CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA
FACULDADE DE TECNOLOGIA DE BOTUCATU
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

JEAN VICTOR MENDES DOS SANTOS

VICTRO ROBOT – DESENVOLVIMENTO DE UM CHATBOT INTELIGENTE UTILIZANDO RASPBERRY PI

Botucatu-SP
Agosto – 2017

CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA
FACULDADE DE TECNOLOGIA DE BOTUCATU
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

JEAN VICTOR MENDES DOS SANTOS

VICTRO ROBOT – DESENVOLVIMENTO DE UM CHATBOT INTELIGENTE UTILIZANDO RASPBERRY PI

Orientador: Prof. Esp. Rogério Ferreira Sgoti
Co-Orientador: Prof. Marco Antonio Nagao

Projeto de Conclusão de Curso apresentado à FATEC - Faculdade de Tecnologia de Botucatu, para obtenção do título de Tecnólogo no Curso Superior de Análise e Desenvolvimento de Sistemas.

Botucatu-SP
Agosto – 2017

SUMÁRIO

Página

LISTA DE IMAGENS

LISTA SIGLAS

1 INTRODUÇÃO

Um chatbot é um software desenvolvido para simular uma conversação inteligente com um ou mais usuários, esta conversa pode acontecer através de texto ou áudio. Chatbots podem ser programados para manter conversas básicas como a do dia-a-dia de uma pessoa ou até responder respostas especificas sobre algum tema programado anteriormente. O chatbot entende o contexto do que foi dito/escrito e entrega uma resposta baseada na mensagem entendida. Chatbots é um exemplo de inteligência artificial pois podem aprender outras formas de entender o contexto e assimilar respostas dadas anteriormente. (GUPTA, S. et al, 2015).

Inteligência artificial ou IA, é uma área de conhecimento humano que busca uma forma de reproduzir a mente humana utilizando elementos computacionais (FERREIRA, L. P., & Uchôa, J. Q, 2008). Não faz parte do projeto apresentado as definições e aplicações de AI nos dias atuais. Para consultar definições e aplicações vide (SELLITTO, M. A, 2002).

A tecnologia esta cada vez mais presente em nossas vidas, com isso tentamos encontrar formas para encurtar e facilitar processos, desta forma o reconhecimento de voz é muito bem-vindo para a maioria das pessoas, sendo a linguagem um processo natural de comunicação. O reconhecimento de voz é um recurso cada vez mais adotado nos desenvolvimentos de praticamente todas plataformas, principalmente para desenvolvimento de aplicativos moveis. Este recurso esta em constante aprimoramento e seu custo esta cada vez menor, tornando mais acessível à maioria das pessoas. (RATO, J. P. C., 2016).

O presente trabalho propõe o desenvolvimento de um chatbot inteligente utilizando hardware de baixo custo. O nome adotado para o chatbot é Victro Robot.

O projeto trata de um chatbot que é resultado da combinação hardwares e softwares cujo sua principal funcionalidade é reconhecimento de voz e controle de hardwares externos. Para o reconhecimento de voz há o recurso de alteração de humor, esta alteração acontece de acordo com a frase que o chatbot escutou, através disto há uma analise de tom e um vetor de emoções é recebido. Ele escuta o que foi dito, pesquisa em seu banco de dados procurando por uma ocorrência com a frase dita e responde. Inicialmente ele entenderá apenas frases ditas em inglês mas há possibilidades de gerar bibliotecas em português e outros idiomas. O hardware que dará suporte ao referido chatbot será o Raspberry Pi

O Raspberry Pi é um hardware desenvolvido pela Fundação Raspberry Pi e sua primeira versão foi lançada em 10 de fevereiro de 2012. O principal objetivo deste hardware é promover o ensino de Ciências da Computação em escolas e faculdades mais popular em países de primeiro mundo, mas este tipo de tecnologia esta se tornando conhecida no Brasil. Ele é um microcomputador capaz de fazer processamento de voz com 1GB de memória RAM, além de ter a capacidade de gerenciar componentes e sensores através de GPIO e USB. Pode ser programado em diversas linguagens como Python, PHP, Java, entre outras. (SIMIONI, M. C.; BETINI, R. C, 2014).

VictroBrain é o software que comanda o ChatBot, ele é um Framework PHP desenvolvido especialmente para este projeto, mas a sua aplicação pode ocorrer em diversos outros tipos de projetos. A definição de Framework tem-se como um software “base” onde pode-se ser feito o desenvolvimento de outro software utilizando funções, técnicas e metodologias do framework, com isto o desenvolvimento é mais rápido e segue um padrão (MINETTO, E. L, 2007).

1.1 OBJETIVO

O objetivo desta pesquisa é desenvolver um chatbot inteligente que terá similaridade com o comportamento humano no que tange conversação, sendo um assistente virtual respondendo por comandos de voz. Ele também terá a capacidade de manter uma conversa em inglês com uma pessoa, falar sobre assuntos variados e controlar hardwares externos compatíveis com o *Framework* VictroBrain.

1.2 JUSTIFICATIVA E RELEVÂNCIA DO TEMA

Através da inteligência do Victro, as pessoas poderão ter um chatbot programável que ajudará a gerenciar o seu dia a dia, interagir com os demais equipamentos e facilitar toda a organização de uma casa, família, empresa.

3 REVISÃO DE LITERATURA

Os avanços tecnológicos permitiram que os mecanismos de reconhecimento de voz oferecessem mais potencia e mais sensibilidade, com retornos mais precisos numa multiplicidade de ambientes (BARBARINI, 2008).

A sigla PHP significa Pré-Processador de Hipertexto. De acordo com Pereira e Poupa (2005) o PHP é uma linguagem de programação que gera arquivos em HTML, e é executado no lado servidor, ou seja, todo processamento é feito antes que a página HTML apareça para o usuário.

Segundo STURZA, é uma boa prática usar o SGBD MySQL pois ele é um banco de dados relacional de código aberto mais usado no mundo, segundo o site oficial da Oracle (2016). Também é afirmado que empresas grandes como Facebook, Google e Adobe usam o SGBD em seus sites.

3 MATERIAL E MÉTODOS

3.1 Raspberry Pi 2

Figura 1 -Raspberry Pi 2

3.2 VictroBrain

Para o processamento e armazenamento de dados foi criado um sistema que inicialmente seria apenas para lidar com dados do robot mas no seu desenvolver as funções foram adaptadas para serem genéricas e ajudar qualquer tipo de desenvolvedor web que deseja trabalhar com PHP, então o sistema se tornou um framework chamado VictroBrain, com ele o código fica mais simples e há uma abstração de códigos complexos como interação com o banco de dados. Com ele é possível fazer conexões em vários tipos de banco de dados, mas o recomendado é o MySql. No projeto o framework atua como um core, onde toda requisição é tratada e também

as requisições de componentes externos, utilizando Arduino ou NodeMCU ESP8862. Esta requisição para componentes externos chama-se VictroAddon e é possível através do próprio framework, no projeto as requisições poderão ser feitas através de comandos de voz do Victro Robot. O framework foi desenvolvido com a linguagem de programação web PHP orientada à objetos e pode ser baixado pelo site victrobrain.com para desenvolvimento de sites e sistema de pequenos até grandes portes.

3.2.1 Instalação

O processo de instalação do framework é bem simples pois não é necessária nenhuma programação, todo o processo é visual e interativo ao usuário.

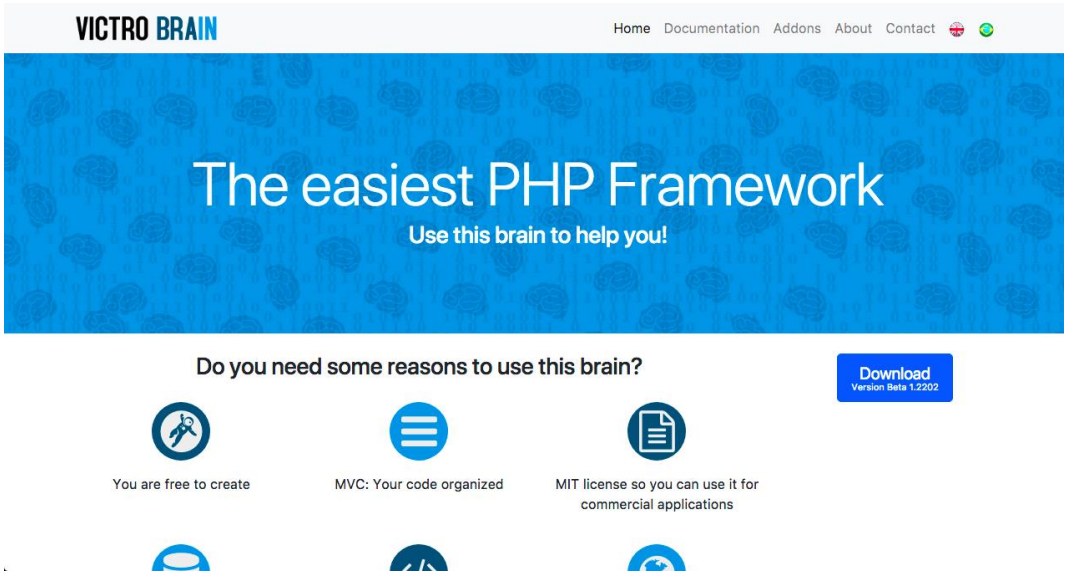

Figura 2- Site oficial para download do framework

É preciso entrar no site oficial do framework, victrobrain.com e clicar no botão Download para baixar a ultima versão do sistema. Será baixado um arquivo tar.gz com o sistema, é necessário um servidor Linux com PHP 7.1 ou posterior e um sistema gerenciador de banco de dados (SGBD) compatível com o sistema. Após isto enviar o arquivo baixado para o servidor Linux, descompactar e então acessar pela url o caminho em que o framework esta.

