**PONTÍFICIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS**

**NÚCLEO DE EDUCAÇÃO A DISTÂNCIA**

**Pós-graduação Latu Sensu em Arquitetura de Software Distribuído**

**Luiz Fernando Dias Santos**

**SISTEMA DE GESTÃO E CONTROLE DE INFORMAÇÕES AMBIENTAIS**

Brasília

2020

**Luiz Fernando Dias Santos**

**SISTEMA DE GESTÃO E CONTROLE DE INFORMAÇÕES AMBIENTAIS**

Trabalho de conclusão do Curso de Especialização em Arquitetura de Software Distribuído como requisito parcial à obtenção do título de especialista.

Orientador: Prof. Dr. Pedro A.Oliveira

Brasília

2020

*Ao meu querido pai José Dias, por ter me ensinado o valor do trabalho.*

*A minha família, pelo carinho*

**RESUMO**

Devido aos impactos no meio ambiente decorrente da atividade industrial e empresarial, bem como a adequação a legislação brasileira, torna-se fundamental que a empresa tenha controle das informações do processo de licenciamento ambiental.

Este projeto aborda a criação de sistema que pode ser acessado de qualquer dispositivo ou computador com segurança, a empresa poderá lançar dados de licenciamento ambiental do empreendimento, as multas aplicadas, gerenciamento da obtenção das licenças ambientais do empreendimento, de modo a auxiliar a empresa na construção da política ambiental, a ter uma atuação ecologicamente correta e aprimorar sua gestão ambiental.

O sistema também fornece um repositório centralizado e nacional de informações, com o objetivo de fornecer relatórios a respeito da obtenção de licenciamento e da aplicação das multas, de acordo com o segmento da empresa e do estado.

**Palavras-chave:** Arquitetura de Software, projeto de software, software de gestão ambiental.

**SUMÁRIO**

[1.Objetivos do trabalho 6](#_Toc38746022)

[**2.Descrição Geral da Solução** 7](#_Toc38746023)

[2.1 Apresentação do Problema 7](#_Toc38746024)

[2.2 Definição geral do *Software*(Escopo) 7](#_Toc38746025)

[**3. Definição da Solução** 7](#_Toc38746026)

[3.1 Requisitos Funcionais 7](#_Toc38746027)

[3.2 Requisitos Não-Funcionais 9](#_Toc38746028)

[3.3 Restrições Arquiteturais 12](#_Toc38746029)

[3.4 Mecanismos Arquiteturais 12](#_Toc38746030)

[**4.Modelagem e Projeto Arquitetural** 13](#_Toc38746031)

[4.1 Diagrama de Processo 13](#_Toc38746032)

[4.2 Descrição resumida dos casos de Uso 13](#_Toc38746033)

[4.3 Modelo de Componentes 16](#_Toc38746034)

[4.4 Modelo de Implantação 16](#_Toc38746035)

[**5 Prova de Conceito (POC)/Projeto Arquitetural** 18](#_Toc38746036)

[5.1 Implantação e Implementação 18](#_Toc38746037)

[**7. Conclusão 18**](#_Toc38746038)

[**REFERÊNCIAS 19**](#_Toc38746039)

[**APÊNDICES 20**](#_Toc38746040)

**1.Objetivos do trabalho**

Este trabalho tem como objetivo apresentar uma proposta de arquitetura para o desenvolvimento de um sistema de gestão ambiental, com o intuito de fornecer uma solução de controle de informações das diversas etapas do processo de aquisição da licença de funcionamento do empreendimento, com boa manutenibilidade, portabilidade e usabilidade.

Os objetivos específicos são:

1. Criação de um módulo de autenticação para o controle de acesso dos colaboradores, o módulo deverá permitir diferentes níveis de acesso bem como permitir que a senha seja recuperada através de e-mail. A aplicação deverá fornecer uma solução de autenticação moderna e segura.

