



**UNIVERSIDADE ESTADUAL DA PARAÍBA
CAMPUS I CAMPINA GRANDE
CENTRO DE CIÊNCIAS E TECNOLOGIA
CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO**

ADALBERTO FRANCISCO MONTEIRO NETO

SIGNUTES: SISTEMAS INTEGRADOS DE GESTÃO PARA O NUTES

**CAMPINA GRANDE
2017**

**UNIVERSIDADE ESTADUAL DA PARAÍBA
CAMPUS I CAMPINA GRANDE
CENTRO DE CIÊNCIAS E TECNOLOGIA**

ADALBERTO FRANCISCO MONTEIRO NETO

SIGNUTES: SISTEMAS INTEGRADOS DE GESTÃO PARA O NUTES

Trabalho de Conclusão de Curso de Graduação em Ciência da Computação da Universidade Estadual da Paraíba, como requisito à obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Frederico Moreira Bublitz.

**CAMPINA GRANDE
2017**

É expressamente proibido a comercialização deste documento, tanto na forma impressa como eletrônica. Sua reprodução total ou parcial é permitida exclusivamente para fins acadêmicos e científicos, desde que na reprodução figure a identificação do autor, título, instituição e ano do Trabalho de Conclusão de Curso.

M775s Monteiro Neto, Adalberto Francisco.
SIGNUTES [manuscrito] : Sistemas Integrados de Gestão para o NUTES / Adalberto Francisco Monteiro Neto. - 2017
41 p. : il. colorido.

Digitado.

Trabalho de Conclusão de Curso (Graduação em Computação) - Universidade Estadual da Paraíba, Centro de Ciências e Tecnologia, 2017.

Orientação : Prof. Dr. Frederico Moreira Bublitz, Coordenação do Curso de Computação - CCT.

1. Sistema integrado. 2. Desenvolvimento de software. 3. Desenvolvimento ágil. 4. Aplicações Web.

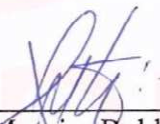
21. ed. CDD 005.3

Adalberto Francisco Monteiro Neto


SIGNUTES: SISTEMAS INTEGRADOS DE GESTÃO PARA O NUTES

Trabalho de Conclusão de Curso de Graduação em Ciência da Computação da Universidade Estadual da Paraíba, como requisito à obtenção do título de Bacharel em Ciência da Computação.

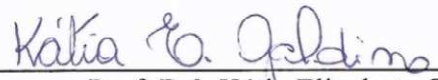
Aprovada em 30 de outubro de 2017.



Prof. Dr. Frederico Moreira Bublitz (UEPB)
Orientador(a)



Prof. Dr. Daniel Scherer
Examinador(a)



Prof. Dr^a. Kátia Elizabete Galdino
Examinador(a)

Aos queridos Adalberto F. Monteiro (*in
memoriam*) e Luiz Alves da Silva (*in
memoriam*), DEDICO.

AGRADECIMENTOS

A toda a minha família, em especial aos meus pais e a minha tia Lucinha, pelo constante suporte e amor, dedicados na vida e ao longo desta árdua jornada.

A minha esposa Alessandra, pela paciência em conseguir aguentar desde um namorado que passava as noites de sexta feira estudando até um marido que continua a fazer a mesma coisa.

Ao amigo Túlio H. Costa, pelos constantes ensinamentos e cobranças, sempre positivas.

A toda a equipe do NUTES, em especial a galera do Laboratório de Engenharia de Software (Sala 16). Sempre considerarei esta, a minha casa.

Ao amigo Lucas Barbosa, pela sincera amizade e em lembrança das noites em claro, sempre que fazíamos os trabalhos das disciplinas às vésperas da entrega.

Ao amigo Marcelo Ricelly, “minha dupla”, pelos trabalhos, sempre mal interpretados pelo querido professor Mota (*in memoriam*).

Ao meu orientador, professor Dr. Frederico Moreira Bublitz, pela confiança e oportunidade dadas.

“Se você faz o que todo mundo faz, chega aonde todos chegam. Se você quer chegar aonde a maioria não chega, precisa fazer algo que a maioria não faz.”

Roberto Shinyashiki

RESUMO

Diante do cenário nacional de investimentos em ciência, tecnologia e inovação, principalmente na área de Saúde, novas parcerias vêm sendo firmadas com o objetivo de fomentar pesquisas. Nesse sentido, o Núcleo de Tecnologias Estratégicas em Saúde (NUTES) atua como facilitador de inovação para o governo, empresas e centros de desenvolvimento. O NUTES é resultado de um convênio entre a Universidade Estadual da Paraíba (UEPB) e a Financiadora de Estudos e Projetos (FINEP), sendo vinculado ao Ministério da Ciência, Tecnologia e Inovação (MCTI), Ministério da Saúde (MS) e executado com apoio da Fundação Parque Tecnológico da Paraíba (PaqTcPB), tem como missão atuar em Sistemas de Saúde de forma inovadora e atendendo às agências reguladoras, gerando negócios com organizações públicas e privadas. Por se tratar de um órgão ligado à universidade, o NUTES possui uma rotatividade ampla de pessoas, fazendo com que o sigilo e confidencialidade de informações sejam primordiais no dia-a-dia, bem como, o gerenciamento de equipamentos e colaboradores. Diante dessas necessidades, foi desenvolvido um sistema interno de gestão para, *a priori*, cadastramento de colaboradores e controle e gerenciamento de portas de acesso. A solução visava a um sistema que fosse multiplataforma, de fácil manutenção e alta escalabilidade. Para tanto, foi desenvolvido um sistema Web, utilizando as linguagens HTML5, CSS3, Javascript e PHP, em uma metodologia ágil de desenvolvimento e adotando-se o *scrum* como padrão de gerenciamento do processo. O presente trabalho tem como objetivo o desenvolvimento de um Sistema Interno de Gestão (SIG) para que, problemas internos, relacionados à cadastramento de colaboradores e controle de acesso, sejam solucionados, bem como, a geração automática do Termo de Confidencialidade e Sigilo e a Declaração de Possíveis Conflitos de Interesse sejam implementadas, proporcionando agilidade ao processo do NUTES.

Palavras-Chave: Sistema integrado, Ferramentas de Gestão, Desenvolvimento de *Software*, Aplicações Web, Desenvolvimento ágil.

ABSTRACT

In the face of the national scenario of investments in science, technology and innovation, especially in the Health area, new partnerships have been established with the aim of promoting research. In this sense, the Center for Strategic Technologies in Health (NUTES) acts as a facilitator of innovation for the government, companies and development centers. It is the result of an agreement between the State University of Paraíba (UEPB) and the Studies and Projects Funding Agency (FINEP), being linked to the Ministry of Science, Technology and Innovation (MCTI), Health Ministry (MS) and executed with the Brazilian Federal Government and the Technology Park Foundation of Paraíba state (PaqTcPB). The NUTES's mission is to act in Health Systems in an innovative way and attending regulatory agencies, generating business with public and private organizations. Because it is an organ connected to the university, it has a wide turnover of people, making the confidentiality of information essential in the day-to-day, as well as the management of equipment and employees. In view of these needs, an internal management system was developed to prioritize employee registration and control, and management of environment's access. The solution was aimed at a system that was multiplatform, easy to maintain and high scalability, for that, a Web system was developed, using technologies like HTML5, CSS3, Javascript and PHP, in an agile development methodology and adopting scrum as standard of process management. The present work has the objective of developing an Internal Management System (GIS) so that internal problems related to employee registration and access control be solved, as well as the automatic generation of the Confidentiality Agreement and the Statement of Possible Conflicts of Interest are implemented, providing agility to the NUTES process.

Keywords: Integrated System, Management Tools, Software Development, Web Applications, Agile Development.

LISTA DE ILUSTRAÇÕES

Figura 1: Modelo de processo de projeto (SOMMERVILLE, 2013)	17
Figura 2: Fluxo da etapa de testes (SOMMERVILLE, 2013).....	18
Figura 3: Processo de Scrum (SOMMERVILLE, 2013)	20
Figura 4: Arquitetura MVC aplicada a aplicações Web (SOMMERVILLE, 2013).....	21
Figura 5: Representação do padrão DAO (Adaptado de MACORATTI).....	22
Figura 6: Padrão MVC aplicado para cada módulo do SIGNUTES.....	27
Figura 7: Tela de autenticação	33
Figura 8: Tela principal do sistema.	34
Figura 9: Menu lateral do sistema.....	34
Figura 10: Visão do cadastro de colaboradores.	35
Figura 11: Visão da tela listar colaboradores.	35
Figura 12: Tela de edição de permissões de acesso.	38
Figura 13: Tela de edição de portas de acesso.	39

LISTA DE DIAGRAMAS

Diagrama 1: Casos de uso do sistema SIGNUTES.....	25
Diagrama 2: Arquitetura do SIGNUTES	27
Diagrama 3: Casos de uso Cenário 1.	29
Diagrama 4: Casos de uso Cenário 2.	30

SUMÁRIO

1.	INTRODUÇÃO.....	12
2.	OBJETIVOS.....	14
2.1.	OBJETIVO GERAL.....	14
2.2.	OBJETIVOS ESPECÍFICOS	14
3.	REVISÃO DA LITERATURA.....	15
3.1.	PROCESSOS DE <i>SOFTWARE</i>	15
3.1.1.	Especificação de software	15
3.1.2.	Projeto e implementação de software.....	16
3.1.3.	Validação de software	17
3.1.4.	Evolução de software	18
3.2.	DESENVOLVIMENTO ÁGIL DE SOFTWARE	18
3.3.	GERENCIAMENTO ÁGIL DE PROJETOS	19
3.4.	PADRÕES DE PROJETO E ARQUITETURA	20
3.4.1.	Padrão de arquitetura Modelo-Visão-Controlador (MVC)	21
3.4.2.	Padrão de Projeto <i>Data-Access-Object</i> (DAO)	21
3.5.	APLICAÇÃO WEB	22
3.6.	<i>CASCADING STYLE SHEETS</i> (CSS).....	23
3.7.	A LINGUAGEM <i>JAVASCRIPT</i>	23
3.8.	A LINGUAGEM PHP.....	24
4.	ESPECIFICAÇÕES DO SIGNUTES	25
5.	PROCESSOS ASSOCIADOS AO DESENVOLVIMENTO DO SIGNUTES	25
5.1.	PROCESSO DE SOFTWARE	25
5.2.	GERENCIAMENTO DE PROCESSO	26
5.3.	PADRÃO DE ARQUITETURA.....	26
6.	REQUISITOS DO SISTEMA.....	28
6.1.	CENÁRIO 1	28
6.2.	CENÁRIO 2	28
6.3.	REQUISITOS ENTREGA 1	29
6.4.	REQUISITOS ENTREGA 2	30
7.	DESENVOLVIMENTO DO SISTEMA.....	31
7.1.	FERRAMENTAS UTILIZADAS NO PROCESSO DE DESENVOLVIMENTO	31