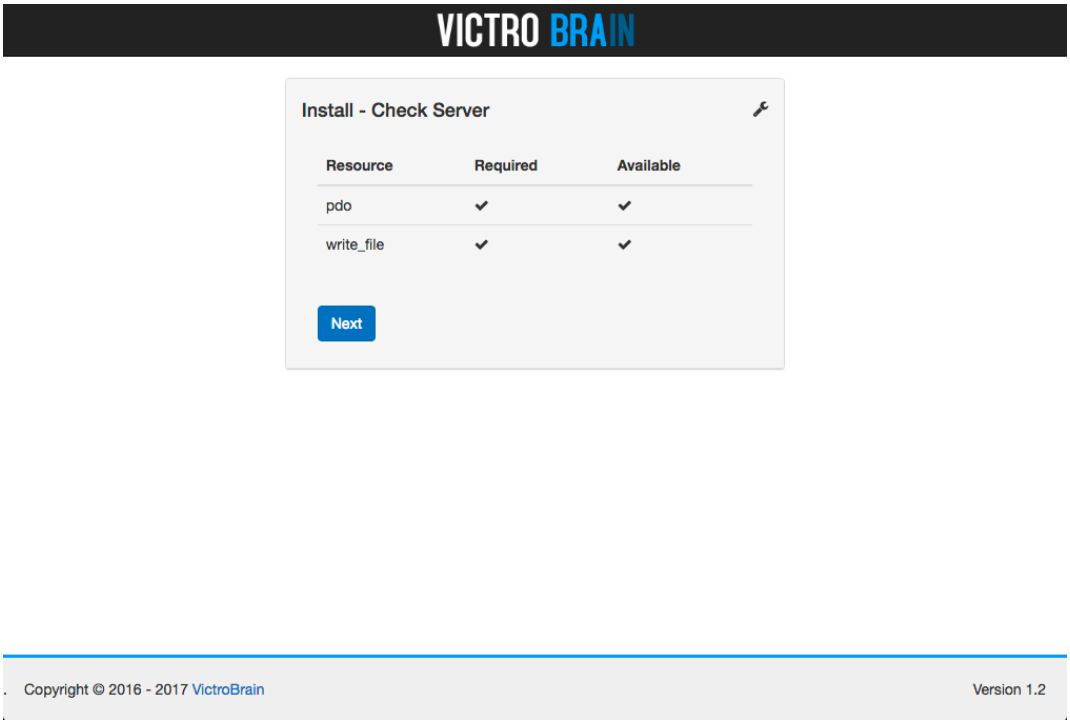

Figura 3 - Verificando funções do servidor

Ao requisitar a url do framework no servidor ele ira detectar que ainda não foi instalado e então passa pela verificação de recursos do servidor. Esta verificação funciona para que o framework trabalhe sem falhas e utilizando todos seus recursos. Alguns dos recursos que o framework exige é o PDO, um modulo PHP para conexão com banco de dados, o mod_rewrite que serve para abstrair a extensão do arquivo, ou seja, ao invés de chamar por '/index.php' só ira chamar '/index' e a possibilidade do framework escrever arquivos na pasta que foi colocando, isto para que ele consiga gerar o arquivo de instalação do banco de dados e outros arquivos que só são gerados a partir da instalação.

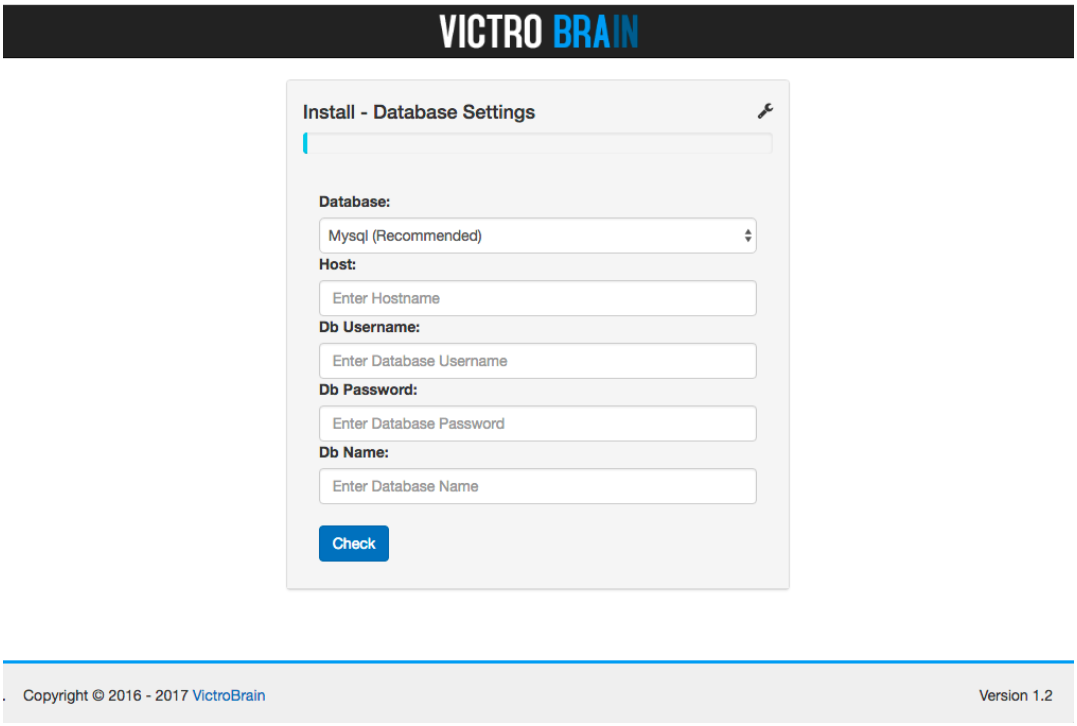

Figura 4 - Configurando banco de dados no framework

Após o framework verificar que os requisitos mínimos estão ativos e clicando em ‘next’ o framework pedira por dados de conexão do banco de dados. O SGBD pode ser selecionado pelo usuário de acordo com sua preferencia, podendo ser Mysql, Firebird, Oracle, MSSql, PostgreSQL e SQLite. O framework recomenda o uso do Mysql por sua maior facilidade. Selecionando o SDBD é necessário informar o host que o banco de dados se encontra, também o usuário do banco de dados, senha e o nome do banco de dados. Após preencher todos os campos e clicar em ‘check’ o framework ira verificar se é possível fazer conexão e se o banco de dados está vazio – é necessário que o banco de dados de instalação esteja vazio – caso aconteça algum erro o sistema ira informar o erro acima do logo do framework.

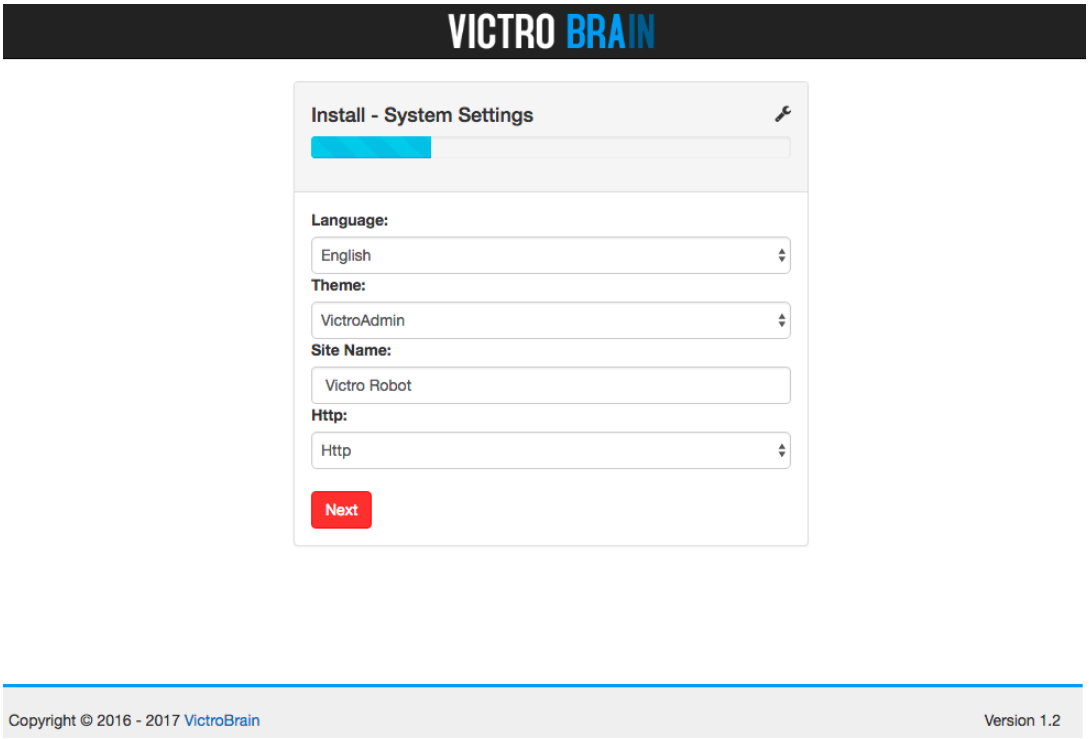

Figura 5 - Informação sobre o sistema

Ao verificar o banco de dados e tudo estiver certo o sistema pedirá dados essenciais para o sistema como idioma, tema (design), nome do site/sistema e o tipo de conexão que será utilizado http ou https. Após preencher todos os campos clique em “next”.