2. Criar um módulo de gestão das licenças ambientais, garantindo o atendimento da legislação e dos processos de licenciamento ambiental do empreendimento, conforme o fluxo pertinente ao processo, deverá também observar o cadastro das condicionantes bem como gerenciar o fluxo das licenças.

3. Desenvolver o sistema de tal forma que a camada de persistência funcione bem com diversos bancos de dados, sejam eles proprietários ou não, de forma que se há a necessidade de se adequar a uma solução de banco de dados existente no cliente, que seja feito com o mínimo de impacto em termos de desenvolvimento para sua adaptação – em outras palavras, deverá utilizar um *framework* objeto-relacional.

4. O sistema deverá permitir o cadastro de colaboradores bem como a definição do nível de acesso, propiciando diferentes visões de negócio de acordo com o cargo do usuário no empreendimento. Deverá observar também que o profissional deve ser legalmente habilitado, conforme estabelecido em resolução, logo deverá abranger também a sua habilitação ou especialidade, deverá também fornecer ao usuário uma solução que permita um acompanhamento adequado e eficiente dos prazos do licenciamento.

5. O sistema deverá ser portável – em outras palavras – deverá utilizar tecnologia que facilite o acesso através de diferentes dispositivos e plataformas, deverá ser acessível de um *smartphone*, de um *tablet* ou de um *notebook* por exemplo, através de um framework moderno e responsivo, ou seja, o *layout* da aplicação deverá se adaptar de modo a permitir uma melhor usabilidade do sistema.

6. Criar um módulo para a gestão dos documentos, estudos e projetos necessários à aquisição de licença ambiental, conforme estabelecido pelos órgãos ambientais competentes.

7. Deverá incluir um módulo de cadastros básicos, isso vai tornar o sistema mais dinâmico e permitir que o usuário final possa customizar os valores das tabelas de domínio do sistema, por exemplo, a inclusão de um novo tipo de condicionante ser facilmente ajustada pelo administrador do sistema.

# **2.Descrição Geral da Solução**

2.1 Apresentação do Problema

Para que o empreendimento esteja de acordo com a legislação vigente e de modo a conciliar o desenvolvimento econômico com a conservação do meio ambiente, é necessário que o empreendedor buscar o licenciamento junto aos órgãos competentes, logo, é fundamental que os responsáveis pelo empreendimento estejam guarnecidos de informação.

Além do que foi exposto acima, para que a empresa obtenha sua adequação as normas do mercado, torna-se necessário o desenvolvimento de um sistema que possibilite a empresa ter uma maior aderência aos padrões do mercado e das normas, em especial, a ISO 14001.

2.2 Definição geral do *Software*(Escopo)

O Sistema SIGA ( Sistema Integrado de Gestão Ambiental ) foi criado com o propósito de auxiliar o processo de aquisição da licença de software para o empreendimento.

Deverá ser acessível por diferentes aparelhos, deverá ser seguro, uma vez que possuí informações importantes do empreendimento e possibilitar o acompanhamento de todo o processo.

# **3. Definição da Solução**

3.1 Requisitos Funcionais

Módulo de autenticação

O sistema deverá fornecer autenticação e autorização para a aplicação, deverá ser seguro de forma a proteger as informações contra ataques e outras formas de intrusão. A tecnologia escolhida foi o *Spring Security*, que está contido no ecossistema do *framework*. O *Spring Security* também permite integração com outros protocolos, como por exemplo *Oauth.*

O modulo de autenticação deverá fornecer um modo que o usuário recupere sua senha através de e-mail bem como permitir que um usuário especifico seja desativado. Através do usuário informado, o sistema vai definir o perfil de acesso. Abaixo são descritos os perfis:

**Administrador:** O administrador do sistema deverá ter acesso ao cadastro de usuários e de documentos, estudos ou projetos. Terá acesso a todas as etapas do processo de licenciamento. O usuário com o perfil de administrador poderá acompanhar todas as etapas do processo de licenciamento. O administrador do sistema terá acesso ao módulo de cadastros básicos.