7.2.	DESENVOLVIMENTO DA PRIMEIRA ENTREGA.....	33
7.2.1.	Tela de autenticação	33
7.2.2.	Tela principal do sistema.....	33
7.2.3.	Menu lateral do sistema.....	34
7.2.4.	Tela do cadastro de colaboradores	34
7.2.5.	Tela de listar colaboradores.....	35
7.2.6.	Padrão de <i>back-end</i> do sistema	35
7.2.7.	<i>Feedback</i> da primeira entrega	37
7.3.	DESENVOLVIMENTO DA SEGUNDA ENTREGA.....	38
7.3.1.	Tela de permissões de acesso	38
7.3.2.	Tela de edição de portas de acesso	38
7.3.3.	<i>Feedback</i> da segunda entrega.....	39
8.	CONSIDERAÇÕES FINAIS	40
	REFERÊNCIAS	41

1. INTRODUÇÃO

Diante dos constantes investimentos do governo em ciência, tecnologia e inovação (PIONTKIEWICZ, et al, 2017), parcerias vêm sendo firmadas com objetivo de alavancar pesquisas, principalmente na área de Saúde (TENÓRIO, et al, 2017).

O Núcleo de Tecnologias Estratégicas em Saúde (NUTES), resultado de um convênio entre a Universidade Estadual da Paraíba (UEPB) e a Financiadora de Estudos e Projetos (FINEP), sendo vinculado ao Ministério da Ciência, Tecnologia e Inovação (MCTI), Ministério da Saúde (MS) e executado com apoio da Fundação Parque Tecnológico da Paraíba (PaqTcPB), tem como missão atuar em Sistemas de Saúde de forma inovadora e atendendo às agências reguladoras, gerando negócios com organizações públicas e privadas.

O NUTES tem como um dos seus principais objetivos trazer inovação à área de saúde, para isso, o núcleo conta com diversas frentes de atuação, dentre as quais destacam-se: (i) Manufatura Aditiva que tem como objetivo ajudar no planejamento de procedimentos cirúrgicos com serviços de prototipagem rápida, aplicada na área Médico-Odontológica; (ii) Avaliação de Conformidade que visa a minimizar a possibilidade de um dispositivo médico, que possui um *software* embarcado, causar danos aos pacientes ou aos operadores do dispositivo; (iii) Usabilidade, tem como objetivo avaliar a usabilidade em *software* de dispositivos eletromédicos, com o intuito de evitar riscos, erro humano e fatores que possam influenciar no uso do dispositivo; (iv) Desenvolvimento de *Software*, possuindo competência para o desenvolvimento de sistemas embarcados em dispositivos eletromédicos, sistemas para tratamento de imagens, processamento de sinais, sistemas de informação e integração por meio de soluções de Conectividade.

Para atender a essa demanda, o NUTES possui uma equipe interdisciplinar de professores, alunos de graduação e pós-graduação, além de profissionais de nível superior e técnicos. Por ser um órgão vinculado à universidade, possui uma alta rotatividade de pessoas e profissionais associados, que lidam diariamente com documentos e tecnologias confidenciais. Há, portanto, a necessidade de um controle interno para a gestão de pessoas, que possibilite o registro de informações, a rastreabilidade destas e o gerenciamento do sigilo das pessoas, além da segurança no interior dos laboratórios. Para

atender às necessidades quanto à gestão interna do NUTES, foi idealizado um Sistema Integrado de Gestão (SIG).

Neste sentido, o presente trabalho tem como objetivo o desenvolvimento de um SIG para que, problemas internos, relacionados à cadastramento de colaboradores e controle de acesso, sejam solucionados, bem como, a geração automática do Termo de Confidencialidade e Sigilo e a Declaração de Possíveis Conflitos de Interesse sejam implementadas, proporcionando agilidade ao processo do NUTES.

2. OBJETIVOS

2.1.OBJETIVO GERAL

Desenvolver um SIG para apoio as atividades de cadastro de colaboradores, controle de acesso e geração automática do Termo de Confidencialidade e Sigilo e a Declaração de Possíveis Conflitos de Interesse.

2.2.OBJETIVOS ESPECÍFICOS

- a) Conhecer as etapas do desenvolvimento de *software*;
- b) Adequar-se a um processo ágil de desenvolvimento de *software*;
- c) Implementar uma arquitetura que vise a padronizar e facilitar a manutenção e atualização do *software*;
- d) Realizar análise de adequação do que foi desenvolvido com o que foi solicitado;
- e) Destacar aspectos positivos e negativos do processo de desenvolvimento bem como do produto final.

3. REVISÃO DA LITERATURA

3.1.PROCESSOS DE *SOFTWARE*

De acordo com Sommerville (2013), um processo de *software* é definido como sendo uma união de atividades relacionadas que conduzem à produção de um produto de *software*. O processo pode ocorrer a partir do zero ou por meio de modificações de sistemas existentes. Apesar de existirem muitos processos diferentes, todos devem incluir quatro etapas fundamentais para a Engenharia de *Software*:

- a) especificação do *software*: levantamento de requisitos funcionais e não funcionais;
- b) projeto e implementação de *software*: produzir o software de forma que atenda às especificações;
- c) validação de *software*: garantia de que o *software* atenda às necessidades do cliente;
- d) evolução de *software*: garantia de que as mudanças do cliente serão atendidas.

Além da complexidade intrínseca de cada etapa, existem atividades como documentação e gerenciamento de configuração que dão apoio ao processo como um todo.

Ainda de acordo com Sommerville (2013), não existe processo único que se adeque a todo tipo de desenvolvimento de *software*. É de responsabilidade da organização criar e definir processos de desenvolvimento que se adequem às suas necessidades, de forma a tirar melhor proveito das habilidades de seus colaboradores. Ainda assim, é possível categorizar processos de *software*. Uma vez que, no processo, todas as atividades são planejadas com antecedência e comparado com o planejamento inicial, este processo é categorizado como dirigido a planos. Se o planejamento é gradativo, tornando mais fácil as necessidades de mudanças do cliente, ele é caracterizado como ágil.

3.1.1. Especificação de *software*

Também denominado Engenharia de Requisitos, é o processo da compreensão e definição das restrições e funcionalidades do sistema. É uma fase extremamente crítica (SOMMERVILLE, 2013) do processo de *software*, pois, problemas gerados por

incompreensão das funcionalidades podem se estender até o final do projeto, levando a um produto que não corresponda às expectativas.

Esse processo tem como objetivo produzir um documento de requisitos que especifica um sistema que satisfaz os requisitos do cliente.

3.1.2. Projeto e implementação de *software*

É a fase de tradução de requisitos em código. Sempre envolve projeto e programação. O projeto, é a fase de descrição da estrutura do *software* a ser implementado, dos modelos, interfaces e estruturas de dados usados pelo sistema. As atividades pertinentes a essa fase, a depender do tipo de sistema, podem variar. São exemplos de atividades de projeto:

- a) projeto de arquitetura: identifica a estrutura geral do sistema, módulos e relacionamentos;
- b) projeto de interface: definir interfaces entre os componentes do sistema;
- c) projeto de componente: define o funcionamento de cada componente do sistema;
- d) projeto de banco de dados: projeta as estruturas de dados do sistema e como são representados em um banco de dados.

Ao final da fase de projeto, são geradas as saídas pertinentes. A **Figura 1** ilustra as entradas, tomando como base o exemplo anterior, e saídas do processo.

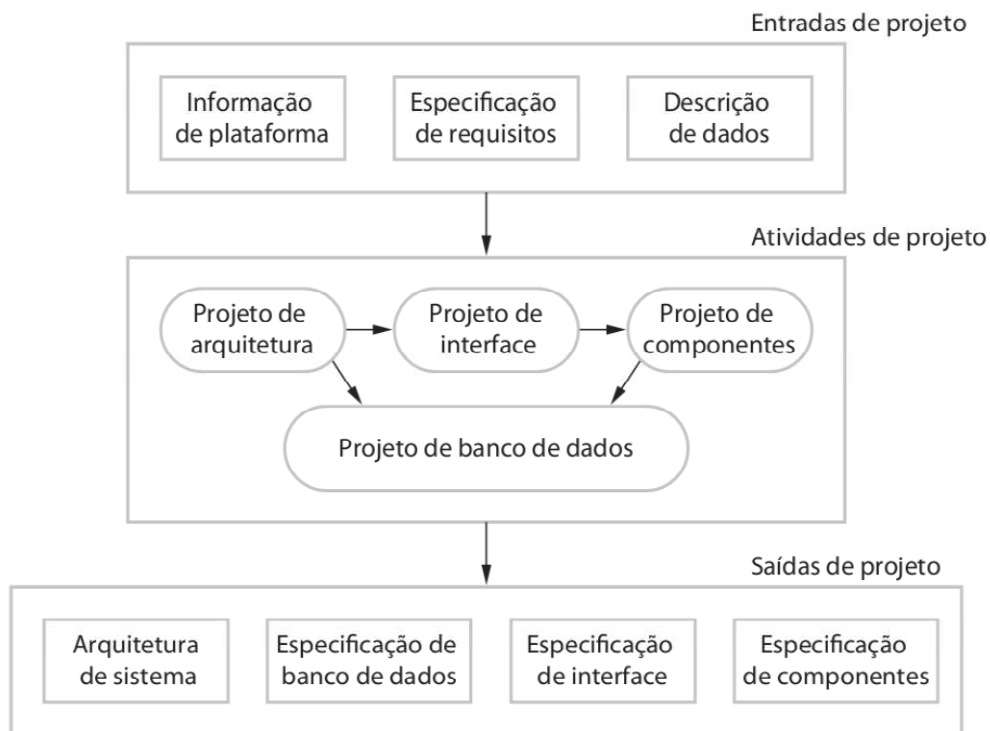


Figura 1: Modelo de processo de projeto (SOMMERVILLE, 2013)

Na fase de programação, não existe um processo a ser seguido. É papel da equipe escolher uma abordagem para o desenvolvimento.

