**Documento de Arquitetura de Software**

ChatBot – Código de Defesa do Consumidor

Data de Criação: 23/10/2017

Versão: 01.00

**Responsável:**

Diego Novaes – Arquiteto de Software

novaesdiego@hotmail.com

**HISTÓRICO DE REVISÕES**

|  |  |  |
| --- | --- | --- |
| **DATA** | **RESPONSÁVEL** | **MUDANÇA** |
| 23/10/17 | Diego Novaes | Criação do documento |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## Índice Analítico

* 1. Introdução

1.1 Finalidade

1.2 Escopo

1.3 Definições, Acrônimos e Abreviações

1.4 Referências

1.5 Visão Geral

* 2. Representação da Arquitetura

3. Metas e Restrições de Arquitetura

4. Visão de Casos de Uso

5. Visão Lógica

5.1 Visão Geral

6. Visão de Processos

7. Visão de Implantação

8. Visão de Implementação

8.1 Visão Geral

8.2 Camadas

8.2.1 View

8.2.2 Model

8.2.3 Controller

8.2.4 Modules

8.2.5 Public

9. Tamanho e Desempenho

10. Qualidade

## Documento de Arquitetura de Software

## 1. Introdução

Este documento tem como objetivo mostrar como deve ser estruturado o sistema em questão de modo que venha atender as restrições e funcionalidades que o software necessita. A solução se define em parte neste documento de arquitetura de software.

### 1.1 Finalidade

Este documento apresenta uma visão geral abrangente da arquitetura do sistema e utiliza uma série de visões arquiteturais diferentes para ilustrar os diversos aspectos do sistema. Sua intenção é capturar e transmitir as decisões significativas do ponto de vista da arquitetura que foram tomadas em relação ao sistema.

**1.2 Escopo**

Este Documento de Arquitetura de Software se aplica à aplicação web Chatbot - CDC.

### 1.3 Definições, Acrônimos e Abreviações

Elicitação: Técnica de obtenção de dados junto aos usuários detentores das informações.

### 1.4 Referências

Documento de Especificação de Objetivos e Requisitos (EOR) : <https://github.com/terciodejesus/chatbot-cdc/blob/master/documentos%20-%20MATB14/requirements/chatbot-req.doc>

Após a definição da finalidade e escopo deste documento, abaixo vamos estar descrevendo a arquitetura escolhida que melhor soluciona os problemas e necessidade encontrados neste projeto. Assim vamos estar apresentando em 5 visões sendo elas casos de uso, logica, processo, implantação e implementação.

## 2. Representação da Arquitetura

Este documento apresenta a arquitetura como uma série de visões: visão de casos de uso, visão lógica, visão de processos, visão de implantação e visão de implementação. Essas visões são apresentadas como modelos e utilizam a Linguagem Unificada de Modelagem (UML).

## 3. Metas e Restrições de Arquitetura

Existem algumas restrições de requisito e de sistema principais que têm uma relação significativa com a arquitetura. São elas:

|  |  |
| --- | --- |
| **Referencia** | **Descrição** |
| REQ09 | Ao se conectar com as redes sociais e for ser feita alguma publicação, sempre pedir permissão. |
| REQ07 | Verificar se a queixa é suficiente diferente e suficiente semelhante a algum caso para realizar votação. O registro deste caso só deve acontecer uma vez caso ocorra. |
| REQ06 | Sistema deve ter registrar que o voto a este caso foi realizado uma única vez pelo usuário. |

## 4. Visões de Casos de Uso

|  |  |
| --- | --- |
| **Caso de uso** | **Descrição** |
| UC01 | Neste caso de uso, o usuário inicializa o sistema Chatbot-cdc através de uma abertura de queixa. |
| UC02 | Neste caso de uso, o sistema e o usuário trocam informações com objetivo de aprimorar o entendimento da queixa do usuário e buscar uma artigo do codigo do consumidor que mais de adequa com o caso do usuário. |
| UC03 | Neste caso de uso, o sistema deve ser capaz de processar a queixa do usuário utilizando técnicas de NLP e api de sinônimos (wordnet ou altervista) para extrair da melhor forma informação da queixa do usuário. |
| UC04 | Neste caso de uso, o sistema irá gerar um relatório em PDF do caso analisado e artigo resultante em uma nova aba do navegador. Somente usuários registrado nos sistema terão acesso caso o requisito REQ09 seja implementado. Caso contrário, todos poderão gerar relatório. |

## 5. Visão Lógica

Descreve a estrutura do sistema, sua organização em pacotes e correlação entre os possíveis subsistemas.