**Gestão:** O usuário do sistema com o perfil de gestor terá acesso a relatórios gerenciais, bem como visualizar detalhes dos documentos a respeito do processo. Entende-se como gestor o responsável pelo empreendimento.

**Operacional:** Vai permitir o cadastro das informações das licenças ambientais. Entende-se como operacional o profissional habilitado.

Módulo de licenciamento ambiental.

De acordo com a Resolução nº 237, de 19 de dezembro de 1997, definimos licenciamento ambiental como “procedimento administrativo pelo qual o órgão ambiental competente licencia a localização, instalação, ampliação e a operaçãode empreendimentos e atividades utilizadoras de recursos ambientais *,* consideradas efetiva ou potencialmente poluidoras ou daquelas que, sob qualquer forma, possam causar degradação ambiental, considerando as disposições legais e regulamentares e as normas técnicas aplicáveis ao caso”.

Deverá armazenar as licenças ambientais bem como armazenar as informações nas diferentes etapas do licenciamento. As licenças são descritas abaixo:

Licença Prévia (LP) – Concedida na etapa de planejamento, atesta a viabilidade de concepção e determina as condicionantes que deverão ser atendidas para a próxima fase.

Licença de Instalação (LI) – Fornecida com o intuito de autorizar o responsável pelo empreendimento a construção ou instalação da obra de acordo com as especificações constantes nos planos, projetos e especificações aprovados, principalmente as medidas de mitigação e controle ambiental para a próxima fase.

Licença de Operação (LO) – Licença de funcionamento. Após a constatação de que todas as exigências e condicionantes foram cumpridas. O empreendedor fica responsável por implementar todas as medidas de mitigação e controle ambiental, sob pena de ter a LO suspensa ou caçada pelo órgão responsável. O sistema deverá entender que uma licença transita entre os três tipos informados acima, deverá, portanto, compreender o fluxo e guardar o histórico do licenciamento. O módulo de licenciamento ambiental deverá abordar também o cadastro de condicionantes.

Módulo Administrativo

No módulo administrativo serão cadastrados os usuários do sistema, bem como atribuição das suas habilitações e controle, conforme requerido por legislação. O módulo administrativo deve incluir também os cadastros básicos da aplicação.

3.2 Requisitos Não-Funcionais

Framework Objeto-Relacional

A aplicação deverá funcionar de acordo com a solução de banco de dados encontrada pelo cliente, em outras palavras, o sistema deverá ser independente de banco de dados de forma que o responsável pelo empreendimento possa optar por uma solução paga de banco de dados já adquirida pelos responsáveis pelo empreendimento, bem como utilizar uma opção gratuita.

|  |  |
| --- | --- |
| **Estímulo** | Mudança de Banco de dados. |
| **Fonte de estímulo** | A empresa optou por uma solução de banco de dados com um melhor custo benefício, ou decisão do gestor. |
| **Ambiente** | Funcionamento, carga normal |
| **Artefato** | Camada de persistência da aplicação |
| **Resposta** | Não houve impactos, porém é necessário um estudo prévio e migração dados da solução antiga para a nova |
| **Medida da resposta** | Funcionamento normal da aplicação. |

* Manutenibilidade – O sistema deverá ter manutenção facilitada

O sistema deverá construído observando as boas práticas de desenvolvimento, com a finalidade de proporcionar a facilidade de manutenção evolutiva e de correção de *bugs*.

|  |  |
| --- | --- |
| **Estímulo** | Ajuste no cadastro de condicionantes para incluir o tipo de condicionante |
| **Fonte de estímulo** | Correção de bug |
| **Ambiente** | Carga normal, em funcionamento. |
| **Artefato** | Módulo de gestão das licenças. |
| **Resposta** | Manutenção facilitada sobretudo ao uso de JPA na arquitetura. |
| **Medida da Resposta** | O sistema deverá utilizar um *framework* objeto-relacional moderno e eficiente. |