3.1.3. Validação de *software*

Tem como objetivo, mostrar que um *software* se adequa às especificações bem como às necessidades do cliente. A principal técnica de validação é o teste com dados simulados. O processo de teste se dá em três etapas, conforme a **Figura 2**:

- a) teste de componente: cada componente do sistema é testado de forma individual, onde são avaliadas as suas funcionalidades e saídas esperadas;
- b) teste de sistema: todos os componentes são integrados para criar o sistema onde são testadas as interações entre esses componentes;
- c) teste de aceitação: é a etapa final, na qual, antes da entrega final, o sistema é testado com dados fornecidos pelo cliente, dados reais.

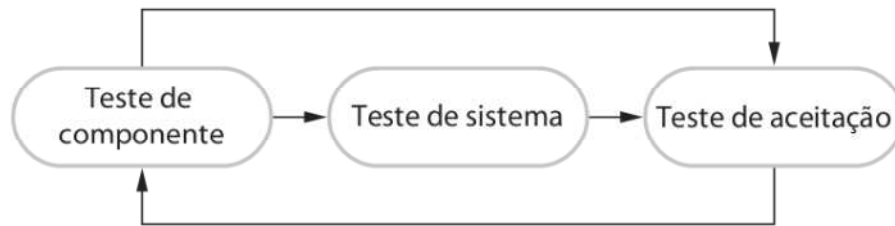


Figura 2: Fluxo da etapa de testes (SOMMERVILLE, 2013)

3.1.4. Evolução de *software*

Mudanças no *software* podem ser feitas a qualquer momento, durante ou após o desenvolvimento do sistema. A manutenção é o processo contínuo que se dá após o desenvolvimento do *software*, são alterações que visam adequar o produto às novas expectativas do modelo de negócio.

O *software* deve poder se adequar a mudanças ao mesmo tempo que o negócio, que o originaram, respondendo de acordo com prioridades gerenciais ou de natureza externa. (SOMMERVILLE, 2013)

3.2. DESENVOLVIMENTO ÁGIL DE *SOFTWARE*

Operando em um ambiente global, com mudanças rápidas, as empresas precisam se adaptar a novos mercados e oportunidades. Por fazerem parte de quase todas as operações de um negócio, os produtos de *software* necessitam, tão rápido quanto necessário, se adaptarem a essas mudanças.

Neste sentido as entregas e o desenvolvimento rápidos são os requisitos mais críticos a serem trabalhados durante toda a execução da fase de desenvolvimento (PRIKLADNICKI et al., 2014). Por via de regra, o desenvolvimento ágil, possibilita o desenvolvimento e entregas rápidas de produtos de *software*, nesse contexto, os produtos de *software* são produzidos não como um uma unidade única, mas sim como um conjunto de vários incrementos, os quais cada incremento representa uma funcionalidade do sistema (SOMMERVILLE, 2013).

Assim como o processo de desenvolvimento dirigido a planos, o processo ágil não é único, podendo se adaptar-se de acordo com a necessidade do sistema, da empresa de

desenvolvimento ou do cliente, no entanto, essas abordagens compartilham algumas características (SOMMERVILLE, 2013):

- a) processos de especificação, projeto e implementação são intercalados. A geração de documentos é mínima; o documento de requisitos possui apenas características fundamentais do sistema; a documentação pode ser gerada automaticamente pelo ambiente de programação adotado para o desenvolvimento do sistema;
- b) o sistema é desenvolvido em uma série de versões. O próprio usuário ou *stakeholders* são responsáveis pela avaliação do sistema onde, quaisquer mudanças propostas serão implementadas já na próxima versão ou entrega;
- c) as interfaces de usuário são desenvolvidas, geralmente, através de um processo rápido de posicionamento de ícones na interface, com um sistema interativo de desenvolvimento, provendo assim, interfaces multiplataforma.

Os métodos ágeis proveem uma entrega incremental de forma a, devido à imersão do cliente no ambiente de desenvolvimento, se obter um *feedback* rápido sobre a evolução das funcionalidades do sistema.

3.3.GERENCIAMENTO ÁGIL DE PROJETOS

Em uma abordagem dirigida a planos, o gerente de projetos deverá ter uma visão estável de todo o projeto bem como dos processos de desenvolvimento (SUTHERLAND, 2014). Visando um melhor uso de tempo e equipe, os métodos ágeis requerem um tipo de gerenciamento que segue uma abordagem adaptada para o desenvolvimento incremental.

Segundo Sommerville (2013), a abordagem *Scrum* é um método ágil onde o seu foco está no gerenciamento do desenvolvimento incremental. É dividido em três fases:

- a) planejamento geral, no qual são estabelecidos os objetivos gerais e da arquitetura do sistema;
- b) uma série de ciclos de *sprints*, onde a cada ciclo uma nova funcionalidade ou melhoria é implementada;
- c) a última fase é a entrega final do projeto, com toda a documentação exigida e uma avaliação acerca das lições aprendidas com o projeto é desenvolvida.

A principal característica do *Scrum* são os ciclos de *sprint*. Um *sprint* é um planejamento onde aquilo que deverá ser desenvolvido é avaliado, os recursos necessários são alocados e o incremento é implementado (PRIKLADNICKI et al., 2014). Ao término, uma funcionalidade é entregue aos principais interessados no sistema (*stakeholders*). São características dos *sprints*:

- a) toda *sprint* possui tamanho fixo;
- b) o *backlog* do produto – lista de todo o trabalho a ser desenvolvido no decorrer do projeto - é o ponto de partida para o planejamento. O cliente atua diretamente nesta etapa, podendo, no início do *sprint*, incluir, priorizar ou remover requisitos ou tarefas;
- c) todos os integrantes da equipe estão envolvidos na fase de seleção da funcionalidade a ser desenvolvida no *sprint*;
- d) uma vez que todos estejam de acordo com o que vai ser desenvolvido no *sprint*, a equipe inicia o desenvolvimento. Através de reuniões diárias e curtas, envolvendo toda a equipe, o processo é analisado e, se preciso, mudanças são realizadas. Neste momento, toda a equipe está isolada do cliente, sendo possível a comunicação apenas por meio do líder da equipe, denominado Mestre *Scrum* (*Scrum Master*) que, por sua vez, além de garantir que distrações externas afetem a equipe, é responsável por garantir que o processo seja eficaz (SUTHERLAND, 2014);
- e) ao final do *sprint*, a funcionalidade é entregue aos *stakeholders* e um novo *sprint* é iniciado em seguida. A **Figura 3** ilustra o processo de *Scrum*.

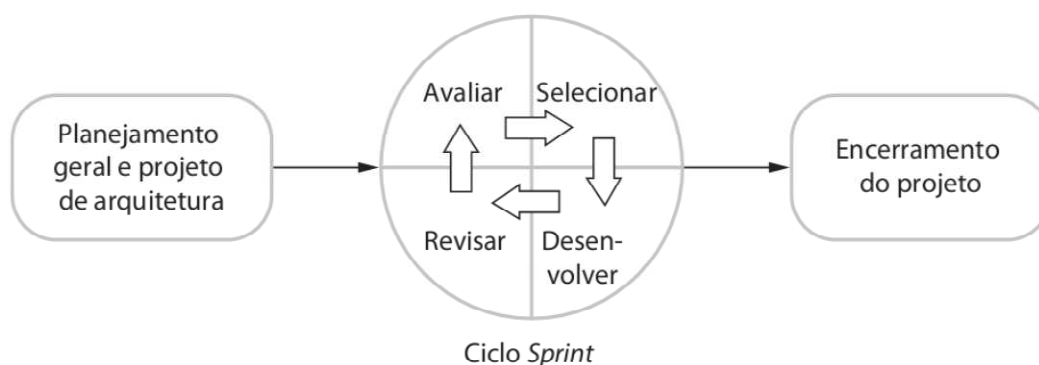


Figura 3: Processo de Scrum (SOMMERVILLE, 2013)

3.4.PADRÕES DE PROJETO E ARQUITETURA

A utilização de padrões de projetos, tem como principal objetivo prover a reutilização de uma prática e/ou arquitetura, bem-sucedidas, em outros projetos. (ERICH GAMMA, et al., 2000), conceituam que cada padrão de projeto descreve um problema e a essência da solução, de tal forma que possa ser utilizada quantas vezes forem necessárias.

3.4.1. Padrão de arquitetura Modelo-Visão-Controlador (MVC)

Esse padrão de arquitetura estrutura o sistema em três componentes lógicos que interagem entre si. O Modelo atua sobre o sistema de dados e as operações associadas a esses dados. O componente Visão define e gerencia a forma como os dados serão apresentados para o usuário. O Controlador passa as informações de interação do usuário para os componentes Visão e Modelo. A **Figura 4** mostra o fluxo de comunicação quando o padrão é aplicado a aplicações Web.