Figura 6 - Informações do super usuário

Em seguida o sistema precisará de informações do super usuário, nesta etapa é necessário inserir o nome, usuário, email, senha (com confirmação) e uma foto do usuário.

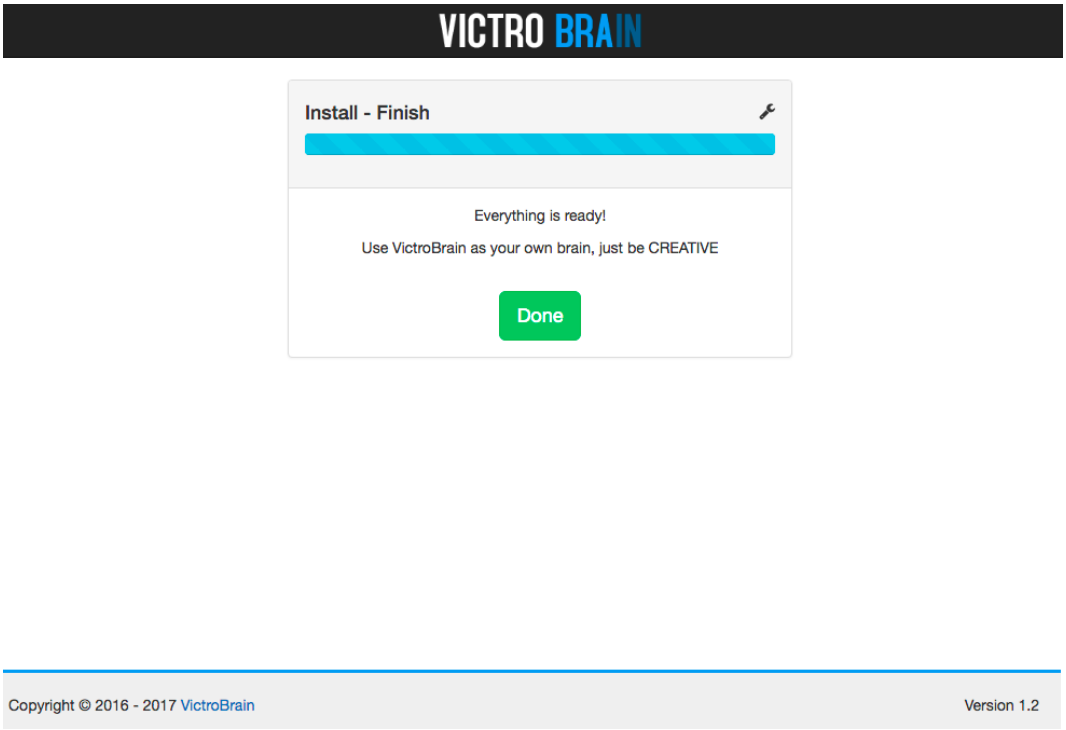

Figura 7 - Conclusão de instalação

Após preencher todos os dados o sistema irá informar que o processo de instalação foi finalizado. Clicando em 'Done' o sistema ira mover os arquivos criados na instalação para seus devidos lugares e o sistema ira carregar a tela de login de usuário.

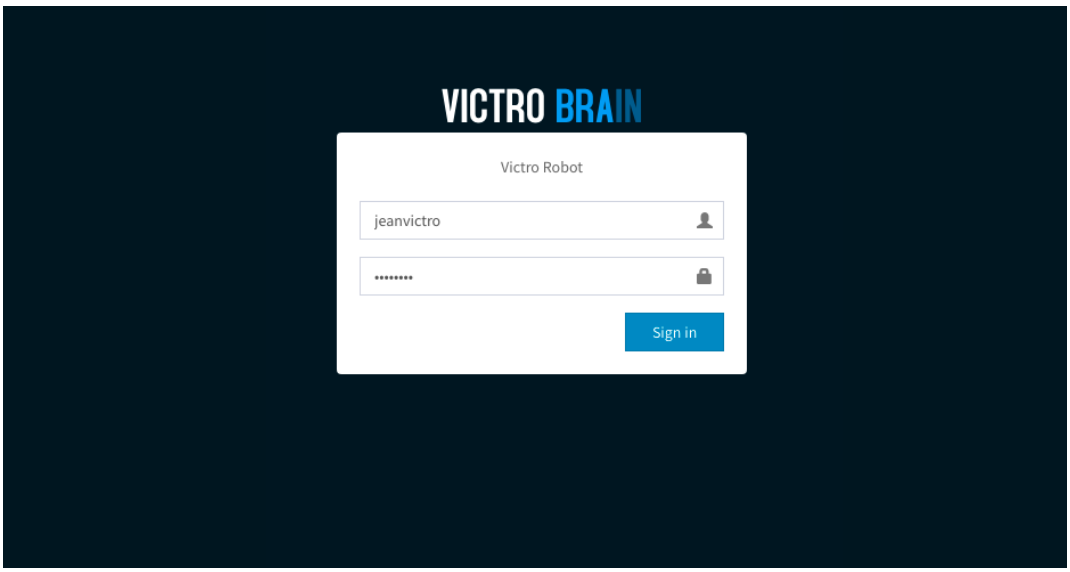

Figura 8 - Tela de login

Na tela de login, após a instalação o usuário precisará entrar com o usuário digitado na etapa de informação de usuário e a senha também cadastrada nesta etapa.

3.2.2 Conhecendo o sistema

Figura 9 - Tela inicial

A tela inicial do framework é dividida em 4 partes: menu lateral esquerdo, menu superior, menu lateral direito (oculto) e o carregador de conteúdo. Na tela inicial é possível visualizar os componentes básicos do sistema, este design pode ser carregado ou não, ou seja, caso o programador queria desenvolver outro design para sua aplicação o design padrão não será exibido. De acordo com a figura 8 os componentes gráficos são:
1 – Busca integrada do sistema, esta busca é capas de buscar por menus ou funções do sistema.
2 – Lista de menus, onde os plug-ins instalados com menu serão visíveis
3 – Ícone para diminuir o menu e ter maior aproveitamento da tela
4 – Ícone de notificações com indicador de quantidade de notificações
5 – Menu para usuário logado onde é possível editar informações do usuário e fazer logout do sistema
6 – Icone para abrir o menu direito do sistema onde é possível alterar configurações inseridas na instalação do sistema, assim como instalar plug-ins e atualizar o sistema
7 – Área onde é carregado os plug-ins que utilizam o design do framework
8 – Informação sobre a função do framework

