API-Bots
TESTE- Everton Menezes

Fase: Especificação Técnica da API

Especificação Técnica

Funcionalidade: **Especificação**

Versão 1.0

HISTÓRICO DE ALTERAÇÕES

Versão	Descrição	Autor	Data
1.0	Criação do Documento	Everton Menezes	01/03/2018

LISTA DE FIGURAS

SUMÁRIO

1. OBJETIVO DOCUMENTO

Este documento tem como servir para especificações Técnicas de desenvolvimento, build & deployment, escalabilidade, tolerância a falhas e testes da API de Bots.

2. REFERÊNCIAS

3. ESCOPO DO DOCUMENTO

Especificação Técnica para API Bots, build & deployment, escalabilidade, tolerância a falhas e testes da API de Bots.

1. Para iniciar o projeto deve ser instalado os pacotes de dependências através do Node utilizando o comando:
    a. "**Npm i**"
2. Após a instalação deve ser realizada a inicialização do serviço de API e Banco de dados:
    a. No caso se você já estiver instalado e configurado os caminhos de PATH o MongoDB, inicie o serviço através do seguinte comando:
        i. "Mongod"
    b. Após iniciar o serviço do MongoDB, inicie o serviço da API atráves do seguinte comando:
        i. **"npm run production"** : este comando irá iniciar o serviço através do PM2 gerenciador de processos para produção, Load balancer, logs, startup script, micro service management.
3. O serviço irá iniciar nas seguintes portas:
    a. Bot Api: http://Localhost:3000
    b. Message Api: http://Localhost:4000

4. REQUISITOS

4.1 BOT API

4.1.1 *DESCRIÇÃO*

Este documento tem o objetivo de descrever algumas especificações técnicas utilizadas para o desenvolvimento da BotsAPI.

4.1.2 *TECNOLOGIAS UTILIZADAS*

Para desenvolvimento da Webpart foram utilizados os seguintes frameworks:

NodeJS – um interpretador de código JavaScript com o código aberto, focado em migrar o Javascript do lado do cliente para servidores.

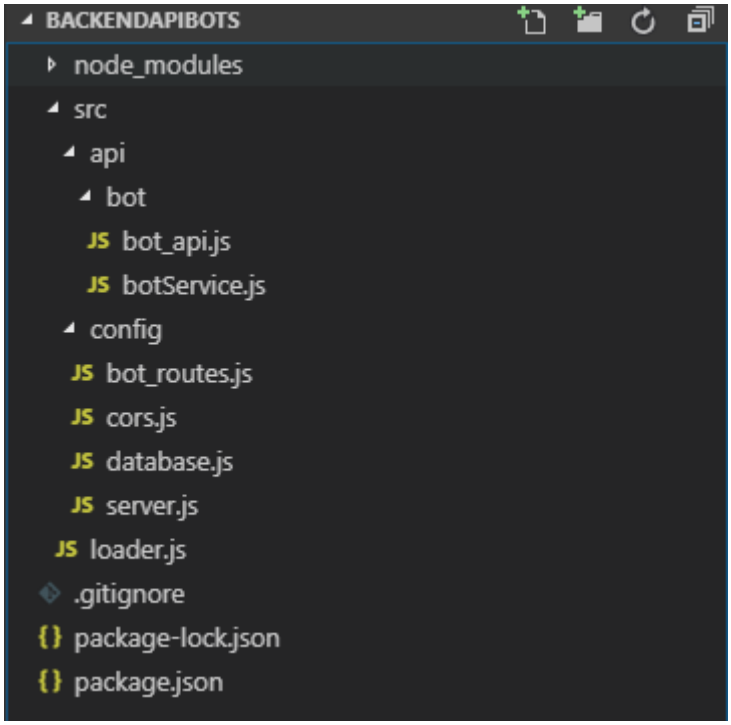
MongoDB – software de banco de dados orientado a documentos livre, de código aberto e multiplataforma, escrito na linguagem C++. Classificado como um programa de banco de dados NoSQL, o MongoDB usa documentos semelhantes a JSON com esquemas.

Express – é uma estrutura de aplicativo da web para o Node.js, lançada como software livre e de código aberto sob a licença MIT. Ele é projetado para construir aplicativos da Web e APIs.

Mongoose – é uma biblioteca do Nodejs que proporciona uma solução baseada em esquemas para modelar os dados da sua aplicação. Ele possui sistema de conversão de tipos, validação, criação de consultas e hooks para lógica de negócios.

Node-Restful – É uma biblioteca para fornecer rapidamente uma API REST com express.

Estrutura do Desenvolvimento:



1. **Realizei a criação de uma estrutura segregando as partes**
    a. **Loader** : Arquivo que carrega os principais arquivos de configuração da minha API.
    b. **Config** : Na pasta config encontram se todos os meus arquivos utilizados para configuração da api, tais como
        i. **Server** : Arquivo que carrega as configurações ao express e servidor, locação de porta e demais.
        ii. **Database**: Arquivo de configuração de conexão a base de dados mongodb utilizando mongoose.
        iii. **Cors** : Arquivo de configuração utilizado habilitar e fornecer a liberação de diferentes requisições HTTP a minha API
        iv. **Routes**: Arquivo de configuração do express, fornecendo as configurações de rotas da API.
    c. **Api**: na pasta api encontra se o arquivo utilizado para criação do schema e serviços da API
        i. **Bot_api** : configurações de Schema da Api, Mongoose e node-resful.
        ii. **botService** : configurações dos serviços de requisição utilizados pela API
2. **Dívidas Técnicas de Implementação**
    a. **Test Automation**
    b. **CI/CD**

4.2 MENSSAGE API

4.2.1 *DESCRIÇÃO*

Este documento tem o objetivo de descrever algumas especificações técnicas utilizadas para o desenvolvimento da BotsAPI.