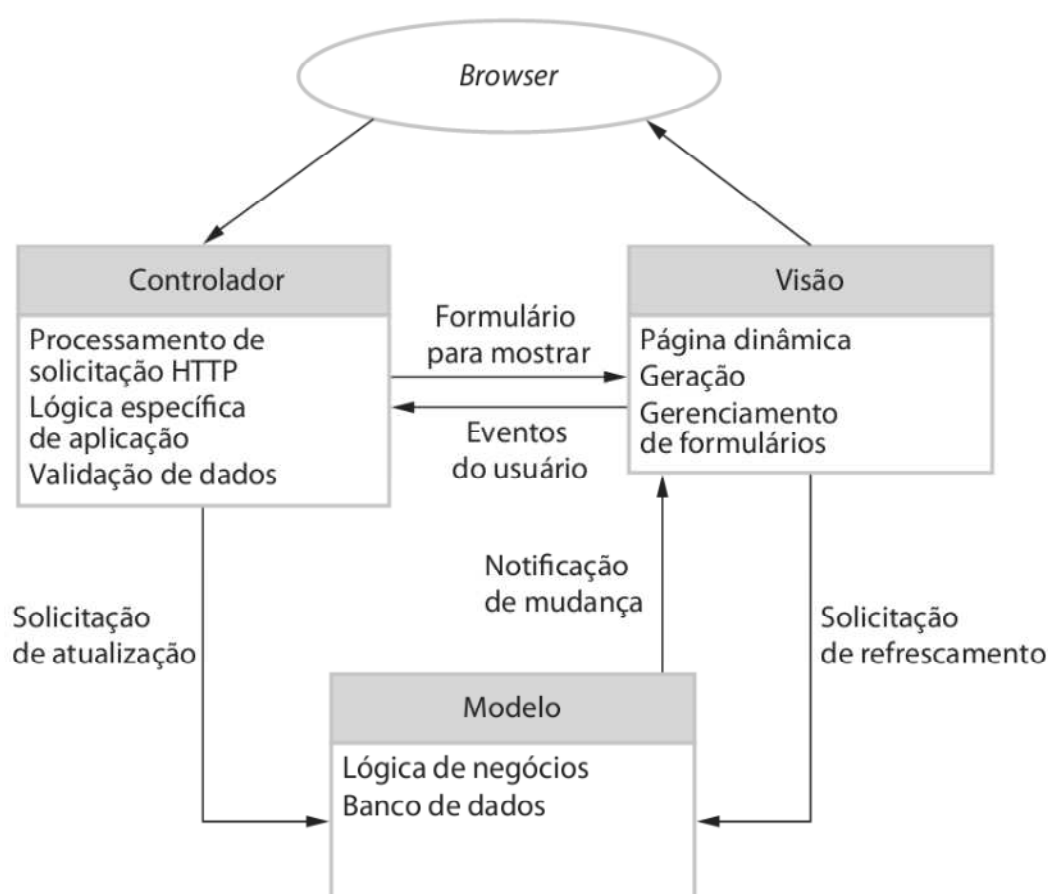


Figura 4: Arquitetura MVC aplicada a aplicações Web (SOMMERVILLE, 2013)

3.4.2. Padrão de Projeto *Data-Access-Object* (DAO)

Aplicações que usam algum tipo de persistência de dados, e.g. Banco de Dados (BD), necessitam, de alguma forma, manipular esses dados. Dessa forma, trechos de códigos escritos na linguagem de manipulação do BD serão adicionados em várias partes distintas do sistema, tornando a manutenção do código cada vez mais complexa.

O padrão DAO soluciona esse problema de forma que, todas as comunicações com mecanismos de persistência serão mediadas por meio de um objeto DAO, o qual mapeará informações, transportadas em objetos, para instruções da API de persistência e os resultados mapeados de volta, para os mesmos objetos de transporte.

Para facilitar o reuso de código e possibilitar menor acoplamento, é importante que o DAO seja especificado por meio de uma interface. A **Figura 5** representa a implementação deste padrão.

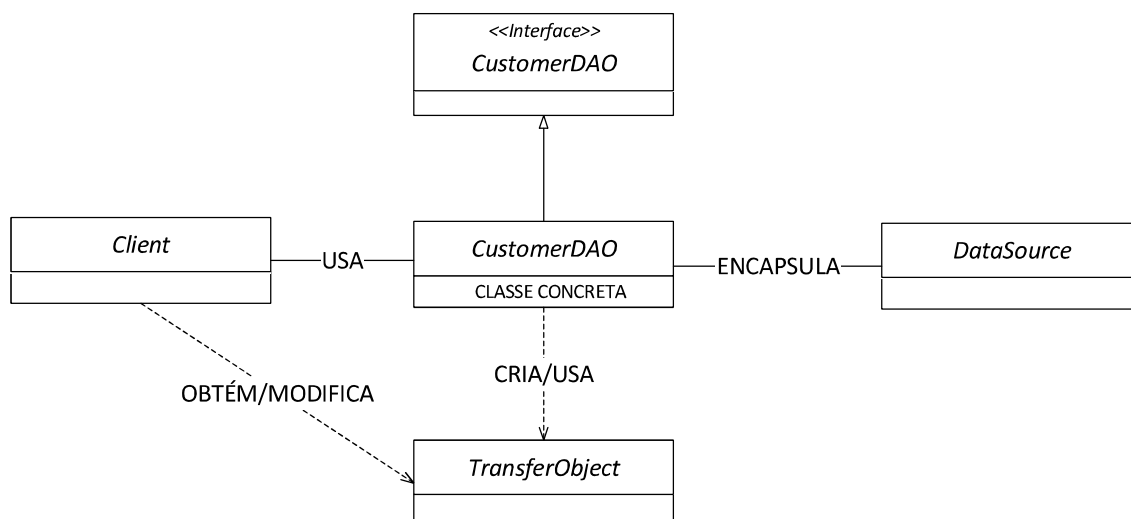


Figura 5: Representação do padrão DAO (Adaptado de MACORATTI)

3.5.APLICAÇÃO WEB

Segundo Sebesta (2012), pode-se definir uma aplicação Web como sendo uma aplicação distribuída, que utiliza uma arquitetura cliente-servidor, a qual o lado servidor é capaz de receber requisições vindas do protocolo HTTP, tratá-las conforme algoritmo interno e retornar o resultado, para quem efetuou a requisição, por meio de uma resposta HTTP. Para exibição dos dados de resposta essas aplicações utilizam, primariamente, a linguagem de marcação HTML.

Dando ênfase à arquitetura do projeto, ainda de acordo com Sebesta (2012), o modelo arquitetural mais utilizado é o Multicamadas o qual a aplicação é dividida em

várias camadas, a depender do padrão de arquitetura utilizado, e cada camada tem um papel específico no âmbito da aplicação.

3.6. CASCADING STYLE SHEETS (CSS)

Utilizada para definir a aparência da apresentação, o CSS define como um documento, desenvolvido em uma linguagem de marcação – como XML, HTML, XHTML, entre outras – serão exibidos. A principal vantagem no seu uso é que é possível separar formato e conteúdo de um documento.

3.7.A LINGUAGEM JAVASCRIPT

É uma linguagem de script, utilizada no lado cliente da aplicação (*client-side language*). É utilizada para controlar o HTML, CSS e manipular comportamentos da página. É criada e mantida pela *European Computer Manufacturer's Association (ECMA)*. Dentre as principais vantagens destacam-se: (FLANAGAN, 2012)

- a) é Orientada a Objetos;
- b) possui bibliotecas como o JQUERY e o *framework* Prototype que facilitam o uso da linguagem.

3.7.1. JQUERY

É uma biblioteca JavaScript, bastante rica em funcionalidades que permitem, com poucas linhas, realizar diversos efeitos, que custariam mais horas de trabalho se programados em JavaScript puro.

Dentre outros, alguns recursos dessa biblioteca destacam-se:

- a) seleção e manipulação de elementos HTML;
- b) manipulação de CSS;
- c) efeitos e animações;
- d) navegação pelo DOM;
- e) eventos.

3.8. A LINGUAGEM PHP

É uma linguagem de script, de código aberto, amplamente utilizada para desenvolvimento Web. É uma linguagem utilizada no lado servidor da aplicação (*server-side language*). Dentre as características, destacam-se:

- a) é uma linguagem interpretada;
- b) orientada a Objetos;
- c) gratuita e de código aberto;
- d) provê suporte a um grande número de base de dados e.g. Oracle, Sybase, PostgreSQL, MySQL, entre outras;

No entanto, por poder ser embutida dentro do HTML, pode gerar aplicações difíceis de manter e fortemente acopladas, caso não associadas a boas práticas de desenvolvimento de *software*.

4. ESPECIFICAÇÕES DO SIGNUTES

O sistema possui quatro módulos: Acesso, Colaboradores, Patrimônio e Mobile, além de uma interface, de usuário, padronizada para integrar todos os módulos. De acordo com um levantamento inicial, o **Diagrama 1** foi elaborado como forma de documentar a visão geral do sistema a ser desenvolvido.