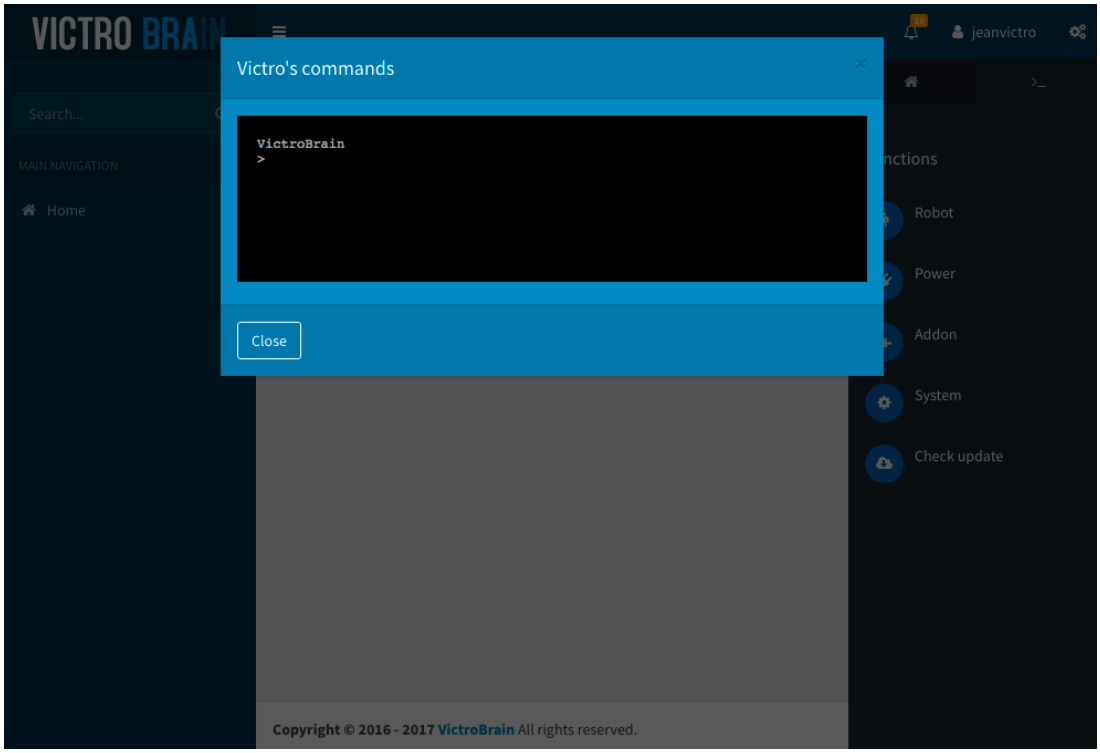

Figura 10 - Terminal do sistema

Os comandos do framework são executados através de um terminal próprio do sistema onde é possível alterar informações do sistema como titulo, local de instalação, usuário padrão, entre outras informações e também instalar plug-ins, helpers e add-nos.

3.2.3 Desenvolvimento

3.2.3.1 Plug-ins (*Robot*)

Os Plug-ins são a principal parte do sistema, eles são chamados de *robot* no sistema. O *robot* é o produto que se deseja desenvolver, ou seja, o software em si. No VictroRobot foi criado um *robot* para detecção de fala, pois esta é uma grande função, para pequenas funções é utilizado os *helpers* que são chamados de *power* no sistema.
Para se desenvolver um robot no sistema é necessário que algumas regras/padrões de desenvolvimento sejam seguidas. Primeiramente é necessário criar a pasta do *robot* dentro da pasta victro_apps/victro_robot, o nome da pasta não poderá conter espaços ou caracteres especiais. Dentro da pasta criada para o *robot* é necessário mais 2 pastas no mínimo, uma chamada *model*, onde são armazenados os arquivos que lidam com o banco de dados, sessões e outras funções, outra chamada *view* que armazenará arquivos de visualização para o usuário. Também é necessário um arquivo de instalação do *robot* chamado *install*.php onde há todas as informações sobre o *robot* como nome, versão, autor, tabelas do banco de dados, menus e outras informações e por ultimo o arquivo *robot*.php que é o controlador do *plugin*, ou seja, as funções chamadas no browser devem estar contidas neste arquivo. Todos os arquivos do *robot* são PHP orientado à objetos. A distribuição dos arquivos deve ficar iguais ao da figura 10.

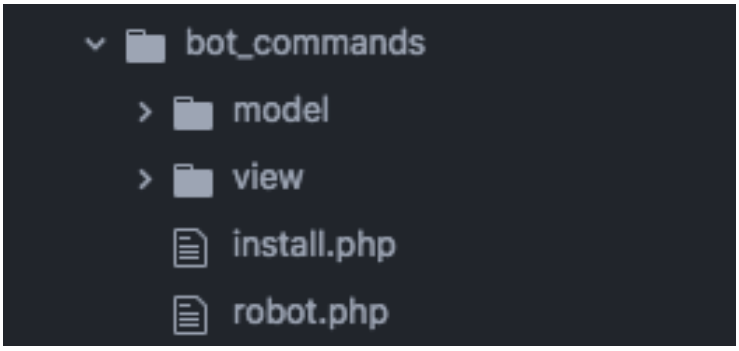

Figura 11 - Distribuição dos arquivos no *robot*

No arquivo de instalação são necessárias algumas informações para que o sistema possa instalar o robot seguramente. Começando pelas informações contidas na figura 11.

```
3  $victro_robot->name("BotCommands");
4  $victro_robot->try_route("robot");
5  $victro_robot->try_route("ro_bot");
6  $victro_robot->author("Jean Victor");
7  $victro_robot->description("This plugin can hear everything");
8  $victro_robot->version(1.0);
9  $victro_robot->icon("fa fa-android");
10 $victro_robot->update("http://victrobot.com/update/[KEY]");
```

Figura 12 - Informações sobre *robot*

Segundo a figura 11, na linha 3 é definido o nome do *robot*, seguindo pela linha 4 podemos ver a função '*try_route'*, esta função tenta criar uma rota (link de fácil memorização) para o *robot* e pode ser usada mais de uma vez no *plugin*, assim que o sistema encontra a primeira rota que não esta em uso ele ira ignorar as outras. Na linha 5 é definido o nome do autor do *robot*. Linha 7 uma pequena descrição do que o *robot* fará no sistema. Linha 8 definirá a versão do *robot*. Linha 9 (opcional) definira um ícone para o *robot* e linha 10 definirá a *url* de atualização para o *robot* (opcional).

```
//TABLE INFORMATION
$victro_robot->table("PHRASE")->engine("INNODB")->if_table("NOT EXISTS");
$victro_robot->column("ID")->type("INT")->value("11")->autoincrement(true)->index("PRIMARY KEY");
$victro_robot->column("PHRASE")->type("varchar")->value("255");
$victro_robot->column("EMOTION_ANGER")->type('float')->value('5');
$victro_robot->column("EMOTION_DISGUST")->type('float')->value('5');


//MENU INFORMATION
$victro_robot->menu("Robot", "1", "fa fa-user")->submenu('Hear', '1')->submenu('Feelings', '5');
```

Figura 13 - Informação de instalação de tabelas no banco de dados

Na figura 12 é possível observar os comandos necessários para criar uma tabela no banco de dados. Todos os comados SQL são abstraídos para comandos PHP do sistema. Abaixo na criação da tabela esta sendo declarado os menus e submenus que este *robot* criará no sistema.

```php
class BotCommands extends controller_robot {
    public function hear(){
        $phrase = $this->input("PHRASE", "POST_GET");
        if($phrase != false){
            if($this->functions($phrase) == true){
                echo $this->conversation("yes");
            } else {
                echo $this->conversation($phrase);
            }
        } else {
            $this->system_view();
        }
    }
}
```

Figura 14 - Exemplo de robot.php

Na figura 13 é possível observar como é construída a classe de um *robot*.php, a classe deve ter o nome do *robot* que foi colocado no arquivo de instalação e deve estender a classe *controller_robot* para que possa se comunicar com o sistema.
Com todos os arquivos programados basta ir ate o terminal do sistema e executar o comando:

- install_bot <nome da pasta do *robot*>: para instalar o *robot*
- update_bot <nome da pasta do *robot*>: para atualizar o *robot* com novas informações preenchidas no arquivo de instalação.

3.2.3.2 *Helper* (*Power*)

O helper é chamado de power no sistema e sua principal função é o uso de pequenas funções para agregar nos robots (plug-ins). Powers não podem inserir dados no banco de dados, mas podem ter sua própria tabela ´fazer consultas para os robots. O desenvolvedor devera informar que o robot necessita de um power para funcionar adequadamente. Para a criação de um power basta criar uma pasta dentro de victro_apps/victro_power com um arquivo PHP chamado function.php, assim como na figura 14.

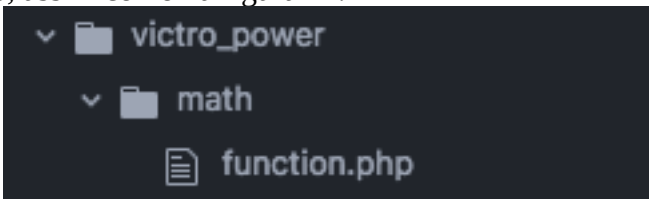

Figura 15 - Distribuição de arquivos para powers

```php
$victro_power->name("math");
$victro_power->author("Jean Victor");
$victro_power->version(1.0);
class math extends power {
  public function sum($num1, $num2){
    return $num1 + $num2;
  }
}
```

Figura 16 - Programação de um *power*

Na figura 15 pode-se observar que a primeira linha é a definição do nome do *power*, seguido pelo nome do autor e sua versão. Logo é necessário criar uma classe PHP com o mesmo nome definido no *power* e estender a classe *power* para a comunicação com o sistema.
Com o arquivo do power preparado basta acessar o terminal do sistema e executar o comando abaixo que se adeque ao uso:

- *install_power* <nome da pasta do *power*>: instala o *power* no sistema e cria suas tabelas no banco de dados caso isso seja definido.
- *update_power* <nome da pasta do *power*>: Atualiza as informações do *power* de acordo com o arquivo de instalação.
- *match_power* <nome da pasta do *power*> <nome da pasta no *robot*>: ativa o *power* para um *robot*, este mesmo *power* pode ser adicionados para qualquer outro *robot* instalado no sistema.
- *unmatch_power* <nome da pasta do *power*> <nome da pasta no *robot*>: desativa o *power* para o *robot* especificado.