* Segurança – O sistema deverá implementar a segurança através do protocolo Oauth2

|  |  |
| --- | --- |
| **Estímulo** | Tentar realizar o acesso ao sistema sem estar logado. |
| **Fonte de estímulo** | Tentativa de acesso não identificado. |
| **Ambiente** | Em funcionamento, com carga normal. |
| **Resposta** | O sistema redireciona o usuário a tela de login. |
| **Medida da resposta** | O sistema não deverá permitir o acesso sem que o usuário esteja devidamente cadastrado. |

* Desempenho – O sistema deverá executar sem lentidão ou travamento

|  |  |
| --- | --- |
| **Estimulo** | Navegação nas telas do sistema. |
| **Fonte de estímulo** | Usuário logado. |
| **Ambiente** | Em funcionamento, com carga normal. |
| **Artefato** | Módulos do sistema. |
| **Resposta** | Normal. |
| **Medida da resposta** | O interagiu entre as telas do sistema de forma normal, sem travamentos ou lentidão. |

* Acessibilidade - O sistema deve suportar o acesso através de diferentes mecanismos e plataformas

|  |  |
| --- | --- |
| **Estímulo** | Usuário com o perfil operacional logado no sistema. |
| **Fonte do estímulo** | Usuário com o perfil operacional acessando o sistema a partir de um *tablet*. |
| **Ambiente** | Em funcionamento, com carga normal. |
| **Artefato** | Módulo de autenticação. |
| **Resposta** | A tela do módulo de autenticação funcionou perfeitamente, devido a tecnologia *frontend* utilizada. |
| **Medida da Resposta** | Identidade visual semelhante em diversas resoluções, teste positivo. |

* Interoperabilidade – O sistema deverá ter um mecanismo de comunicação com terceiros

|  |  |
| --- | --- |
| **Estímulo** | Teste de conexão |
| **Fonte do estímulo** | Script de teste externo simulando conexão com o sistema. |
| **Ambiente** | Funcionamento normal, com carga normal. |
| **Artefato** | Módulo de comunicação. |
| **Resposta** | Foi enviada uma mensagem ao sistema com sucesso. |
| **Medida da resposta** | A comunicação ocorreu com sucesso. |

3.3 Restrições Arquiteturais

* O sistema deverá utilizar um *framework* objeto-relacional, de modo a não sofrer impactos na hipótese da alteração do banco de dados;
* Deverá ser desenvolvido utilizando a abordagem de serviços;
* Deverá utilizar o protocolo *Oauth 2.0* para autenticação;
* Deverá ser portável de modo a funcionar nas mais diferentes plataformas, sem impactar na usabilidade do produto.

3.4 Mecanismos Arquiteturais

|  |  |  |
| --- | --- | --- |
| **Mecanismo de Análise** | **Mecanismo de *Design*** | **Mecanismo de implementação** |
| Comunicação entre processos | Servidor de aplicação e servidor HTTP | *Wildfly 18.01* |
| Versionamento | Versionamento do código fonte da aplicação | *Github* |
| Autenticação e Autorização | Verificação das credenciais através de *token* | *Spring Security, Spring Boot e Oauth 2.0*. |
| *Frontend* | Interface de comunicação com os usuários do sistema | Angular, Bootstrap |
| *Build* | Geração de artefato. | *Gradle* |
| *Log* | Mecanismo de log da aplicação | Log4J |
| *ESB* | Integração de mensagens da aplicação com outros sistemas/barramento | RabbitMQ |

4.Modelagem e Projeto Arquitetural

Através desta seção, vamos compreender alguns diagramas que permitem um melhor entendimento da estrutura fundamental da aplicação.