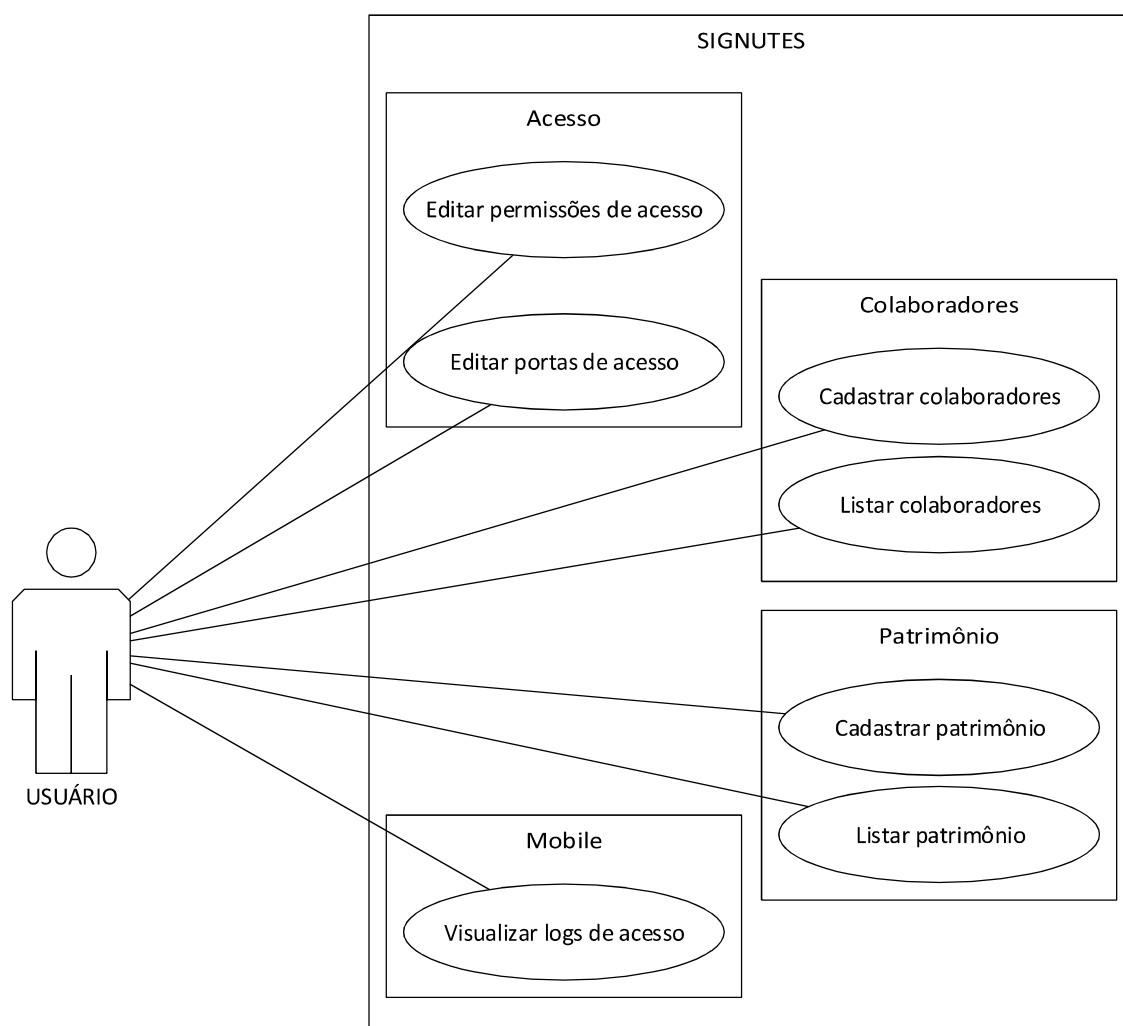


Diagrama 1: Casos de uso do sistema SIGNUTES

5. PROCESSOS ASSOCIADOS AO DESENVOLVIMENTO DO SIGNUTES

5.1. PROCESSO DE SOFTWARE

Diante das necessidades iniciais levantadas acerca do SIGNUTES, observou-se que seria necessário utilizar um processo que gerasse entregas rápidas, pouca

documentação inicial e que permitisse fácil alteração e manutenção de requisitos. Neste contexto, utilizou-se um processo ágil de desenvolvimento, adaptado a tais necessidades, e o *Scrum* – também adaptado - como método de gerenciamento.

5.2.GERENCIAMENTO DE PROCESSO

O *Scrum*, utilizado no gerenciamento do processo de desenvolvimento do SIGNUTES, utilizava a seguinte abordagem:

- a) o planejamento era alterado à cada mudança ou adição de requisitos;
- b) *sprints* de quatro dias, implicando em entregas pequenas e rápidas;
- c) as entregas eram enviadas diretamente para o servidor *Web*;
- d) os testes eram apenas unitários;
- e) problemas relacionados à integração eram retornados como *bugs*;
- f) os *bugs* tinham prioridade em relação ao que estava sendo produzido;
- g) os *stakeholders* atuavam diretamente no desenvolvimento do sistema, interagindo com o desenvolvedor.

5.3.PADRÃO DE ARQUITETURA

Visando a um sistema que fosse de fácil manutenibilidade e buscando estabelecer um modelo padrão para que todos os módulos pudessem ser desenvolvidos, o modelo MVC foi escolhido como base da arquitetura padrão do sistema. O **Diagrama 2** ilustra a arquitetura desenvolvida e a **Figura 6** exemplifica a estrutura física de cada módulo, bem como as linguagens de desenvolvimento predominantes em cada um deles.

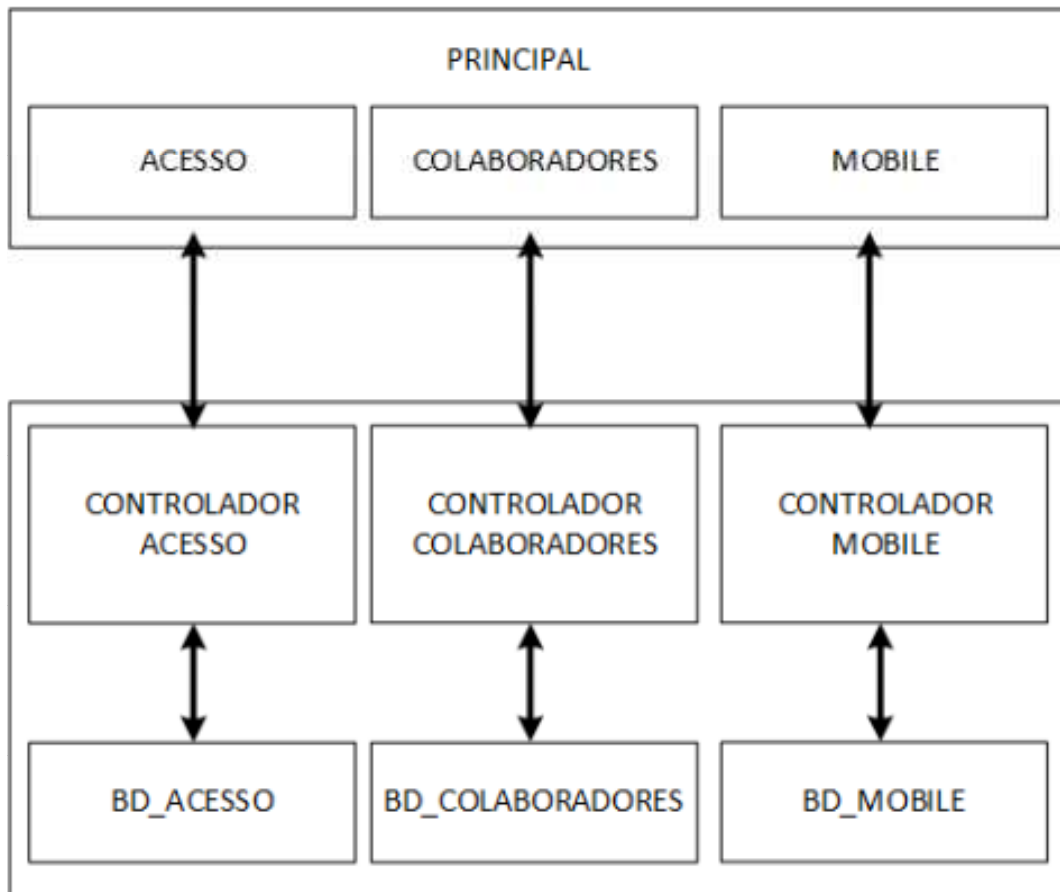


Diagrama 2: Arquitetura do SIGNUTES

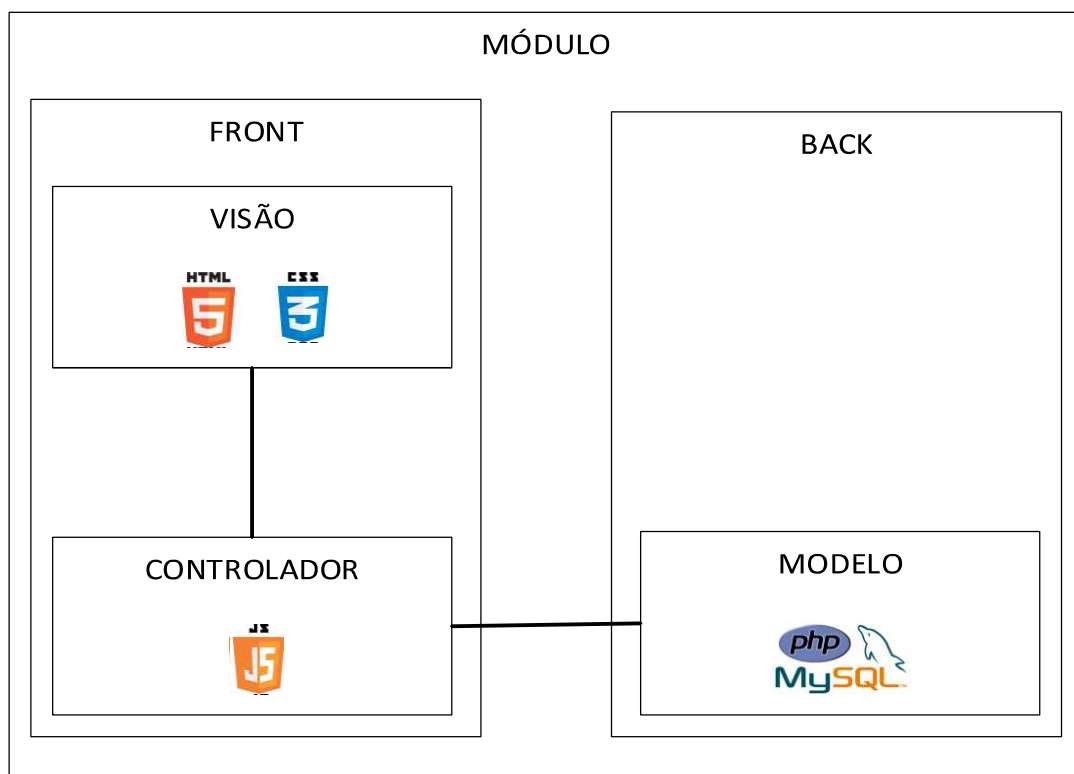


Figura 6: Padrão MVC aplicado para cada módulo do SIGNUTES

6. REQUISITOS DO SISTEMA

As duas entregas do sistema foram desenvolvidas, levando-se em consideração os cenários 1 e 2, conforme os itens 6.1 e 6.2, a seguir.