3.2.3.3 *Add-nos*

Add-ons são hardwares externos que podem interagir com o framework através de redes WI-FI. O Hardware externo precisa estar utilizando um código fornecido pelo próprio framework para que esta interação aconteça. O modulo aconselhado pelo framework é o NodeMCU ESP8266, sua linguagem é LUA e o framework disponibiliza o código para ser executado no modulo e deixa-lo compatível com o sistema. Os add-ons são controlados através de um robot (plugin). A junção de add-ons e robot pode resultar em uma automação residencial, por exemplo.

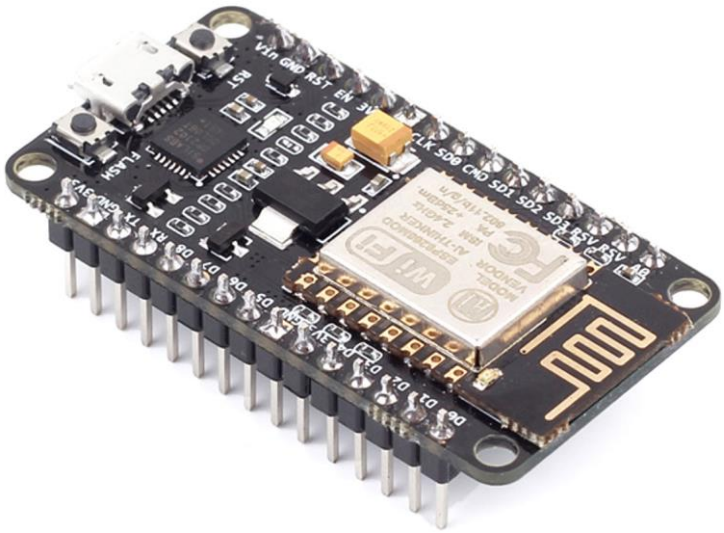

Figura 17 - Modulo NodeMCU ESP8266

Também é possível utilizar Arduino ou Raspberry PI para esta comunicação, mas ainda sim é necessário que eles possuam conexão WI-FI para enviar infomações ao framework.

Figura 18 - Arduino UNO

3.3 Reconhecimento de voz

3.3.1 Softwares existentes

- **Siri (Apple):** A integrante mais antiga quando se trata de assistente pessoal com reconhecimento de voz, tem seu aprimoramento a cada nova versão do sistema IOS. Ela atualmente pode receber comandos de voz tanto para funções básicas do sistema como enviar um SMS como também pode executar funções em aplicativos de terceiros como executar uma musica no Spotify ou enviar uma mensagem pelo

WhatsApp. A Siri pode ser executada tanto no iPhone quando no iPad, iWatch e computadores Apple.

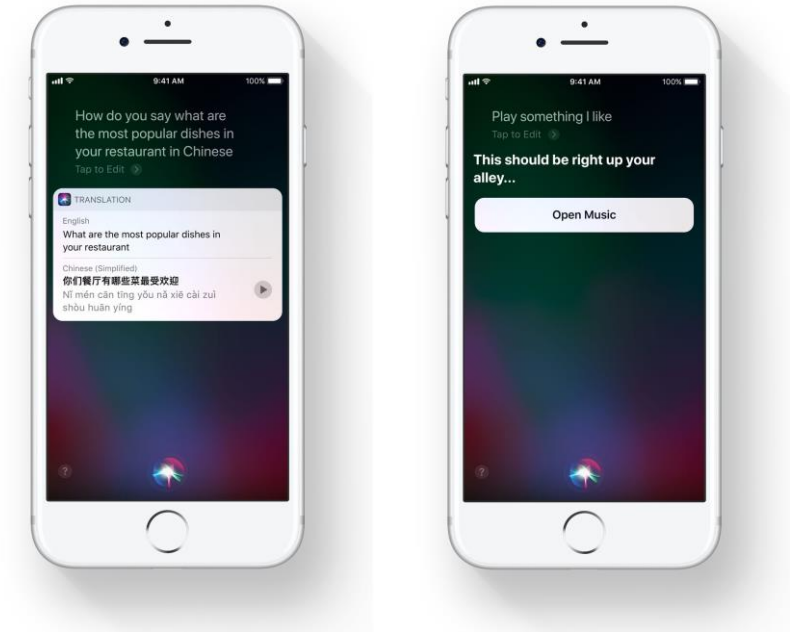

Figura 19 - Siri em execução

- **Cortana (Microsoft):** Lançada junto com o Windows 8.1 a assistente pessoal da Microsoft tem grande expectativa do mercado. O ponto forte da Cortana é sua capacidade de falar frases inteiras com uma naturalidade. Suas principais funções são chamadas, previsão do tempo e alarmes.

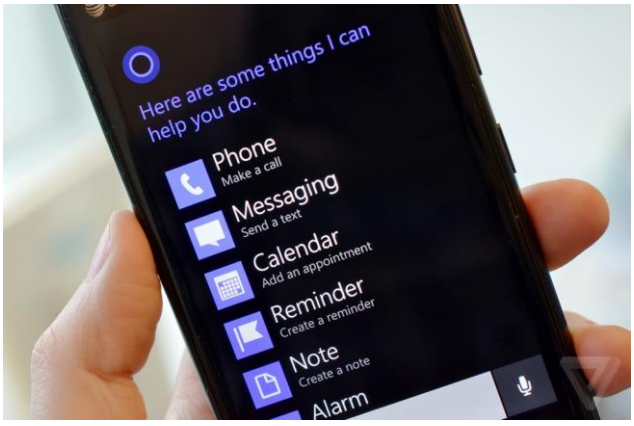

Figura 20 - Cortana em execução

- **Google Now:** O assistente pessoal do Google é o assistente pessoal mais utilizado no momento por ser o assistente pessoal do sistema operacional móvel mais utilizado – Android –, a grande vantagem deste assistente é a naturalidade dele para reconhecer o idioma português. Ele pode executar funções básicas do sistema android e abrir aplicativos instalados.

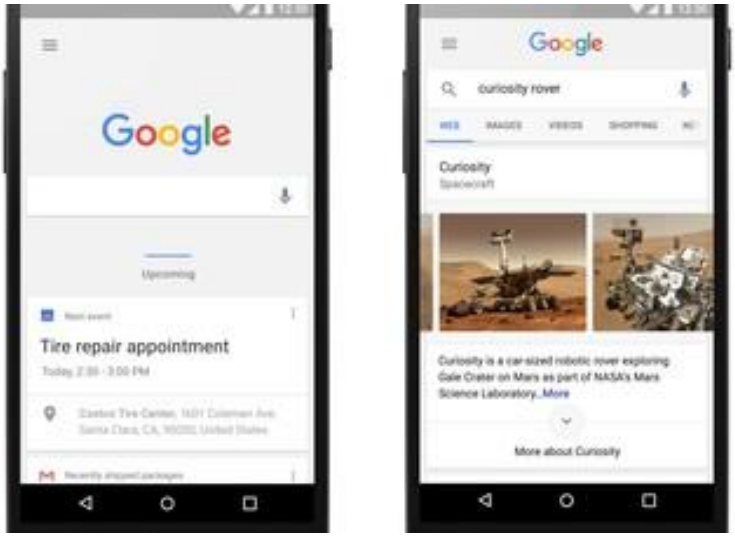

Figura 21 - Google Now em execução

- **Google Assistente:** O novo assistente do google, disponível para os SO Android 6 e posterior. Ele traz todas as facilidades do Google Now mas agora trabalhando com inteligência artificial.

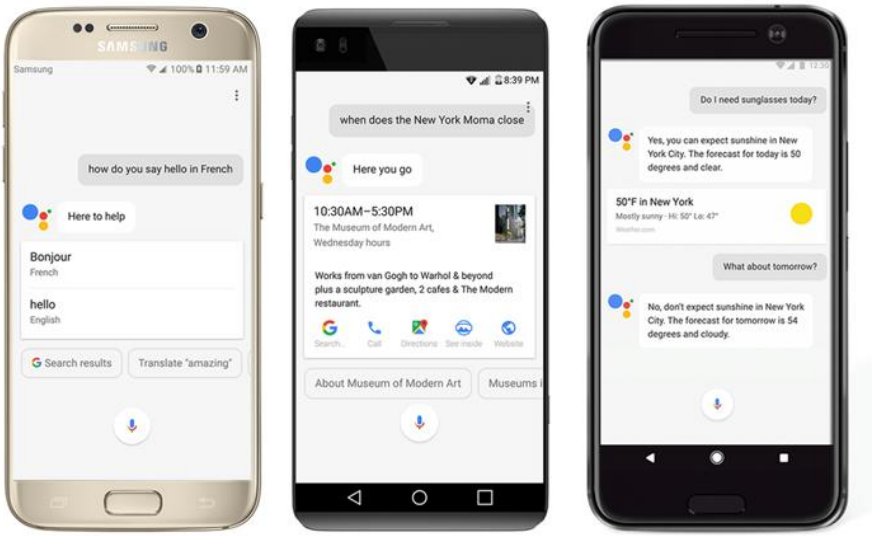

Figura 22 - Google Assistente em execução

- **Google Home:** É destinado para ficar parado em um cômodo da residência. Ele é capaz de interagir via comandos de voz com as pessoas, mantendo uma conversação e gravando preferencias por pessoa, além de também interagir com outros equipamentos na casa como Smart TV.