4.2.2 *TECNOLOGIAS UTILIZADAS*

Para desenvolvimento da Webpart foram utilizados os seguintes frameworks:

NodeJS – um interpretador de código JavaScript com o código aberto, focado em migrar o Javascript do lado do cliente para servidores.

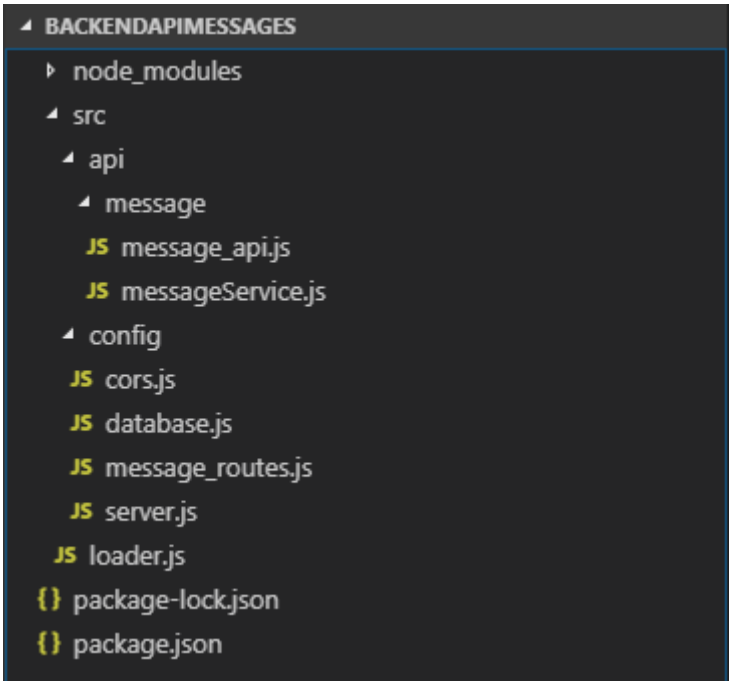
MongoDB – software de banco de dados orientado a documentos livre, de código aberto e multiplataforma, escrito na linguagem C++. Classificado como um programa de banco de dados NoSQL, o MongoDB usa documentos semelhantes a JSON com esquemas.

Express – é uma estrutura de aplicativo da web para o Node.js, lançada como software livre e de código aberto sob a licença MIT. Ele é projetado para construir aplicativos da Web e APIs.

Mongoose – é uma biblioteca do Nodejs que proporciona uma solução baseada em esquemas para modelar os dados da sua aplicação. Ele possui sistema de conversão de tipos, validação, criação de consultas e hooks para lógica de negócios.

Node-Restful – É uma biblioteca para fornecer rapidamente uma API REST com express.

Estrutura do Desenvolvimento:



1. Realizei a criação de uma estrutura segregando as partes
    a. **Loader** : Arquivo que carrega os principais arquivos de configuração da minha API.
    b. **Config** : Na pasta config encontram se todos os meus arquivos utilizados para configuração da api, tais como
        i. **Server** : Arquivo que carrega as configurações ao express e servidor, locação de porta e demais
        ii. **Database** : Arquivo de configuração de conexão a base de dados mongodb utilizando mongoose.
        iii. **Cors** : Arquivo de configuração utilizado habilitar e fornecer a liberação de diferentes requisições HTTP a minha API
        iv. **Routes** : Arquivo de configuração do express, fornecendo as configurações de rotas da API.
    c. **Api**: na pasta api encontra se o arquivo utilizado para criação do schema e serviços da API
        i. **Message_api** : configurações de Schema da Api, Mongoose e node-resful.
        ii. **MessageService** : configurações dos serviços de requisição utilizados pela API

2. **Dívidas Técnicas de Implementação**
    a. **Test Automation**
    b. **CI/CD**

4.3 SUGESTÕES DE IMPLEMENTAÇÃO

- **Docker** para realizar a escalação do serviço, assim dividindo a api em containers para implementação, facilitando a escalabidade da aplicação
- **Kubernetes**: para realizar orquestração de contêiners auto scaling e gerenciamento
- **Kafka**, para publicação de tópicos colocando inteligência para por exemplo acionar um determinado

serviço solicitado pelo cliente, facilitando a interação ou até mesmo para registrar de forma inteligente em uma base de dados as informações que estão sendo mais solicitadas. Ele ficaria ouvindo um tópico específico e acionaria um determinado serviço ao usuário.

- **NLP :** Para processar as interações realizadas através da plataforma de mensagens
- **MACHINE LEARNING :** Quanto mais mensagens o bot recebe ele melhora o atendimento, através de um aprendizado supervisionado.
- **INFORMATION SOURCES**
    - **DATA LAKE :** para armazenar os dados estruturados e não estruturados e realizar análises futuras.
    - **API para Parceiros :** poder fornecer a API para parceiros assim, gerando mais valor ao negócio podendo se tornar um serviço para o mercado.
    - **BIG DATA:** Repositório de Questões e base de conhecimento.