6.1.CENÁRIO 1

O usuário, com privilégio, cadastra um aluno, funcionário ou pesquisador do NUTES; após o preenchimento de todos os campos obrigatórios o usuário pode somente salvar as informações ou gerar e imprimir o Termo de Confidencialidade e Sigilo bem como a Declaração de Possíveis Conflitos de Interesse. É obrigatório capturar uma foto da pessoa no ato do cadastro. Em um outro momento, o usuário poderá imprimir os termos de qualquer pessoa cadastrada no sistema ou editar os dados cadastrais da mesma.

6.2.CENÁRIO 2

O SIGNUTES comunica-se com o sistema interno de controle de portas e é possível cadastrar as pessoas para que, por meio de um sistema *Mobile*, elas tenham acesso ao NUTES, dadas suas devidas permissões.

O usuário, com privilégio edita e/ou define as permissões de acesso das pessoas, a depender do privilégio de cada um. O **Quadro 1**, exemplifica os níveis de permissões para cada colaborador cadastrado no sistema.

NOME	FUNÇÃO	NÍVEL DE ACESSO
Colaborador A	Professor	1,1,1,1,1
Colaborador B	Aluno	1,0,0,1,0
Colaborador C	Funcionário	1,1,1,1,1
Colaborador D	Aluno	1,0,0,0,0

Quadro 1: *Nível de acesso de Colaboradores.*

O Colaborador A é um professor e, por via de regra, acessa todas as portas, no qual, o nível de acesso 1 representa acesso permitido e o nível 0 acesso não permitido; o Colaborador B é um aluno e acessa apenas o laboratório sob supervisão do seu professor;

o Colaborador C é um funcionário que possui acesso a todas as portas, e.g. Funcionário da Limpeza; o Colaborador D é um aluno recém cadastrado no sistema.

6.3.REQUISITOS ENTREGA 1

A partir do Cenário 1, representado pelo **Diagrama 3**, foi levantado o *backlog*, da primeira entrega do sistema, de acordo com o **Quadro 2**.

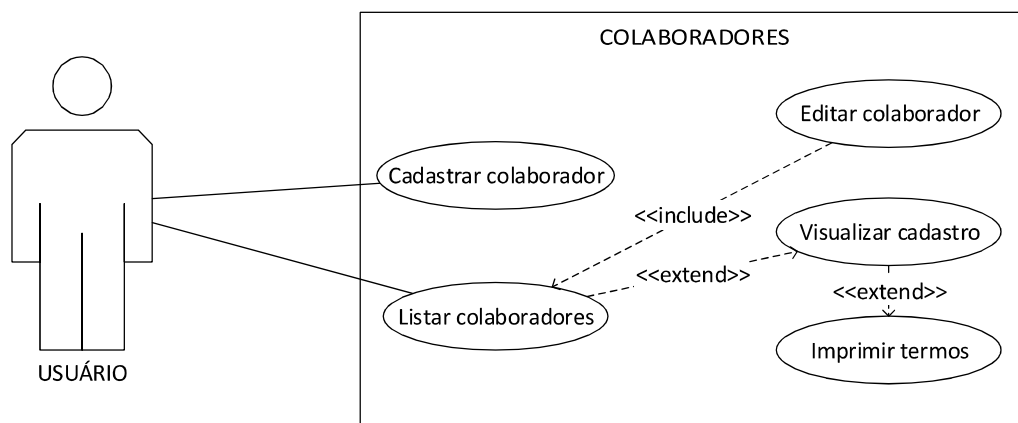


Diagrama 3: Casos de uso Cenário 1.

Casos de Uso	Tarefas
Cadastrar colaborador	Implementar visão do cadastro
	Gerar feedback na obrigatoriedade dos campos
	Criar janela modal da foto
	Garantir que a foto seja obrigatória
	Implementar persistência do cadastro
	Gerar termos da pessoa cadastrada
Listar colaboradores	Gerar lista dinâmica com todos os cadastros do banco de dados
Editar colaborador	Alterar foto usando janela modal
	Implementar alteração de cadastro
	Garantir obrigatoriedade da foto na alteração do cadastro
Visualizar cadastro	Implementar remoção de cadastro
	Implementar geração de termos

Quadro 2: Backlog Cenário 1.

6.4.REQUISITOS ENTREGA 2

A partir do Cenário 2, modelado no **Diagrama 4**, foi possível levantar o *backlog*, da segunda entrega do sistema, de acordo com o **Quadro 3**.

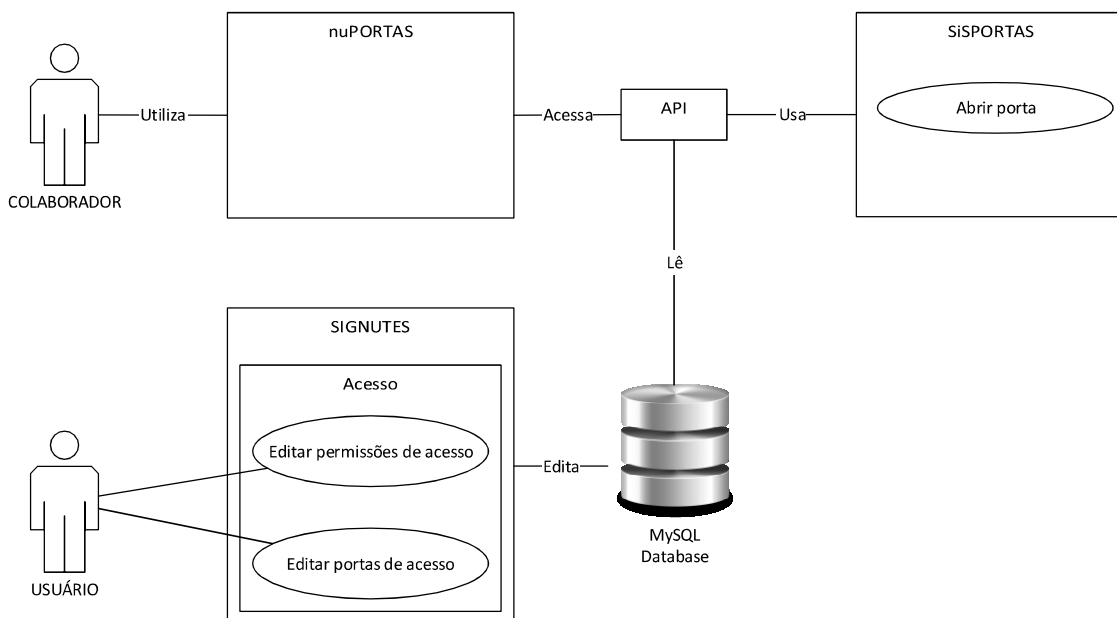


Diagrama 4: Casos de uso Cenário 2.

Casos de Uso	Tarefas
Editar permissões de acesso	Implementar visão do gerenciamento de permissões de colaboradores
	Garantir que todo colaborador cadastrado tenha um nível de acesso padrão
	Fazer comunicação com a API
	Implementar feedback de cadastro
	Implementar persistência do cadastro
Editar portas de acesso	Implementar visão do gerenciamento de portas
	Implementar edição de nome de portas
	Implementar edição de código de abertura de porta

Quadro 3: Backlog Cenário 2.

7. DESENVOLVIMENTO DO SISTEMA

Durante o processo de desenvolvimento do SIGNUTES, várias ferramentas foram utilizadas, o item 7.1, a seguir, descreve-as, apontando vantagens e desvantagens encontradas durante o processo.

7.1.FERRAMENTAS UTILIZADAS NO PROCESSO DE DESENVOLVIMENTO

O **Quadro 4**, descreve as ferramentas utilizadas durante o desenvolvimento do SIGNUTES, bem como as principais vantagens e desvantagens associadas ao uso, encontradas durante o processo.

Ferramenta	Descrição	Vantagens	Desvantagens
Bootstrap Studio	É uma ferramenta proprietária. Faz uso do <i>framework Bootstrap</i> para desenvolver páginas <i>Web</i> .	Possibilita trabalhar com folhas de estilo CSS e também com <i>Javascript</i> .	Limita-se a trabalhar no <i>front-end</i> .
		Possui pré-visualização automática da página (se habilitado).	Gera, por padrão, apenas uma folha de estilo para todo o projeto.
		Equipe de desenvolvimento muito ativa, possibilitando atualizações constantes da ferramenta.	

Visual Studio Code	Ferramenta livre, de código aberto, desenvolvida pela Microsoft®.	Possui um amplo acervo de plugins e extensões, possibilitando a integração com diversas linguagens de programação bem como, com a ferramenta de controle de versão, GIT.	Não foi constatada nenhuma desvantagem no uso da ferramenta, durante o processo.
		Possui o recurso de complemento automático para a linguagem PHP (utilizada no SIGNUTES), aumentando a velocidade na escrita de códigos.	
		Baixo consumo de recursos de máquina.	
EasyPHP	Permite a criação de um servidor em máquina local, para desenvolvimento Web.	Dá suporte a PHP, MySQL e Apache	A mudança do local de hospedagem, requer muito esforço.
		Interface interativa de configuração e utilização.	

Quadro 4: Ferramentas utilizadas durante o desenvolvimento do SIGNUTES

7.2.DESENVOLVIMENTO DA PRIMEIRA ENTREGA

Além dos casos de uso descritos no **Quadro 2**, o desenvolvimento da primeira entrega do sistema consistiu em:

- a) Definição do guia de estilo a ser utilizado;
- b) Implementação do *layout* padrão do sistema, conforme equipe de *design*.

Os itens 7.2.1, 7.2.2 e 7.2.3 apresentam as telas que foram implementadas e que representam o *layout* padrão do SIGNUTES.