Figura 23 - Google Home

- **Alexa (Amazon):** A Alexa é a assistente pessoal virtual com mais “habilidades” a oferecer ao usuário. São mais de 15 mil funções de interação a outros aplicativos.

3.3.2 - Web Speech API

A Web Speech é uma API da linguagem javascript que permite implementar a função de reconhecimento de fala e a conversão de áudio para texto em um modo continuo. A grande vantagem desta API é sua simplicidade de integração e sua facilidade na utilização.

Para que a API entre em execução o navegador pedira a autorização para uso do microfone para captar dados do microfone. A autorização de utilizador pode incluir, por exemplo:

- O usuário pode selecionar um elemento gráfico contido na página para disparar a função de reconhecimento da fala.
- Aceitar a requisição do prompt feita pelo navegador, assim iniciando o reconhecimento da voz (SpeechRecognition.start).
- Autorização já concedida anteriormente que faz que o reconhecimento de voz ocorra sem nenhuma nova requisição do navegador.

O navegador sempre informara ao usuário que o recurso de microfone esta habilitado através de um ícone no topo da página como na figura 23.

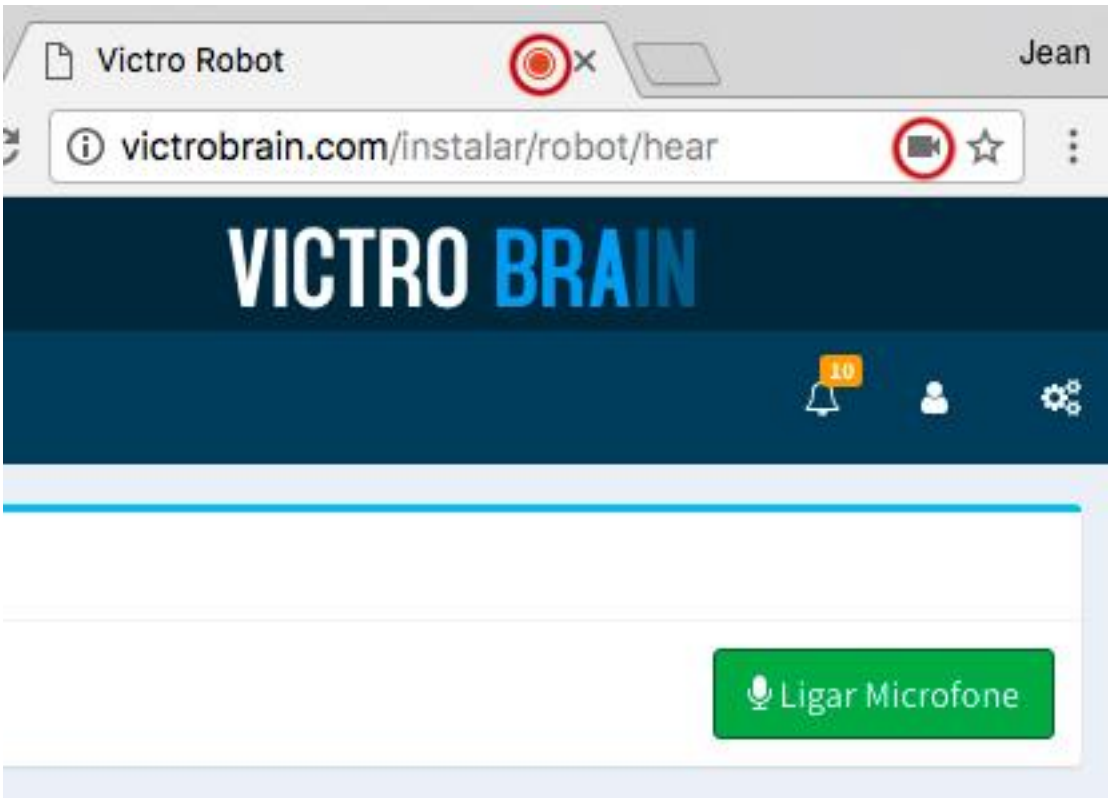

Figura 24 - Indicador de uso de microfone no Google Chrome/Chromium

A incompatibilidade de alguns navegadores traz algumas desvantagens para a API. A tabela 1 mostra que ainda assim a API funciona nos navegadores mais utilizados hoje em dia.

Navegador	**Suporte**
Chrome	Suportado (prefixo webkit) versão chrome 33 +

EDGE	Suportado
Firefox	Não suportado por bugs no navegador mas é possível através do media.webspeech.recognition.enable
Internet Explorer	Não suportado
Opera	Não suportado
Safari	Não suportado

O Web Speech possibilita as aplicações web lidar com os dados de áudios. Há dois componentes principais da API:

- Speech synthesis
- Speech recognition

O reconhecimento é feito através da interface SpeechRecognition, que possibilita a aplicação captar áudio a partir de uma entrada de áudio e converter em texto. Para que a interface seja iniciada é necessário criar um objeto da interface SpeechRecognition, que traz várias funções chamadas de eventos, são eles:

- **audiostart** – Evento chamado ao inicar o serviço de captura de áudio.
- **soundstart** – O evento soundstart é disparado quando qualquer som reconhecido pelo speech ou não, for detectada. O evento audiostart deve ser sempre ativado antes do evento soundstart.
- **speechstart**- O evento speechstart é disparado quando o som reconhecido pelo serviço de reconhecimento de voz é uma fala. O evento audiostart deve ser sempre ativado antes do evento speechstart.
- **speechend** - O evento de speechend é disparado quando o discurso reconhecido pelo serviço de reconhecimento de fala parou de ser detectado. O evento speechstart deve ser sempre inativado antes do evento speechend.
- **soundend** - O evento soundend é disparado quando qualquer discurso reconhecido pelo som ou não, deixou de ser detectado. O evento soundstart deve ser sempre inativado antes do evento soundend.
- **audioend** - O evento audioend é disparado quando o agente do usuário finalizou a captura de áudio para reconhecimento de fala. O evento audiostart deve sempre tersido despedido antes audioend.

- **result** - O evento de resultado da Web Speech API é disparado quando o serviço de reconhecimento de voz retorna um resultado - uma palavra ou frase foi reconhecida positivamente e isso foi comunicado de volta ao aplicativo. O evento audiostart deve ser sempre ativado o antes do evento de resultado.
- **NoMatch** - O evento de nomatch é disparado quando o serviço de reconhecimento de fala retorna um resultado final sem reconhecimento significativo. O evento audiostart deve ser sempre ativado antes do evento NoMatch.
- **error** - O evento de erro é disparado quando ocorreu um erro; As circunstâncias exatas variam, os eventos desse nome são usados a partir de uma variedade de APIs.
- **start** - O evento start do objeto Web Speech API SpeechRecognition é disparado quando o serviço de reconhecimento de fala começou a ouvir áudio recebido com a intenção de reconhecer gramáticas associadas ao SpeechRecognition atual.
- **end** - O evento end é disparado quando o serviço de reconhecimento de fala foi desconectado.

A interface SpeechGrammar representa um conjunto de palavras ou padrões de palavras que queremos que o reconhecimento de voz reconheça. Gramática é definida usando JSpeech Grammar Format (JSGF). Outros formatos também podem ser suportados no futuro.

SpeechRecognition Interfaces

- **SpeechRecognition** - A interface SpeechRecognition da Web Speech API é a interface do controlador para o serviço de reconhecimento; isso também lida com o SpeechRecognitionEvent enviado do serviço de reconhecimento.
- **SpeechRecognitionAlternative** - A interface SpeechRecognitionAlternative representa uma única palavra que foi reconhecida pelo serviço de reconhecimento de fala.
- **SpeechRecognitionError** - A interface SpeechRecognitionError representa mensagens de erro do serviço de reconhecimento.
- **SpeechRecognitionEvent** – A interface SpeechRecognitionEvent representa o objeto de evento para os eventos de result e de nomatch e contém todos os dados associados a um resultado de reconhecimento de fala provisório ou final.
- **SpeechGrammar** - A interface SpeechGrammar representa um conjunto de palavras ou padrões de palavras que queremos que o reconhecimento reconheça.
- **SpeechGrammarList**- A interface SpeechGrammarList representa uma lista de objetos SpeechGrammar contendo palavras ou padrões de palavras que queremos que o

reconhecimento de voz reconheça. A gramática é definida usando JSpeech Grammar Format (JSGF). Outros formatos também podem ser suportados no futuro.

- **SpeechRecognitionResult** - A interface SpeechRecognitionResult representa uma combinação de reconhecimento único, que pode conter vários objetos SpeechRecognitionAlternative.

- **SpeechRecognitionResultList** - A interface SpeechRecognitionResultList representa uma lista de objetos SpeechRecognitionResult ou uma única se os resultados estiverem sendo capturados no modo contínuo.