### 5.1 Visão Geral

A visão geral do Sistema em si é composta por 2 subsistemas:

* Elastic Search (ES): responsável pela busca, correlação e acesso ao corpus do código de defesa do consumidor. Papel de search engine do sistema.
* Sistema Web ou Chatbot: Este determina o fluxo de apresentação, serve de ponte entre o usuário e a search engine, e realiza o filtro de informação entre o resultado do ES e o chatbot do sistema. Ao final, esse sistema gera um output contendo informação filtrada provinda do ES e do contato com o usuário.

## 6. Visão de Processos

O sistema é iniciado através do comando ‘npm start’. Onde executa-se o arquivo StartServers.js pelo Node. Nele são executados dois processos filhos:

* Elastic Search
* Aplicação web contendo a interface do sistema e o chatbot.

## 7. Visão de Implantação

O Elastic Search (5.0.0) foi construído na linguagem java e acessado através de uma API escrita em node (elasticsearch.js). O sistema web foi construído na linguagem Javascript. O ambiente de testes foi realizado utilizando a arquitetura de x64 e em máquinas Linux. E tem como pre-requisito também a instalação dos pacotes Java8 jdk e NodeJs 6.x.

## 8. Visão de Implementação

### 8.1 Visão Geral

Seguindo a influência do padrão arquitetural MVC ,o sistema web foi dividido em pastas : model, view, controller, modules e public. Os arquivos da camada view são responsáveis apenas pelo front-end do sistema. Os arquivos da camada model são as classes atômicas e as classes compostas essenciais para construção do sistema. Os arquivos da camada controller são responsáveis pelo acesso entre o frontend e o backend.

### 8.2 Camadas

### 8.2.1 View

A camada view é responsável pela interação do usuário com o sistema ,nessa camada são realizados etapas como construção de interface de usuário e interação para a configuração das funcionalidades. Contém os view-templates no formato ‘.ejs’.

### 8.2.2 Model

A camada model é responsável pelo armazenamento das classes mais atômicas do projeto ,nessa camada são realizados conexões com o banco de dados: cadastro, login, remoção de registros, consulta de dados.

### 8.2.3 Controller

A camada controller é responsável pela execução de algoritmos complexos como interpretação dos dados da interface para o sistema e o controle do ciclo de vida da aplicação. Todo acesso às páginas são gerenciado aqui através das rotas do módulo express do NodeJs.

### 8.2.4 Modules

A camada Modules contém todos os módulos necessários para a aplicação funcionar. Instalados pelo Node Package Manager. (NPM)

### 8.2.5 Public

A camada public contém todos os arquivos estáticos do sistema. Servindo de suporte para outras camadas do sistema podem ser encontrados nesta pasta arquivos em javascript, css, images e pdf. Também citado como a pasta “asset” em outras arquiteturas de software Node.

## 9. Tamanho e Desempenho

O sistema Chatbot-Cdc possui tamanho em disco de +-30Mb e é desenvolvido para a plataforma web, entretanto o sistema deve ser construído visando a comunicação entre os dois servidores e o desempenho realizado pelos servidores: o elasticsearch pesquisa por relações entre palavras e a keywords digitadas pelo usuário e a interface web fica responsável pela troca de informações e o usuário que não usufrui bastante processamento. O tamanho do corpus é pequeno e fixo contendo apenas o elasticsearch. No entanto a filtragem dos artigos que ocorre na aplicação web (interface) escala baseado na quantidade de históricos de usuários com queixas similares. Quanto mais queixas são realizadas no sistema, maior o histórico do sistema, maior a filtragem de artigos que o sistema deve realizar.

## 10. Qualidade

Visando trazer portabilidade e simplicidade ao sistema. O padrão de arquitetura escolhido MVC foi a solução mais satisfatória para atender a qualidade esperada do sistema. Sistema este que será desenvolvido na linguagem Javascript, utilizando já a linguagem escrita da aplicação anterior, buscador jurídico. Visando uma interface que seja interativa e fácil de se usar.