**Equipe de Interação II**

**Fatequino: Soluções e interação mobile para usuários com Raspberry Pi**

Carlos Vinicius – Fatec Carapicuíba

Felipe Silva – Fatec Carapicuíba

Felipe Augusto – Fatec Carapicuíba

Gabriel Sales – Fatec Carapicuíba

Genilson – Fatec Carapicuíba

Guilherme Dourado – Fatec Carapicuíba

Matheus Oliveira – Fatec Carapicuíba

Vinicius Bueno – Fatec Carapicuíba

Prof(a). Me Mario Marques – Fatec Carapicuíba – *e-mail*:

MarioMarques@fatec.sp.gov.br

# INTRODUÇÃO

Este trabalho é referente a disciplina de Tópicos Especiais em Informática. A proposta inicial da disciplina consiste em desenvolver um robô chamado Fatequino que auxiliará os membros da Fatec Carapicuíba (professores, alunos, funcionários etc.) com diversas informações sobre a Fatec.

Com base nesse robô, foram divididos alguns temas para serem trabalhados, tais como, Visão, Web e Mecânica 1 e Mecânica 2, Interação 1 e Interação 2, e o nosso grupo será responsável por trabalhar sobre o tema de Interação 2.

A equipe de Interação 2 é a equipe responsável por pesquisar e desenvolver a solução de interação como:

* Comunicação do Raspberry com celulares via WIFI.
* Desenvolvimento de app para Android para comunicação.
* Integração com o software de visão computacional para reconhecimento prévio de alunos, professores e funcionários.

# Análise de Gap

Através dos materiais disponibilizados pelo grupo de controle e professor tais como, GitHub de aplicação e relatórios foi possível avaliar o que já havia sido construído para a parte de Interação 2.

Até o momento temos disponível para a parte de interação uma aplicação Mobile desenvolvida em linguagem de programação Dart utilizando o framework flutter que possui um cliente HTTP que permite a realização de uma chamada POST passando como parâmetro a mensagem que o usuário digitou na tela de conversa com o chatbot.

A aplicação ainda conta com uma tela para seleção do tipo de forma de comunicação com o chatbot respectivamente web e bluetooth.

A documentação desenvolvida pela equipe anterior conta com um embasamento técnico e teórico a respeito da construção executada, possuindo informações de implementação, pontos que agregam valor ou não, além de análise de viabilidade de algumas implementações referente ao aplicativo Mobile.

# Possíveis melhorias a serem implementadas

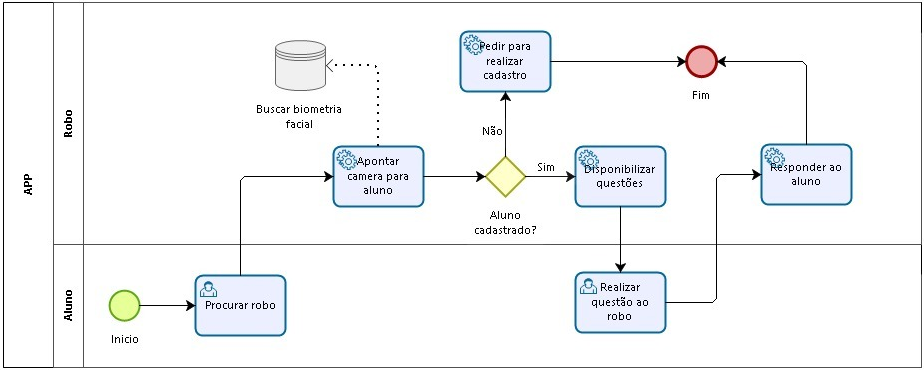
Através da avaliação realizada dos materiais fornecidos do que foi construído para o tema de Interação 2 foi possível verificar que a aplicação mobile ainda possui uma estrutura que facilita o desenvolvimento, é necessário então fazer modificações na estrutura de chamada da API do chatbot da aplicação para que tome sua forma final a ser implantada no Fatequino.

No relatório evidenciou-se a falta de viabilidade e dificuldade que é imposta ao trabalhar com conexão através de bluetooth, porém a aplicação ainda possui o botão com ícone de bluetooth, sendo necessário realizar alterações na tela de seleção do tipo de comunicação.