O retorno em texto da voz reconhecida é feito por JSON. JSON é um acrônimo para "JavaScript Object Notation", é um formato de padrão aberto que utiliza texto legível a humanos para transmitir objetos de dados consistindo de pares atributo-valor.

```
{
  ..
  results: {
    0: {
      0: {
        confidence: 0.895017397403717,
        transcript: "Hello, I am Victro!"
      },
      isFinal:true,
      length:1
    },
    length:1
  },
  ..
}
```

Figura 25 – Exemplo de retorno do reconhecimento de voz

No Victro Robot a chamada para reconhecimento de voz acontece no seu processo de inicialização. Assim que o Raspberry Pi é iniciado é feita uma requisição para iniciar o navegador Chromium, e logo após é chamado o framework VictroBrain que inicia uma página HTML5 chamando o script e assim começando o reconhecimento de voz. Todo este processo de inicialização do Raspberry Pi, navegador e framework demoram cerca de 60 segundos.

Segundo o site oficial (2017) o Chromium é um projeto de navegador de código aberto que visa construir uma maneira mais segura, rápida e estável para todos os usuários de Internet de experimentar a web. Ele é uma versão Open-Source do navegador Google Chrome.

3.4 Analise de Tons (IBM)

O serviço IBM Watson Tone Analyzer utiliza análises linguísticas para detectar tons emocionais nos textos enviados. O serviço pode analisar o tom nos níveis de documentos e frases. É possível usar o serviço para entender como as comunicações escritas são percebidas e melhora-las a fim de melhorar o tom da comunicação. As empresas podem usar o serviço para aprender o tom das comunicações de seus clientes e responder a cada cliente adequadamente, ou para entender e melhorar suas conversas de clientes em geral.

O funcionamento desta API é relativamente fácil, basta enviar um texto em JSON com o texto ou frase que queira saber as emoções, pode-se enviar ate 128KB de texto, que seria cerca de 1000 frases. O serviço retorna resultados em JSON que relatam o tom da frase de entrada.

```
Phrase: Hello, my name is VictroRobot and I love talk to you!
{
  "document_tone": {
    "tones": [
      {
        "score": 0.771154,
        "tone_id": "joy",
        "tone_name": "Joy"
      }
    ]
  }
}
```

Figura 26 - Exemplo de retorno JSON do Tone Analyzer

Na figura 25 pode-se observar o JSON que a API retorna para a aplicação. O JSON contém a emoção que a API detectou na frase, podendo ser:

- Emoção: raiva, medo, aproveitamento (felicidade), tristeza e desgosto (desativado no segundo semestre de 2017).
- Linguagem: analítica, confiante e tentativa.
- Social (Todos desativados no segundo semestre de 2017): abertura, agregabilidade, concienciosidade, extraversão e gama emocional.

Para integrar a API ao framework foi desenvolvido um robot (plugin) responsável por fazer a chamada do serviço via cURL do PHP. De acordo com a documentação oficial o PHP suporta libcurl, uma biblioteca criada por Daniel Stenberg, que permite a conectividade e a comunicação com diferentes tipos de servidores usando diferentes tipos de protocolos. libcurl atualmente suporta os protocolos https, ftp, gopher, telnet, dict, file e ldap. libcurl também

suporta certificados HTTPS, HTTP POST, HTTP PUT, upload via FTP, upload HTTP por formulário, proxies, cookies, e autenticação com usuário e senha.

```php
$data = json_encode(array('text' => $phrase));
$URL = 'https://gateway.watsonplatform.net/tone-analyzer/api/v3/tone?version=2016-05-19';

$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, $URL);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_TIMEOUT, 30); //timeout after 30 seconds
curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_ANY);
curl_setopt($ch, CURLOPT_USERPWD, "$username:$password");
curl_setopt($ch, CURLOPT_HTTPHEADER, array('Content-Type: application/json'));
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_POSTFIELDS, $data);
$result = curl_exec($ch);
curl_close($ch);
$newResult = json_decode($result);
```

Figura 27 - Exemplo de chamada cURL para Tone Analyser

A figura 26 mostra como é feita a chamada cURL para a API Tone Analyzer, primeiramente a frase escutada pelo Web Speech API é transformada em JSON e é informado a URL da API Tone Analyzer, após isso o cURL é aberto e enviado como parâmetro POST a frase assim como usuário e senha fornecidos pelo site da IBM Watson.
No Victro Robot a analise de tom é muito importante pois ela ajudara a controlar o humor do chatbot, tornando-o mais humano quando se trata de conversação, ou seja, ao falar algo para o chatbot ele enviará para o tone analyzer e com base no retorno fará a alteração de humor para a próxima resposta.

3.5 DialogFlow

DialogFlow é uma API para criar chatbots, usando *machine learning* para processar as mensagens e tornar a conversa entre o bot e o usuário mais agradável e mais natural. A API possui uma rica documentação e quem comanda este serviço é o Google.

A API irá trabalhar com o VictroRobot para encontrar a intenção daquilo que foi dito e encontrar a resposta na programada para aquela intenção.

3.5.1 – Como usar o DialogFlow

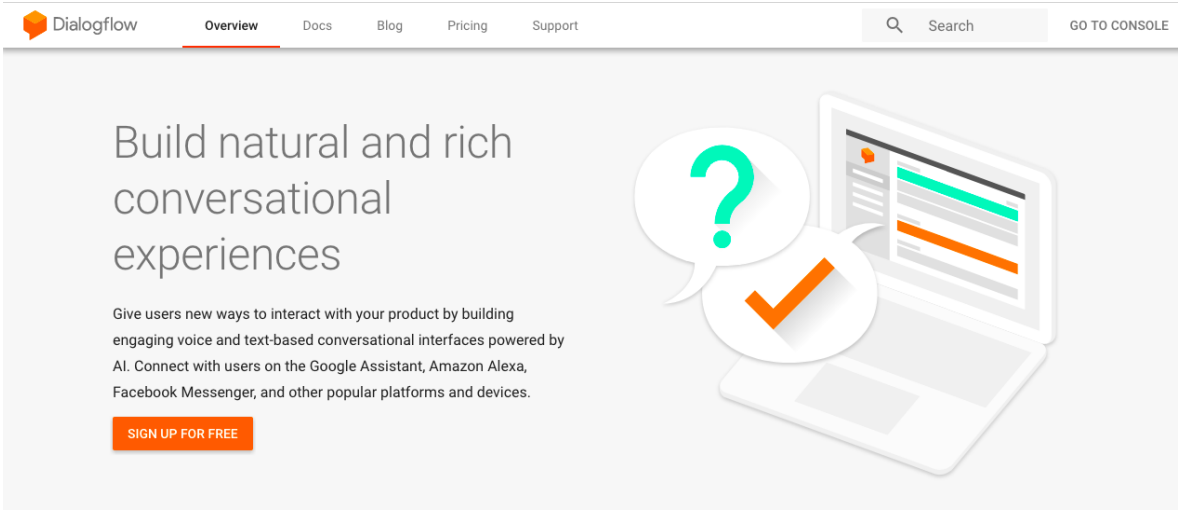

Figura 28 - Acessando dialogflow.com

Após acessar dialogflow.com, como ilustrado na figura 27, é necessário clicar em ‘go to console’ no menu superior, ao lado da caixa de busca.

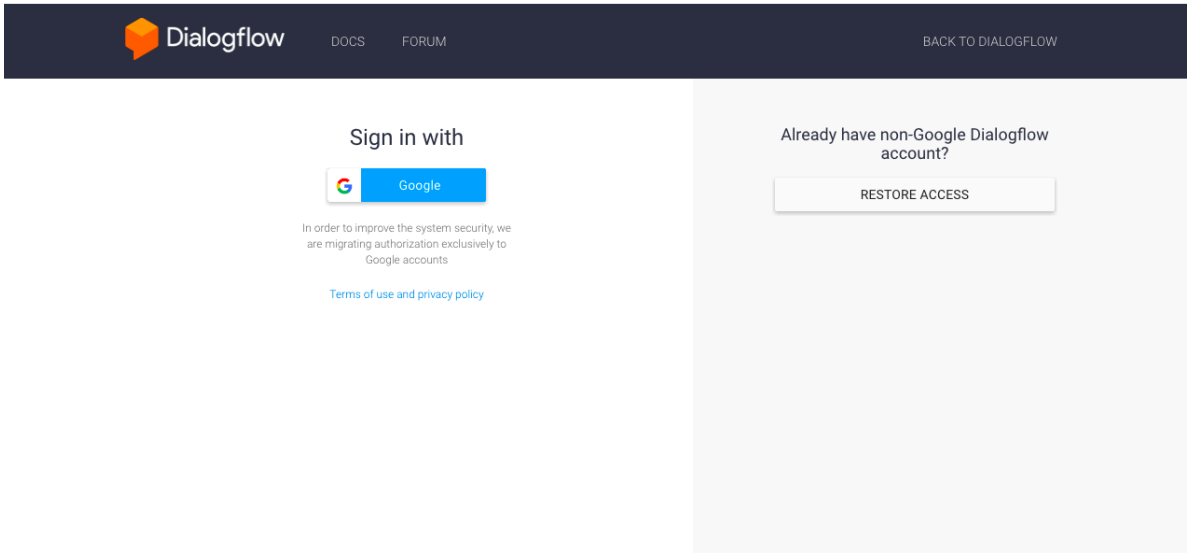

Figura 29 - Fazendo login no DialogFlow

Será aberto uma nova página para que o usuário faça o login, para isto é necessário que o usuário do dialogflow.com tenha uma conta google. Clicando em ‘Sign in with Google’ o usuário será redirecionado a pagina de login do Google e será perguntado que o usuário realmente quer usar sua conta google na plataforma DialogFlow, aceitando esta requisição o usuário será redirecionado a pagina principal, chamado console.