4.1 Diagrama de Processo

O diagrama de processo retrata as principais etapas do fluxo de licenciamento ambiental, conforme é descrito na resolução CONAMA nº 237, ou seja, através do diagrama, pode-se visualizar o fluxo básico que o sistema deverá atender.

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

4.2 Descrição resumida dos casos de Uso

Caso de uso 1: Cadastro de licença ambiental

**Descrição resumida**

Este caso de uso deve permitir ao usuário com o perfil operacional cadastrar os dados de uma licença ambiental, ele deve fornecer as informações e poderá anexar arquivos, o sistema deverá guardar o usuário que está entrando com as informações. O sistema também deverá atentar aos prazos e ao correto fluxo do licenciamento ambiental, conforme legislação específica.

|  |  |
| --- | --- |
| **ID** | 1 |
| **Estória do usuário** | Cadastrar licença ambiental |
| **Criador** | Com usuário com o perfil operacional |
| **Descrição** | Necessidade de cadastrar as informações das licenças ambientais, de modo que o sistema guarde o correto fluxo do processo, assim como anexar arquivo PDF e JPG referentes ao licenciamento |
| **Valor do negócio** | Módulo principal do sistema |
| **Prioridade** | Alta |

Caso de uso 2 – Cadastro de Condicionante

**Descrição Resumida**

De acordo com Coli e Dias (2017, pag. 22 ) temos que:

As condicionantes representam obrigações que são impostas pelo órgão ambiental, com base nos estudos e monitoramentos desenvolvidos ao longo do licenciamento e na análise dos projetos apresentados, que determinam a forma em que foi autorizada a concepção implantação ou operação da atividade.

|  |  |
| --- | --- |
| **ID** | 2 |
| **Estória do usuário** | Cadastrar condicionante |
| **Criador** | Com usuário com o perfil operacional |
| **Descrição** | Utilidade de cadastrar as obrigações que o responsável ou responsáveis pelo empreendimento com os prazos, também é preciso selecionar um processo de licenciamento e atrelar essa condicionante cadastrada. |
| **Valor do negócio** | Módulo principal do sistema |
| **Prioridade** | Alta |

Caso de uso 3 – Recuperação de senha para autenticação no sistema

**Descrição Resumida**

Este caso de uso detalha a funcionalidade de recuperar a senha através de uma opção na tela de login, através desta, será encaminhado um e-mail para o usuário cadastrado para redefinição de senha.

|  |  |
| --- | --- |
| **ID** | 3 |
| **Estória do usuário** | Recuperar a senha para login no sistema |
| **Criador** | Com usuário com o perfil operacional |
| **Descrição** | É preciso que o sistema forneça na tela de login uma funcionalidade de envio de senha para o e-mail do usuário cadastrado |
| **Valor do negócio** | Módulo de autenticação do sistema, fundamental para realizar o *login.* |
| **Prioridade** | Médio |