Não foi encontrado estudo documentado referente a implementação do chatbot no robô Fatequino quanto a formas, viabilidade, capacidade de processamento e memória, disponibilização de hotspot WI-FI para executar essa implementação, sendo então proposto o levantamento destes pontos.

Proposição de uma solução para integração com o software de visão computacional que pode ser visto pelo modelo BPMN da figura 1 abaixo contendo a tarefa que demonstram as etapas a serem executadas integrando os diversos temas do projeto:

Figura 1: processo de integração com software de visão para reconhecimento e utilização do chatbot.



Fonte: diagrama elaborado pelos próprios autores.

# melhorias Implantadas

Através da análise de gap levantada foram executadas melhorias evolutivas sobre o conteúdo consolidado para a parte de Interação 2, tais como, pontos de código (*refactoring* e funcionamento), Melhorias visuais da aplicação, Criação de Manual Explicativo de como rodar a aplicação, Criação do ReadMe para Interação 2 e Aplicativo Fatequino e Estudo sobre como disponibilizar WI-FI através do Raspberry.

A equipe realizou modificações sobre a classe de serviço da aplicação, que através de testes realizados não estava conseguindo realizar requisições http, utilizando a classe HttpClient, um cliente que permite receber conteúdos como páginas web, arquivos e dados de servidores pelo protocolo HTTP. Através de um conjunto de métodos que permite enviar requisições e receber respostas dessas requisições. Além de remoções de trechos de códigos para otimizar a aplicação e facilitar a manutenção.

Como forma de melhorar a usabilidade da aplicação foram removidos componentes da aplicação como a inserção dos parâmetros do servidor (inputs para URL e porta do servidor, aba para inserção desses parâmetros e ícone para bluetooth).

Foi desenvolvido um manual passo a passo para configuração do ambiente em Android Studio que permite rodar a aplicação ou mesmo desenvolver novas aplicações em Flutter, o manual foi disponibilizado no Drive do projeto do Fatequino.

Conforme proposto foi desenvolvido README da equipe de Interação 2 utilizando o pacote do NPM “readme-md-generator”, que facilita e otimiza a geração de READMEs, possuindo descrições e acesso a documentação da equipe.

Foi realizado um estudo para inclusão da funcionalidade de disponibilização de WI-FI através do Raspberry que será apresentado a seguir:

## RASPBERRY

Através de levantamento realizado obtemos o modelo do Raspberry utilizado no Fatequino sendo um Raspberry Pi 4+ que se trata de um hardware que permite as funções de um desktop convencional como ver vídeos, editar documentos, rodar um sistema operacional, realizar processamentos, e executar funções de rede wireless, ethernet, bluetooth e USB-C.

O Raspberry possui como formas de entrada de internet opções como wireless, ethernet, bluetooth e USB-C e permite o roteamento de sinal WI-FI com um canal, ou seja, ou o dispositivo pode ser configurado para ou enviar ou receber dados.

Tendo em vista o ponto exposto temos algumas possibilidades de disponibilização de WI-FI, na qual para obtermos acesso à internet temos que utilizar ou a entrada GIGABIT Ethernet, ou através dos conectores USB como formas mais práticas e acessíveis.

## Acess Point

Após a obtenção de Internet no Raspberry, é necessário convertê-lo em um ponto de acesso WI-FI utilizando o Raspbian que possui o software daemon **Hostapd** que permite que uma placa de interface de rede (Raspberry) atue como ponto de acesso e servidor de autenticação e o **Dnsmasq** paragerenciar os serviços de rede, pode ser visto como realizar a configuração através do site do próprio raspberry.

### Hostapd

Hostapd foi concebido para prover todas as funcionalidades do protocolo WPA independente do hardware, driver ou sistema operacional, construído em linguagem C, possui gerenciamento de pontos de acesso IEEE 802.1X, WPA, WPA2, um servidor EAP e funcionalidades de servidor de autenticação RADIUS, podendo ser utilizado através de diversas formas de configuração conforme descrito em *hostapd Reference Manual*.

