**III.1.2**  **Pedido ao Wit.ai** 7](#_Toc72079234)

[**III.2** **Script responseHandler.js** 8](#_Toc72079235)

[**III.2.1**  **Saudações** 8](#_Toc72079236)

[**III.2.2**  **Reservar Quartos** 9](#_Toc72079237)

[**III.2.3**  **Alterar Reserva** 9](#_Toc72079238)

[**III.2.4**  **Cancelar Reserva** 9](#_Toc72079239)

[**III.2.5**  **Espaço do Ginásio** 9](#_Toc72079240)

[**III.2.6** **Espaço da Piscina** 10](#_Toc72079241)

[**III.3** **Ficheiro Chatdata.json** 11](#_Toc72079242)

[**Capítulo IV – Correr o Projeto** 12](#_Toc72079243)

[**I.1.2**  **Instalar os Pacotes** 12](#_Toc72079244)

[**I.1.3**  **Correr a App e a Api** 12](#_Toc72079245)

[**Capítulo v – Conclusão** 13](#_Toc72079246)

# **Capítulo I – 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 *Chatbot*. Por exemplo, é possível fazer reservas, informar sobre preços de quartos e serviços, ou ainda, saber informações dos espaços comuns do hotel.

O *Chatbot* 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.

# **Capítulo II – Front-End**

Para o Front-End utilizamos todo o trabalho feito anteriormente na primeira parte, apenas alterando a imagem de fundo, o texto e os ícones.

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.

O website ficou com o aspeto apresentado na figura X.

Graphical user interface

Description automatically generated

# **Capítulo III – Back-End**

No back-end utilizamos o wit.ai, o Express e o JSON.

## **III.1 Wit.ai**

Após uma breve análise realizada por parte do grupo, concordámos que seria melhor começar pelo *Wit.ai*.

### **III.1.1 Intents e Entities do Wit.ai**

No wit.ai escolhemos quais eram os *Intents* para os tópicos de conversa que o utilizador pode ter com o *Chatbot* e quais as *Entities* que estes utilizam, Na figura X estão os *Intents* e os respetivos *Entities* que colocamos inicialmente.

Table

Description automatically generated

### **III.1.2 Pedido ao Wit.ai**

Para enviar o que o utilizador escreve no chat para o wit.ai é utilizado o *endpoint* “/api/sendQuestion” no ficheiro *server.js*, apresentado na figura X, que faz o *get* com o parâmetro *question* e o valor do texto escrito.

Depois, é recebido um *JSON* com o respetivo *intent* relativo ao texto e a lista das *entities* que nele estão e o script responseHandler.js trata da resposta para ser dada ao utilizador.

Text

Description automatically generated

## **III.2 Script responseHandler.js**

Neste Script é onde as respostas ao utilizador são escolhidas e onde está toda a logica correspondente á sequencia do texto apresentado ao utilizador como resposta ao que este diz.

Text

Description automatically generatedNeste script existe um *switch-case* que chama uma função diferente para cada tópico de conversa identificado pelo *Intent* recebido pelo wit.ai como está apresentado na figura X.

### **III.2.1 Saudações**

No caso das saudações é utilizado a função *handleGreetings, apresentada na figura X que já sabendo qual é o tópico afirma que a resposta a devolver será uma das respostas, escolhida de forma aleatória, presentes no ficheiro JSON onde estão as respostas possíveis para o utilizar para aquele Intent.*

Text

Description automatically generated

### **III.2.2 Reservar Quartos**

### **III.2.3 Alterar Reserva**

### **III.2.4 Cancelar Reserva**

### **III.2.5 Espaço do Ginásio**

No caso do utilizador fizer perguntas relativas ao ginásio é utilizada a função *handleGymAcess* apresentada na figura X. Nesta função são identificados quais são as *Entities* presentes no texto que o utilizador escreveu e dá uma das respostas que estão no ficheiro *JSON* de acordo com a pergunta. Sendo estas sobre o preço de entrada, sobre as coisas que há no ginásio ou sobre o ginásio em geral.

Text

Description automatically generated

### **III.2.6 Espaço da Piscina**

No caso do utilizador fizer perguntas relativas á piscina é utilizada a função *handlePoolAcess* apresentada na figura X. Nesta função são identificados quais são as *Entities* presentes no texto que o utilizador escreveu e dá uma das respostas que estão no ficheiro *JSON* de acordo com a pergunta. Sendo estas sobre o preço de entrada, sobre as coisas que há na piscina ou sobre a piscina em geral.

Text

Description automatically generated

## **III.3 Ficheiro Chatdata.json**

Neste ficheiro JSON estão guardadas todas as respostas que são dadas ao utilizador de acordo com a conversa que ele tem com o *Chatbot*. Como está apresentada na imagem X que tem um pequeno trecho do código este documento tem uma resposta pré-definida para quando o *Chatbot* não entende a pergunta e depois está organizado por *Intents em que dentro de cada um tem um objeto com várias respostas para cada Entitie.*

Text

Description automatically generated

# **Capítulo IV – Correr o Projeto**

Para instalar os pacotes no projeto foi utilizado o npm. Para correr a API foi utilizado o node.

### **I.1.2 Instalar os Pacotes**

Para instalar os pacotes basta fazer “npm install” nas diretorias “…\npl-chatbot\api>” e \npl-chatbot\app> como está exemplificado na imagem X.

Text

Description automatically generated

Para fazer o auth do wit.ai também é necessário estar a pasta “.env” dentro da diretoria \app com o seguinte texto: “ WIT\_TOKEN=UZK7JIWKVTJH37QYGFUZ36PYKNK7JDVZ ”.

### **I.1.3 Correr a App e a Api**

Para correr a app basta estar dentro da pasta \app e fazer o comando “npm start”.

Para correr a api basta estar dentro da pasta \api e fazer o comando “nodemon server.js”.

Text

Description automatically generated

# **Capítulo v – Conclusão**

Texto TextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTexto Texto

Texto TextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTexto Texto

Texto TextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTexto Texto

Texto TextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTextoTexto Texto