Caso de uso 4 – Cadastro de Usuário

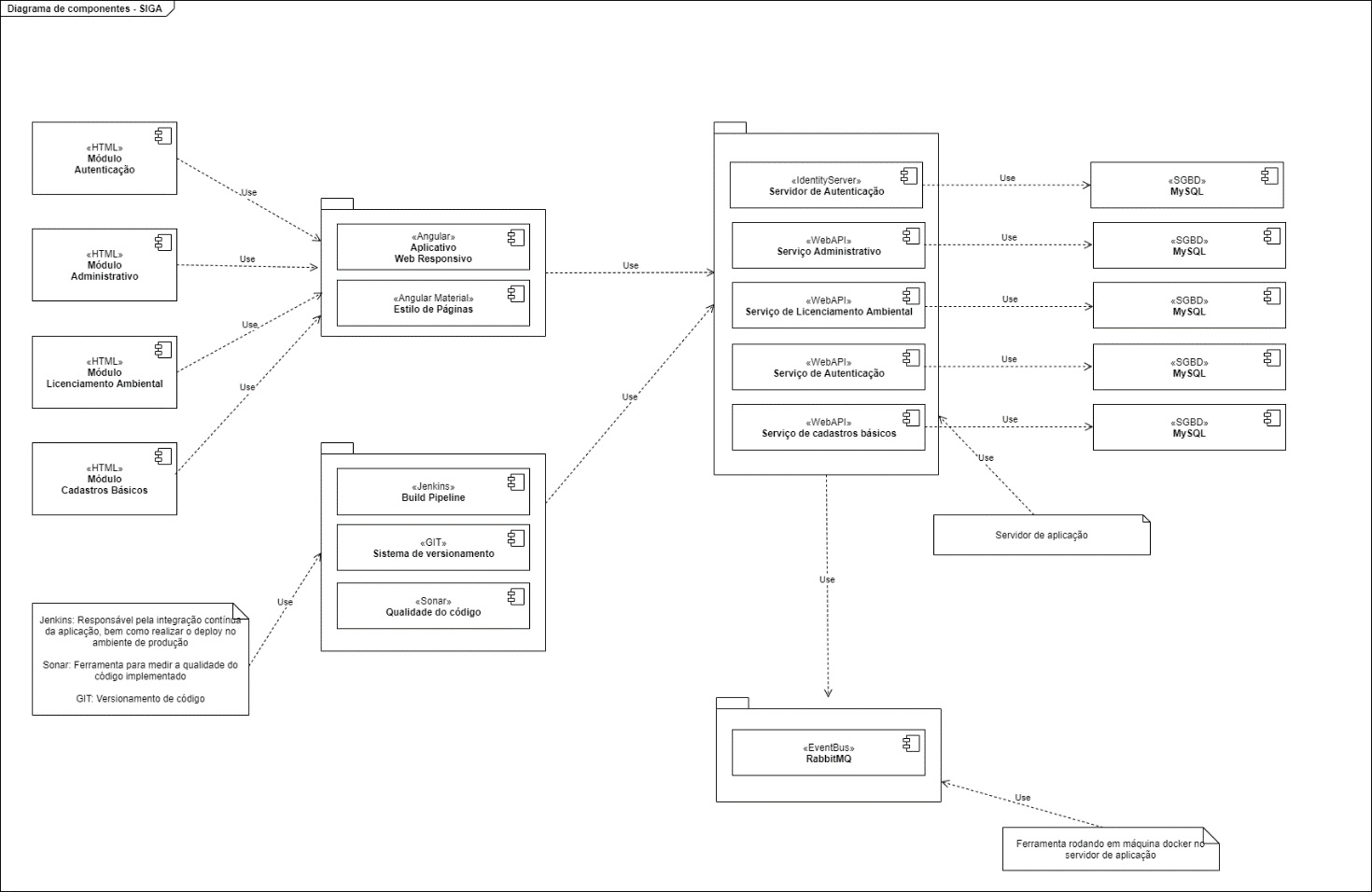
**Descrição Resumida**

Caso de uso que especifica o cadastro do usuário no módulo administrativo, importante ressaltar que o sistema deverá ser seguro e não permitir o cadastro de qualquer colaborador, já o usuário com o perfil de administrador poderá inclusive criar um pré-cadastro

|  |  |
| --- | --- |
| **ID** | 4 |
| **Estória do usuário** | Realizar o cadastro de usuário |
| **Criador** | Com usuário com o perfil administrador cadastra um usuário |
| **Descrição** | Necessidade de cadastrar um usuário ao sistema, ele pode utilizar os dados das redes sociais para realizar uma espécie de pré-cadastro. |
| **Valor do negócio** | Módulo administrativo |
| **Prioridade** | Alta |

4.3 Modelo de Componentes

Como cita Melo(2010, pág. 214), o diagrama de componentes é uma especificação de um conjunto de construções que especifica um sistema de software, abaixo segue diagrama:



4.4 Modelo de Implantação

Abaixo está o modelo de implantação do sistema, De acordo com MARTINS (2007, pág. 444), temos que:

O diagrama de implantação mostra como várias partes físicas do sistema serão distribuídas no ambiente computacional e se comunicarão entre si, como, por exemplo, máquinas clientes de usuário, servidores de banco de dados, servidores de comunicação e outros.