Dentre as possibilidades deste pacote de arquivos podemos utilizá-lo para configurar a plataforma como um ponto de acesso sem fio onde ele permite realizar a autenticação e associação da rede com a configuração de arquivos.

### Dnsmasq

O dnsmasq é um servidor leve e de tamanho reduzido com a intenção de prover serviços DHCP e DNS para a rede local (LAN), pode aceitar consultas DNS e fornece respostas a partir de um pequeno armazenamento em cache local otimizando a velocidade de pesquisa em endereços já visitados antes ou então encaminha para um outro servidor DNS. Consegue suportar atribuições de endereços estáticos de IP e várias redes de conexões conectando-se a smartphones e pontos de acessos portáteis.

## Limitação de Acesso

Com a rede configurada para limitar o acesso a alternativa encontrada para restringir o acesso é o servidor de proxy **Squid** que permite através de configurações restringir o acesso a sites.

### Squid

O squid é um servidor proxy de cache voltado para web, suporta protocolos http, https, entre outros. Através do armazenamento em cache permite a otimização de respostas de solicitações assim como o Dnsmasq que são solicitadas com uma certa frequência.

O Squid possui controles de acesso e funciona na maioria das plataformas, incluindo Windows e Linux, assim como a otimização em cache disponibilizada através do Squid é possível definir restrições de acesso atuando como um firewall, assim disponibilizando um controle de conteúdo.

Na configuração de proxy o Squid atua como um intermediário entre o solicitante (host que faz a requisição) e o solicitado (servidor web que foi requisitado) de forma que ao receber a requisição do solicitante é feita uma busca no cache que se houver informações da requisição é retornada ao solicitante, caso contrário se não houver restrição é enviada ao servidor solicitado.

O proxy não consegue proteger todo o acesso à internet, geralmente ele atua no controle de acesso para navegadores, assim acessos de formas distintas necessitam a adoção de outros métodos de controle e proteção, como o caso de emails.

# soluções de armazenamento de dados em NoSQL

## Banco de dados

Um elemento presente na maioria dos sistemas é o banco de dados que tem a função de armazenar dados a respeito do sistema e do negócio (Machado e Abreu, 1996) o banco de dados tem como fundamento básico manter e definir os dados independente da aplicação que interage com eles.

Assim podemos definir banco de dados segundo Machado e Abreu (1996, p.32), como “uma coleção de fatos registrados que refletem o estado de certos aspectos de interesse do mundo real. A todo momento o conteúdo do banco de dados representa uma visão instantânea do estado do mundo real.” (Machado e Abreu).

### Bancos de dados não relacionais

Banco de dados não relacionais são tipos de bancos que não possuem tabelas de relacionamento, cardinalidade, possuindo coleções com atributos dinâmicos, que podem ser do tipo chave-valor, documentos JSON, entre outros.

O MongoDB é um banco de dados não relacional (NOSQL) que armazena coleções de documentos em formato JSON, onde cada documento pode ter atributos dinâmicos como mostrado na figura 2 na qual dois documentos da mesma coleção, o primeiro documento possui o atributo “*cuisine”* e o outro documento não o possui, ou seja, podem haver documentos de uma mesma coleção que possuem ou não os mesmos atributos, sendo Open-Source e com capacidade para armazenar arquivos que excedem 16 MB, utiliza uma interface interativa em javascript chamada mongo Shell permitindo executar atividades de CRUD, realização de queries e operações administrativas, além de possuir vários clientes com interface de usuário guiadas (GUI) como o Robo 3T e o MongoDB Compass.

Figura 2: documentos de uma coleção com atributos dinâmicos.



Fonte: Registro de coleções obtidas no MongoDB Atlas pelos Próprios autores.

Como soluções disponíveis para o MongoDB tem-se o MongoDB Atlas como um serviço em nuvem que dispõe de gerenciamento pela Amazon, Azure ou Google Cloud Platform, conta com disponibilidade a todo momento, possui gráficos de análise de dados para visualização de padrões e tendências, monitoramento, escalabilidade, provê uma API com estilo arquitetural RESTFul, disponibilizando quatro planos incluindo um plano gratuito.