7.2.1. Tela de autenticação

Conforme ilustração da **Erro! Fonte de referência não encontrada.**, a tela de autenticação foi desenvolvida. Uma pré-validação dos campos Usuário e Senha foi implementada neste nível, utilizando HTML5.

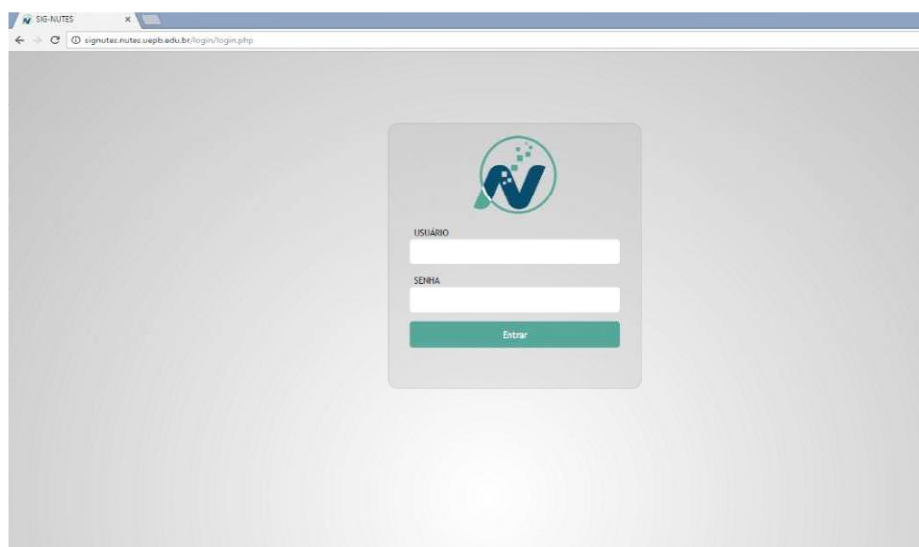


Figura 7: Tela de autenticação

7.2.2. Tela principal do sistema

Segundo a ilustração da **Figura 8**, a tela principal, apresenta um menu lateral com todas as funcionalidades do sistema e um botão “Sair”, do lado superior direito, o qual refere-se à sessão do usuário autenticado.

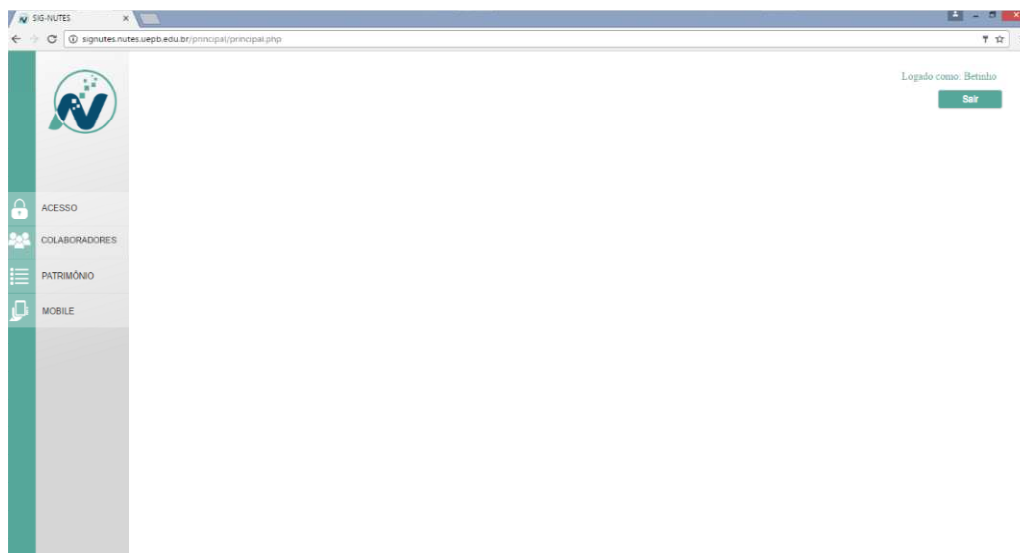


Figura 8: Tela principal do sistema.

7.2.3. Menu lateral do sistema

A **Figura 9** ilustra o menu lateral do sistema. Ao selecionar um módulo do sistema, um submenu é apresentado, com as devidas opções, associadas ao módulo.

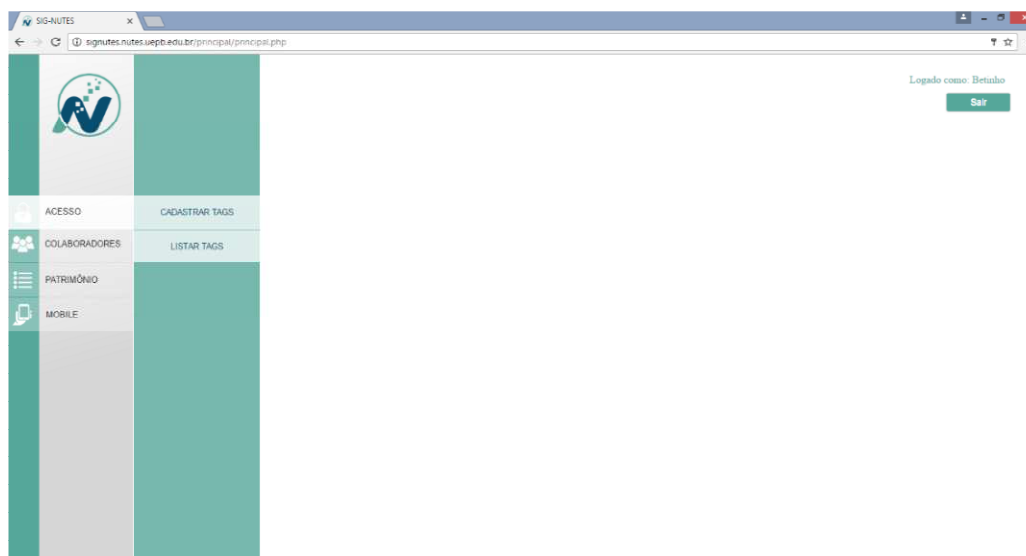


Figura 9: Menu lateral do sistema

7.2.4. Tela do cadastro de colaboradores

A partir do *layout* implementado, o cadastro de colaboradores foi desenvolvido de acordo com o modelo de *design*, ilustrado pela **Figura 10**.

7.2.5. Tela de listar colaboradores

Conforme **Figura 11**, a listagem de colaboradores, desenvolvida. Possui um campo de pesquisa e uma tabela com as colunas Nome, Função, Celular, Email e Ações. Na coluna Ações, têm-se, consecutivamente, as opções Editar, Gerar Termos e Excluir.

Figura 10: Visão do cadastro de colaboradores.

Nome	Função	Celular	Email	Ações
Adalberto Francisco Monteiro Neto	aluno	(83) 99304-8290	betinho_fm@gmail.com	[Edit] [Generate Terms] [Delete]
ADRIELE BARBOSA DA COSTA	aluno	(83) 99103-3269	adrielebarbosacc@hotmail.com	[Edit] [Generate Terms] [Delete]
Alcinael Fernandes Pereira	aluno	(83) 99965-8122	alcinael@hotmail.com	[Edit] [Generate Terms] [Delete]
ANDRÉ ALMEIDA	aluno	(83) 99665-1042	andre_almd@gmail.com	[Edit] [Generate Terms] [Delete]
ANTONIO MARRINHO DE ARAUJO NETO	aluno	(83) 99958-4841	antonio.aman@hotmail.com	[Edit] [Generate Terms] [Delete]
Breno Lacerda de Alustau Paiva	aluno	(83) 99846-0600	brenolacerda@gmail.com	[Edit] [Generate Terms] [Delete]
jessika azevedo soares	aluno	(83) 99914-1599	jessika.azvdo@gmail.com	[Edit] [Generate Terms] [Delete]
JOSE LAYRTON DE ALMEIDA RIBEIRO	aluno	(83) 90337-1107	123layrton@gmail.com	[Edit] [Generate Terms] [Delete]

Figura 11: Visão da tela listar colaboradores.

7.2.6. Padrão de *back-end* do sistema

Seguindo a arquitetura adotada para o desenvolvimento do sistema, todos os módulos do SIGNUTES, os quais possuam algum tipo de cadastro, devem apresentar uma classe que encapsule os dados, de acordo com o **Quadro 5**.

Código PHP

```

class myObject{
    private $var1 = null;
    private $var2 = null;
    public function __construct($var1=null, $var2=null){
        $this->var1 = $var1;
        $this->var2 = $var2;
    }
    public function getVar1(){
        return $this->var1;
    }
    public function setVar1($var1){
        $this->var1 = $var1;
    }
    public function getVar2(){
        return $this->var2;
    }
    public function setVar2($var2){
        $this->var2 = $var2;
    }
}

```

Quadro 5: Classe do objeto de cadastro.

Para a comunicação com o banco de dados, faz-se necessária a criação de uma classe, exemplificada com nome *myDAO*, que implemente a Interface *iDataBase* conforme o **Quadro 6**. Seus métodos são descritos no **Quadro 7**.

Código PHP

```

class myDAO implements iDataBase{
    protected $adapter;
    public function __construct(iDataBase $object){
        $this->adapter = $object;
    }
    public function setTableName($name){
        $this->adapter->setTableName($name);
    }
    public function connect(){
        $this->adapter->connect();
    }
    public function createTable($name, $columns){
        return $this->adapter->createTable($name,
        $columns);
    }
    public function insert($data){
        return $this->adapter->insert($data);
    }
}

```

```

    }
    public function update($data, $where){
        return $this->adapter->update($data, $where);
    }
    public function select($where){
        return $this->adapter->select($where);
    }
    public function delete($where){
        return $this->adapter->deleta($where);
    }
    public function close(){
        $this->adapter->close();
    }
}

```

Quadro 6: Implementação do DAO.