Tela de celular com texto preto sobre fundo branco

Descrição gerada automaticamente

Ainda, de acordo com Guedes (2018, p. 363) destaca-se que:

O diagrama de implantação é o diagrama com a visão mais física da UML. Enfoca a questão da organização da arquitetura física sobre o qual o software será implantado e executado, em termos de hardware ou seja, as máquinas(computadores pessoais, servidores etc.) que suportarão o sistema, além de definir como estas estarão conectadas e por meio de quais protocolos se comunicarão e transmitirão informações.

Acima é exposto de como deverá ficar distribuído o sistema na implantação. Para o *deploy* da aplicação será utilizada o Jenkins, e na máquina teremos o *Sonar* para o controle de qualidade de código e o GIT para o versionamento de código. Será utilizada também para a mensageria no sentido de fornecer integração com outras aplicações um *container* Docker com a aplicação RabbitMQ.

A aplicação utiliza Oauth2, haverá a possibilidade de utilizar o Google, Facebook e LinkedIn para autenticação bem como recuperar a senha através de e-mail conforme diagrama acima, como será utilizado serviço na web, não há a necessidade de se ter uma máquina de servidor de e-mail. Abaixo segue a descrição dos componentes acima:

|  |  |
| --- | --- |
| **Componente** | **Descrição** |
| Navegador | *Software* utilizado para acessar o aplicativo *web* |
| Servidor de Integração contínua | Servidor onde deverá ficar o software de automação, o sistema de versionamento GIT e também a ferramenta Sonar |
| Servidor de aplicação | Máquina onde ficará distribuída a aplicação *backend* |
| Servidor mySQL | Máquina que vai hospedar a base de dados |
| Docker | O Docker é uma plataforma de código aberto para criação de *containers* |

5 Prova de Conceito (POC)/Projeto Arquitetural

5.1 Implantação e Implementação

Como é citado por Pressman (2016), entendemos como prova de conceito é um teste que demostra a viabilidade de uma tecnologia, além disso, visa também verificar o atendimento dos requisitos funcionais e não funcionais. Abaixo as validações dos requisitos não funcionais:

O sistema deverá utilizar um *framework* objeto-relacional

Este Requisito não-funcional foi definido devido a necessidade de se construir uma aplicação moderna e independente de banco de dados.

Os critérios para validação são descritos a seguir:

* A aplicação não deverá sofrer retrabalho se o cliente adotar uma solução de banco de dados divergente do *MySQL* em termos de código. A preocupação deverá ser somente migração de dados.
* O sistema não deverá ser impactado, em outras palavras, o sistema deverá utilizar bem a solução de persistência utilizada, de modo que não impacte na performance.

O sistema deverá ser responsivo, pois será acessado de diferentes dispositivos

Devido a evolução dos dispositivos portáteis, torna-se fundamental que o sistema utilize um *framework* de *frontend* moderno e responsivo, de modo que a página não sofra distorções ou que a experiência de acessar o sistema através de um celular ou um *smartphone* não prejudique a usabilidade.

Os critérios de validação são descritos a seguir:

* O sistema deverá se manter acessível de modo a não apresentar distorções ou que a usabilidade seja prejudicada, uma vez que o sistema é acessado através de um *tablet* ou *smartphone.*
* O sistema deverá ser compatível com os navegadores mais utilizados pelo mercado.
* O sistema também não deverá sofrer prejuízo em relação a seu esquema de cores.

O sistema deverá ser seguro

Como o sistema contém informações importantes sobre o processo de licenciamento ambiental do empreendimento, foi utilizado o protocolo *Oauth2* utilizando a implementação fornecida pelo *framework* *Spring Security*.