A solução On-Premises permite ser dono da infraestrutura sobre o qual o servidor do MongoDB roda, com as opções Enterprise e Community que disponibilizam as mesmas opções da versão disponível em cloud, no entanto a parte de escalabilidade e disponibilidade fica por conta da infraestrutura do servidor.

Ainda como uma opção é possível criar um servidor Mongodb utilizando Docker através de imagens disponibilizadas pela comunidade ou imagens oficiais disponíveis no Docker hub, permitindo com que o banco seja implantado em diversas plataformas como Windows, Linux, MAC IOS e suas versões obviamente desde que tenha imagens compatíveis com sua arquitetura.

## Docker

O Docker é uma plataforma voltada para o desenvolvimento, execução e compartilhamento de aplicações usando contêineres, simplificando o processo de virtualização além de facilitar a criação e gestão de ambientes isolados, de maneira que é possível desenvolver uma aplicação e empacotá-la levando dentro do container as dependências, variáveis e bibliotecas necessárias para rodar a aplicação em qualquer plataforma, além de permitir a execução de vários contêineres simultaneamente.

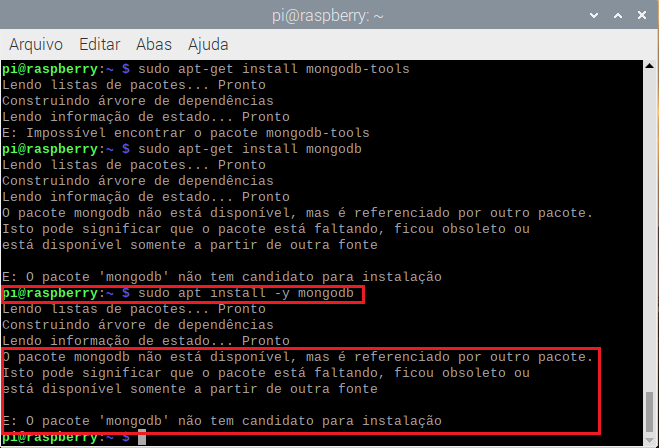
Contêineres são unidades que encapsulam software com as dependências permitindo a execução do software de maneira mais rápida e confiável, os contêineres são formados a partir de imagens executadas pelo Docker Engine. O Docker disponibiliza no Docker Hub uma grande quantidade de imagens oficiais e imagens criadas pela comunidade conforme a necessidade demandada.

## MongoDB com Raspberry

Existem algumas alternativas para criar instâncias de servidor MongoDB para rodarem sobre o Raspberry, tais como a utilização da instalação do MongoDB diretamente sobre o Raspberry, utilizando o sistema operacional oficial do Raspberry, o Raspbian, fazendo a instalação através do gerenciador de pacotes.

No entanto para as versões mais atuais disponíveis para o Raspberry PI 4 devido a arquitetura ARM de 32 bits que possui muitas limitações e encontra-se obsoleta e desatualizada conforme foi encontrado em fóruns de discussões sobre o assunto não foi obtido sucesso na tentativa de criação do servidor visto na figura 3, onde ao tentar realizar o download dos pacotes do MongoDB não foi encontrado nenhuma opção disponível.