Método	Descrição
connect()	Efetua a conexão com o banco de dados utilizado
createTable(\$name, \$columns)	Cria, se não existir, a tabela no banco de dados. Recebe como parâmetros o nome da tabela e as colunas a serem criadas.
setTableName(\$name)	Seta qual a tabela a ser utilizada nesta conexão com o banco de dados. Recebe como parâmetro o nome da tabela.
insert(\$data)	Inserir um registro na tabela. Recebe como parâmetro o que será inserido.
update(\$data, \$where)	Atualiza um registro na tabela. Recebe como parâmetros o que será atualizado e onde será atualizado.
select(\$where)	Efetua uma pesquisa na tabela. Recebe como parâmetro o que será pesquisado.
delete(\$where)	Deleta algum registro da tabela. Recebe como parâmetro o que será deletado.
close()	Fecha a conexão com o banco de dados.

Quadro 7: Descrição dos métodos da Interface *iDataBase*.

7.2.7. Feedback da primeira entrega

Com um custo total de treze *sprints*, a primeira entrega foi realizada. O produto desenvolvido apresentou uma série de *bugs*, os quais foram solucionados sob demanda,

não apresentando rastreabilidade no processo de desenvolvimento em questão. Nenhuma alteração nos requisitos iniciais implicou em mudanças críticas no sistema.

7.3.DESENVOLVIMENTO DA SEGUNDA ENTREGA

Além dos casos de uso descritos no **Quadro 3**, para a segunda entrega, fez-se necessário efetuar mudanças na API de comunicação com o sistema das portas, bem como no aplicativo, antes desenvolvidos por outra equipe, utilizado para abertura das portas do NUTES, denominado nuPortas.

Os itens 7.3.1 e 7.3.2 ilustram as telas que foram implementadas, de acordo com o padrão já estipulado.

7.3.1. Tela de permissões de acesso

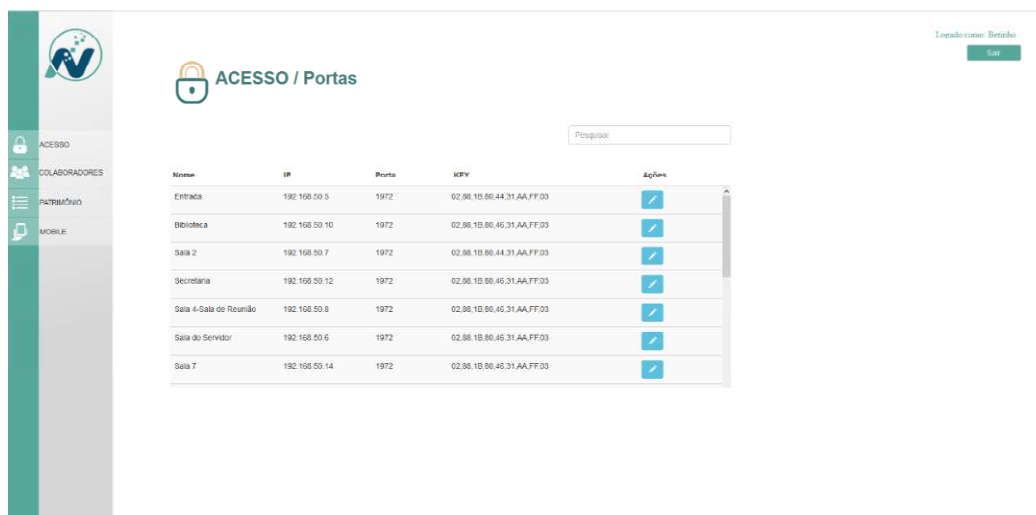
Conforme ilustrado na **Figura 12**, a tela de edição de permissões de acesso, possui uma listagem de todos os colaboradores cadastrados, apresentando suas respectivas TAGS (caso possuam), o número de identificação único, global do celular (IMEI) cadastrado para utilizar o aplicativo nuPortas e suas permissões de acesso. As ações associadas a cada linha são, respectivamente, Editar TAG, Editar IMEI e Editar Permissões.

Nome	TAG	IMEI	Permissões	Ações
Adalberto Francisco Monteiro Neto		259491055112915	1.1.1.1.1.1.1.1.1.1.1.1.1.0	[Editar TAG] [Editar IMEI] [Editar Permissões]
Adriano Araújo Felleberto		358997083416484	1.0.0.0.0.0.0.0.0.0.0.1.0.0.0	[Editar TAG] [Editar IMEI] [Editar Permissões]
Adrielle Barbosa Da Costa			1.0.0.0.0.0.0.0.0.0.0.0.0.0.0	[Editar TAG] [Editar IMEI] [Editar Permissões]
Adrielle Barbosa Da Costa		355461067190152	1.0.0.0.0.0.0.0.0.0.0.0.0.0.0	[Editar TAG] [Editar IMEI] [Editar Permissões]
Alicinei Fernandes Pereira			1.0.0.0.0.0.0.0.0.0.0.0.0.0.0	[Editar TAG] [Editar IMEI] [Editar Permissões]
Alexandre Augusto Nobrega Diniz			1.0.0.0.0.0.0.0.0.0.0.0.0.0.0	[Editar TAG] [Editar IMEI] [Editar Permissões]
Aline De Carvalho Mendes		35333402064007	1.1.0.0.0.0.0.0.0.0.0.0.0.0.0	[Editar TAG] [Editar IMEI] [Editar Permissões]

Figura 12: Tela de edição de permissões de acesso.

7.3.2. Tela de edição de portas de acesso

A **Figura 12** ilustra a tela de edição de portas, e possui uma listagem de todas as portas de acesso, previamente cadastradas em um banco de dados. É possível visualizar o nome da porta, o endereço IP, a porta de comunicação e o respectivo código chave para a abertura da mesma, *via* rede local. A ação associada a cada porta é o menu Editar porta, na qual é possível alterar qualquer informação listada sobre a mesma.










Nome	IP	Porta	MAC	Ações
Entrada	192.168.50.5	1972	02.88.1B.80.44.31AA.FF.03	
Biblioteca	192.168.50.10	1972	02.88.1B.80.46.31AA.FF.03	
Sala 2	192.168.50.7	1972	02.88.1B.80.44.31AA.FF.03	
Secretaria	192.168.50.12	1972	02.88.1B.80.46.31AA.FF.03	
Sala 4-Sala de Reunião	192.168.50.8	1972	02.88.1B.80.46.31AA.FF.03	
Sala do Servidor	192.168.50.6	1972	02.88.1B.80.46.31AA.FF.03	
Sala 7	192.168.50.14	1972	02.88.1B.80.46.31AA.FF.03	

Figura 13: Tela de edição de portas de acesso.

Conforme ilustrado no **Diagrama 4:** Casos de uso Cenário 2. O presente subsistema, comunica-se com um banco de dados para alterar seu conteúdo, logo segue modelo de *back-end* de acordo com os quadros **Quadro 6:** Implementação do DAO e **Quadro 7:** Descrição dos métodos da Interface iDataBase.

7.3.3. Feedback da segunda entrega

Com um custo de três *sprints*, a segunda entrega foi realizada. O subsistema não apresentou nenhum *bug*. Não houveram modificações nos requisitos iniciais.

8. CONSIDERAÇÕES FINAIS

Ao final de dezesseis *sprints*, duas entregas foram realizadas, implicando no desenvolvimento do SIGNUTES. Diante do Cenário 1 e Cenário 2, itens 6.1 e 6.2, respectivamente, o produto desenvolvido atendeu às expectativas. Os problemas, inicialmente levantados, tais como, cadastramento de colaboradores e controle de acesso, a geração automática do Termo de Confidencialidade e Sigilo e a Declaração de Possíveis Conflitos de Interesse, foram resolvidos com o uso do sistema.

A arquitetura padronizada permite que novos módulos sejam integrados ao sistema, seguindo o padrão definido pela equipe de desenvolvimento, alcançando-se com isso um melhor entendimento da estrutura do projeto e agilidade na adição de funcionalidades que possam surgir futuramente.

O processo utilizado, garantiu o desenvolvimento do projeto, no entanto, um ponto negativo é que o processo não viabilizou a etapa de testes, fazendo com que erros fossem encontrados durante o uso do sistema. Nesse sentido, problemas críticos poderiam ter posto em risco a entrega final do mesmo.

No tocante à experiência, foi dada a oportunidade de trabalhar com novas ferramentas e linguagens de programação, aumentando a experiência com desenvolvimento *Web*.

Espera-se que novas funcionalidades sejam adicionadas ao sistema, de acordo com a necessidade, incrementando-o, e que a arquitetura atual possibilite maior escalabilidade do mesmo.

REFERÊNCIAS

FLANAGAN, David. JavaScript: The definitive guide. 6. Ed. Bookman. 2012.

GAMMA, Erich; HELM, Richard; JOHNSON, Ralph; et al. Design Patterns. 1. Ed. Bookman. 2000.

MACORATTI. Apresentando o padrão DAO. Disponível em: <
http://www.macoratti.net/11/10/pp_dao1.htm>. Acessado em: Setembro de 2017.

PIONTKEWICZ, Regiane; FREITAS, Maria do Carmo Duarte; BIZ, Alexandre Augusto. Benefícios fiscais para incentivo à inovação tecnológica no Brasil: informação para uso no processo de tomada de decisão em indústrias de grande porte. RACEF – Revista de Administração, Contabilidade e Economia da Fundace. v. 8, n. 2, p. 31-47, 2017.

PRIKLADNICKI, Rafael; WILLI, Renato; MILANI, Fabiano. Métodos ágeis para desenvolvimento de *software*. 1. Ed. Bookman, 2014.

SEBESTA, Robert W. Programming the world wide web. 7. Ed. Pearson. 2012.

SOMMERVILLE, Ian. Engenharia de Software. 9. Ed. Pearson. 2013.

SUTHERLAND, Jeff. The art of doing twice the work in half the time. Crown Business. 2014.

TENÓRIO, Marge; ARANTES, Guilherme Mello; D'ÁVILA, Ana Luiza Viana. Políticas de fomento à ciência, tecnologia e inovação em saúde no Brasil e o lugar da pesquisa clínica. Ciência & Saúde Coletiva, vol. 22, núm. 5, pp. 1441-1454, maio, 2017.