Os critérios de validação são informados abaixo:

* O sistema não deverá permitir que o usuário não autenticado acesse páginas com informação sensível do sistema sem informar as suas credenciais.
* O sistema deve possuir o total controle das credenciais, de modo que o usuário de determinado perfil não acesse funcionalidades de outro, ou seja, o sistema tem que gerenciar de modo apropriado as permissões.
* Se a sessão do usuário estiver expirada, então o sistema deverá redirecionar o usuário para a tela de login.
* Se o usuário tentar acessar uma URL diferente do *login* sem estar autenticado, então ele vai ser redirecionado para a tela de login.
* O sistema deve permitir que o usuário recupere a senha através de e-mail.

O sistema deverá possuir um bom desempenho

É necessário prover ao usuário um sistema que possua um bom desempenho de forma que a navegação seja fluída e diminuir ao máximo a possibilidade de travamentos ou lentidão.

Para validação, é necessário:

* Que o sistema possua uma boa navegabilidade, sem travamentos ou lentidão.
* Que o sistema seja programado de forma que utilize o *framework* objeto-relacional de modo correto, para que não sobrecarregue o banco de dados com consultas desnecessárias bem como que otimize o processamento.
* Que o *software* seja programado de acordo com as boas práticas de desenvolvimento, evitando o retrabalho e que o recurso computacional não seja desperdiçado com código que não otimize o *hardware* disponível, como código que carregue uma mesma lista em memória várias vezes, por exemplo.

Testabilidade – O sistema deverá ter facilidade de teste, ou utilizar alguma ferramenta que facilidade sua manutenção.

A qualidade é o requisito mais importante do *software* a ser desenvolvido, uma vez que se o *software* possuir boa qualidade, vai sofrer menos manutenção, em consequência disso, vamos economizar em termos de mão de obra. Se o *software* tem boa qualidade, então a manutenção vai ser facilitada. Abaixo são enumerados os elementos para validação:

* O sistema deverá permitir o teste unitário através de alguma ferramenta ou *framework.*
* O sistema deve ser desenvolvido dentro de um fluxo de trabalho onde as funcionalidades sejam testadas utilizando um *bug tracker* como o Mantis ou ferramenta similar.

O sistema deverá ter boa usabilidade

O sistema deverá ser fácil de utilizar, facilitando o treinamento com o *software* desenvolvido, bem como o seu acesso através de dispositivo portátil.

7. Conclusão

O trabalho apresenta a arquitetura de um sistema de gestão ambiental baseado em *Spring Boot, Spring Security* utilizando autenticação Oauth 2.0 e JPA. A arquitetura proposta é simples, porém eficiente para uma equipe pequena de desenvolvedores.

REFERÊNCIAS

BRASIL. Resolução nº 237 de 19 de dezembro de 1997. CONAMA

GUEDES, Gilleanes T.A . *UML – Uma Abordagem prática.*3ª ed. São Paulo, Editora Novatec, 2018.

COLI, Adriana; Dias, Pedro. *O Setor Elétrico e o meio ambiente*, 1º ed. Rio de Janeiro, Editora Synergia, 2017

MELO, Ana Cristina. *Desenvolvendo Aplicações com UML 2.2.* 3º Ed. Rio de Janeiro, Editora Brasport, 2010

MARTINS, José Carlos Cordeiro. *Tecnicas Para Gerenciamento de Projeto de Software.* 1º Ed. Rio de Janeiro, Editora Brasport, 2007.

PRESSMAN, Roger S. ; Bruce R. Maxim. Engenharia de Software, Uma Abordagem Profissional, 8° ed. Porto Alegre: AMGH, 2016.

APÊNDICES

*Usuário com perfil de administrador*

*Usuário:* admin

*Senha:* admin

*Usuário com o perfil de operacional*

*Usuário:* cliente

*Senha:* cliente