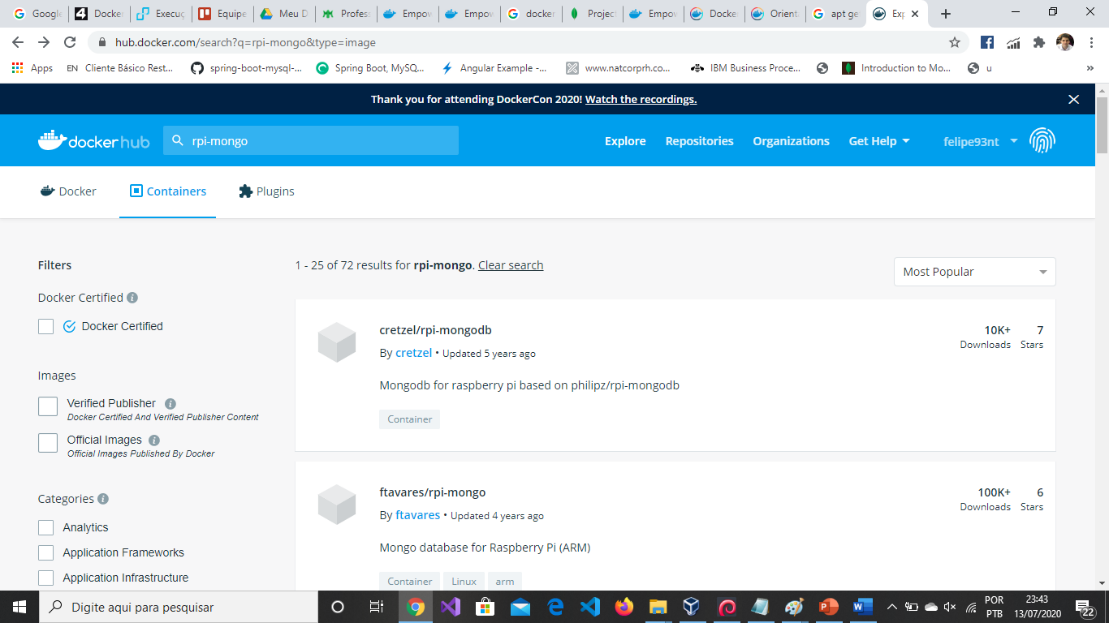
Figura 3: download do MongoDB para raspbian buster versão 10.



Fonte: Próprios autores.

Uma alternativa seria a instalação através de uma imagem do MongoDB para criação de um container Docker, após pesquisas realizadas não foram encontradas imagens oficiais disponibilizadas para Raspberry então a solução é utilizar uma versão criada pela comunidade ou criar a própria imagem, conforme a figura 4, podemos ver as opções disponibilizadas pela comunidade.

Figura 4: imagens MongoDB para Docker.

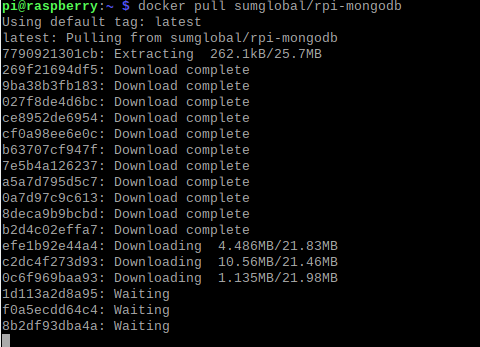


Fonte: Página web de imagens Docker disponível em <<https://hub.docker.com/>> Acesso em jul. 2020.

De acordo com os conceitos abordados sobre Docker a utilização de contêineres viabiliza a criação de um servidor MongoDB abstraindo-se os problemas com a infraestrutura encontrada no Raspberry.

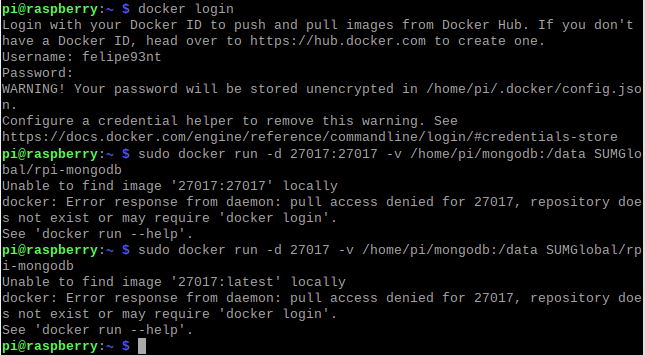
Foram feitas tentativas em duas imagens que apresentaram maior quantidade de utilização e atualizações recentes a primeira utilizando a imagem “sumglobal/rpi-mongodb” porém não foi possível criar o container com a imagem.

Figura 5: primeira imagem fazendo download.



Fonte: Próprios autores.

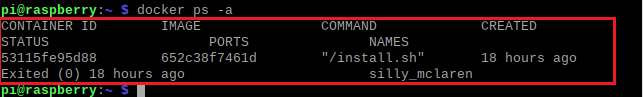
Figura 6: falha da primeira tentativa.