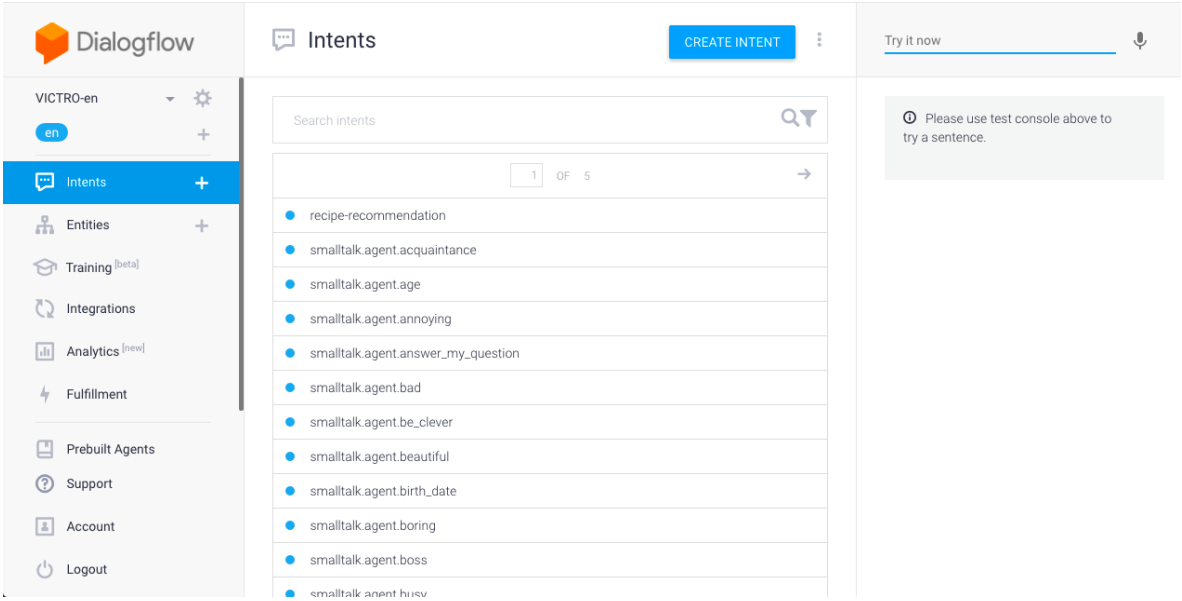

Figura 30 - Console DialogFlow

Pode-se observar na figura 29 que o console é dividido em 3 áreas, menu lateral esquerdo onde é possível navegar pelas paginas do console, área central onde o conteúdo é carregado e o menu lateral direito onde é possível realizar testes no chatbot criado.

A criação de um chatbot pelo DialogFlow é divido em:

- Agente: O chatbot em si, ele é tratado como um agente no DialogFlow
- Intenções: Permitem definir quais ações que o programa executará, dependendo do que um usuário diga. Uma amostra de intenção seria “Converter Moeda”. Em seguida, listar todas as possíveis frases que o usuário diria se quisesse converter moeda. Por exemplo, um usuário poderia dizer “Quanto é @sys.number:number @currency:fromCurrency em @currency:toCurrency?” Neste exemplo, foi utilizado duas entidades: @sys.number e @currency. Usando os dois pontos depois da entidade permite definir um alias para essa entidade. Este alias pode então ser usado no código para obter o valor da entidade.
- Entidades: é um objeto usado para extrair valores a partir das mensagens em linguagem natural. Qualquer dado que seja necessário interpretar deve ser especificado como entidade.

Assim como a API IBM Watson Tone Analyzer a API DialogFlow também faz a integração com PHP através do cURL, como mostra a figura 30.

```
$data = array("query" => $phrase, "lang" => "en", "sessionId" => "1234567890");
$data_string = json_encode($data);

$ch = curl_init('https://api.api.ai/v1/query?v=20150910');
curl_setopt($ch, CURLOPT_CUSTOMREQUEST, "POST");
curl_setopt($ch, CURLOPT_POSTFIELDS, $data_string);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_HTTPHEADER, array(
    'Content-Type: application/json',
//    'Content-Length: ' . strlen($data_string),
    'Content-type: application/json',
    'Authorization: Bearer xxxxxxx',
    'ocp-apim-subscription-key: XxxX-xxx')
);
```

Figura 31 - Integração cURL entre DialogFlow e VictroBrain

4 CRONOGRAMA

ATIVIDADES	**Jul.**	**Ago.**	**Set.**	**Out.**	**Nov.**
1 Levantamento de literatura	X	X			
2 Coleta de dados		X	X		
3 Tratamento e Análise dos dados			X		
4 Formatação do TCC			X		
5 Revisão do texto			X	X	
6 Desenvolvimento do Chatbot		X	X	X	

REFERÊNCIAS

BARBARINI, Elisa Signoreto et al. **Aplicações de um sistema de comando por voz e um software de controle na engenharia de reabilitação**. 2008. Tese de Doutorado. UNIVERSIDADE DE SÃO PAULO. Disponível em: <file:///Users/contato/Downloads/Barbarini_Elisa_Signoreto.pdf> Acesso em: 30 Out 2017.

Chromium – Site Oficial. Disponível em: < https://www.chromium.org/Home >. Acesso em 12 Out 2017.

FERREIRA, L. P.; UCHÔA, Joaquim Quinteiro. **Desenvolvimento de um chatbot para auxiliar o ensino de Espanhol como Língua Estrangeira**. 2008. Disponível em: <http://repositorio.ufla.br/bitstream/1/9629/1/ARTIGO_Desenvolvimento%20de%20um%20chatbot%20para%20auxiliar%20o%20ensino%20de%20espanhol%20como%20l%C3%ADngua%20estrangeira.pdf>. Acesso em: 26 Ago 2017.

GUPTA, S. et al. An E-Commerce Website based Chatbot. **International Journal of Computer Science and Information Technologies**, v. 6, n. 2, p. 1483-1485, 2015. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.734.8303&rep=rep1&type=pdf>. Acesso em: 28 Ago 2017.

MINETTO, E. L. **Frameworks para Desenvolvimento em PHP**. São Paulo: Novatec, 2007. Disponível em: <http://www.martinsfontespaulista.com.br/anexos/produtos/capitulos/243418.pdf >. Acesso em: 30 Ago 2017.

OLIVEIRA, R. et al. Recursos para desenvolvimento de aplicativos com suporte a reconhecimento de voz para desktop e sistemas embarcados. **12o Fórum Internacional de Software Livre**, 2011. Disponível em: <www.researchgate.net/profile/Nelson_Neto/publication/267709371_Recursos_para_Desenvolvimento_de_Aplicativos_com_Suporte_a_Reconhecimento_de_Voz_para_Desktop_e_Sistemas_Embarcados/links/5492c40a0cf209fc7e9f7ec2.pdf>. Acesso em: 22 Nov 2016.

RATO, J. P. C. **Conversação homem-máquina. Caracterização e avaliação do estado actual das soluções de speech recognition, speech synthesis e sistemas de conversação homem-máquina**. 2016. Tese de Doutorado.). Disponível em: <https://iconline.ipleiria.pt/bitstream/10400.8/2375/1/jo%C3%A3o%20Rato-Mestrado%20em%20Eng.Inform%C3%A1tica-Computa%C3%A7%C3%A3o%20M%C3%B3vel.pdf >. Acesso em: 29 Ago 2017.

SELLITTO, M. A. **Inteligência artificial: uma aplicação em uma indústria de processo contínuo**. Gestão & Produção, v. 9, n. 3, p. 363-376, 2002. Disponível em: <http://www.scielo.br/pdf/%0D/gp/v9n3/14574.pdf >

SIMIONI, M. C.; BETINI, R. C. **Monitoramento Da Frequência Cardíaca Via Método De Magnificação De Vídeo Euleriana**, 2014. Disponível em: <http://www.canal6.com.br/cbeb/2014/artigos/cbeb2014_submission_703.pdf>. Acesso em: 24 Nov 2016.

WEB SPEECH API – **Documentation**. Disponível em < https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API >. Acesso em 10 Out 2017.

Botucatu, ____ de _____ de 2017.

_____
Jean Victor Mendes dos Santos

De Acordo:

_____
Prof. Esp. Rogério Ferreira Sgoti

_____
Prof.
Coordenador do Curso de Análise e Desenvolvimento de Sistemas