Fonte: Próprios autores.

A segunda imagem “timotto/rpi-mongodb” permitiu gerar o container, porém não foi possível realizar a conexão e configurar o servidor.

Figura 7: Container MongoDB rodando



Fonte: Desenvolvido pelos próprios autores.

## Alternativas

Como alternativa existe a possibilidade de utilizar a versão em nuvem do MongoDB fazendo uso da API disponibilizada para se ter acesso aos dados disponíveis no MongoDB, apresentando-se como a melhor opção pelos motivos de abstrair-se a preocupação com a administração da infraestrutura e da segurança, ter disponibilidade de acesso e possibilidade de escalar de forma automática caso exista um elevado número de solicitações ao banco.

# Considerações finais

Através dos resultados e pesquisas realizados pode-se propor e executar melhorias de forma evolutiva sobre todo o material desenvolvido, referente a parte de interação com usuários através de melhorias implementadas na aplicação mobile, e estudos propostos por equipes anteriores. Foi criado manuais explicativos sobre detalhes técnicos para servir como base e apoio para equipes futuras efetuarem as próximas evoluções, além de estudos de soluções para as próximas tarefas.

# Trabalhos futuros

Como sugestões de trabalhos futuros as próximas equipes poderão realizar estudos sobre como executar a integração com o software de visão computacional para reconhecimento prévio de alunos, professores e funcionários, criação de um cliente que possibilite a integração com o software de visão computacional no Raspberry e implantar os passos para conversão do Raspberry Pi em um Ponto de Acesso Wireless caso haja disponibilidade do hardware em mãos (Raspberry Pi).

**REFERÊNCIAS**  
Marques, Mario. **Tópicos Especiais em Informática.** Disponível em: https://sites.google.com/site/professormariomarques/fatequino/projeto/201902. Acesso em: 17 de maio de 2020.

Silva, Adenilson GitLab **fatequinoApp.** Disponível em: https://gitlab.com/adenilson.elias2/fatequinoapp/-/commits/master.

Acesso em: 12 de maio de 2020.

SILVA, Adenilson E.; SILVA, Gabriel C. M.; OLIVEIRA, Luiz R. A.; SOUZA, Tacio S. **FATEQUINO. Equipe de Interação**, São Paulo, 2019.

Acesso em 20 de maio de 2020.

Dart:io. **HttpClient class**. Disponível em:

<https://api.dart.dev/stable/2.8.4/dart-io/HttpClient-class.html>.

Acesso em: 6 de junho de 2020.

Andrews, Todd. Linux Wireless. **About hostapd**. Disponível em:

<https://wireless.wiki.kernel.org/en/users/documentation/hostapd#about_hostapd> Acesso em 05 de julho de 2020

Doxygen. **hostapd Reference Manual**. Disponível em: <https://hostap.epitest.fi/hostapd/hostapd-devel.pdf>

Acesso em 06 de julho de 2020

Kelley, Simon. **Dnsmasq**. Disponível em:

<http://www.thekelleys.org.uk/dnsmasq/doc.html>

Acesso em 06 de julho de 2020

Floriano, William Dresch. **GERENCIAMENTO DO PROXY SQUID ATRAVÉS DE UMA FERRAMENTA WEB COM BASE NA CRIAÇÃO DE PERFIS DE CONTROLE. Disponível em:**

<http://www.redes.ufsm.br/docs/tccs/Willian-Floriano.pdf>

Acesso em 23 de junho de 2020.

MACHADO, Felipe Nery Rodrigues; ABREU, Maurício Pereira de. **Projeto de Banco de Dados:** uma visão prática. 8. ed. - São Paulo: Érica, 1996.

MongoDB: **Get started with MongoDB.** Disponível em:

<https://docs.mongodb.com/>

Acesso em 12 de julho de 2020

Ondrus, David. **Docker on Raspberry Pi 4 — will it work?** Disponível em:

<https://touch4it.com/blog/docker-on-raspberry-pi-4-will-it-work>

Acesso em 13 de julho de 2020.

**“O conteúdo expresso no trabalho é de inteira responsabilidade do(s) autor(es).”**