# Resultado de imagem para fafic cajazeiras

# FUNDAÇÃO DE ENSINO SUPERIOR DE CAJAZEIRAS - FESC

# FACULDADE DE FILOSOFIA, CIÊNCIAS E LETRAS DE CAJAZEIRAS –

# FAFIC

# CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

# SILAS MAIKE DA SILVA SOUSA

# SISTEMA GESTOR DE PROCESSOS INTERNOS – SGPI

# CAJAZEIRAS-PB

**2019**

**SILAS MAIKE DA SILVA SOUSA**

**SISTEMA GESTOR DE PROCESSOS INTERNOS – SGPI**

Trabalho de Conclusão de Curso apresentado à Faculdade de Filosofia, Ciências e Letras de Cajazeiras – FAFIC, como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação.

**Orientador (a):** Prof. Esp. Luciano Tavares de Luna

# CAJAZEIRAS-PB

**2019**

**SILAS MAIKE DA SILVA SOUSA**

**SISTEMA GESTOR DE PROCESSOS INTERNOS – SGPI**

Trabalho de Conclusão de Curso apresentado à Faculdade de Filosofia, Ciências e Letras de Cajazeiras – FAFIC, como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação.

**Aprovado em:** \_\_\_/\_\_\_/\_\_\_

**BANCA EXAMINADORA**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Dr(a).\_\_\_\_\_\_\_\_\_\_ (FAFIC)

ORIENTADOR(A)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Dr(a).\_\_\_\_\_\_\_\_\_\_ (FAFIC)

EXAMINADOR(A)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Dr(a).\_\_\_\_\_\_\_\_\_\_ (FAFIC)

EXAMINADOR(A)

**SISTEMA GESTOR DE PROCESSOS INTERNOS – SGPI**

Silas Maike da Silva Sousa

**RESUMO**

Esse artigo propõe um modelo de software com o intuito de auxiliar nas atividades exercidas na Secretaria Geral da Faculdade de Filosofia, Ciências e Letras de Cajazeiras (FAFIC), diante disso, foram elaboradas varias técnicas e ferramentas utilizadas ao longo da leitura demonstrando o desenvolvimento de todo o trabalho. O software procura automatizar processos que foram levantados ao longo da pesquisa buscando aperfeiçoar e melhorar o desempenho dessas atividades.

**Palavras-chave: Processos. Software. Automatizar**

**1 INTRODUÇÃO**

A evolução tecnológica é algo que sempre esteve na vida humana desde os primórdios até, e principalmente, nos dias atuais com o aumento significativo da interação humano-computador. A tecnologia é uma ferramenta que ajuda os seres humanos, transformando atividades mais complexas em atividades mais simples de serem realizadas, trazendo uma grande facilidade para quem a utiliza.

O uso de sistema da informação é indispensável e de grande auxílio para tarefas do dia a dia, pois, com sua utilização, atividades que demandam muito tempo e esforço para serem realizadas se tornarão menos trabalhosas e seriam feitas rapidamente.

Foi observado na Faculdade de Filosofia, Ciências e Letras de Cajazeiras - FAFIC no setor da Secretaria Geral (SG) que algumas das atividades são realizadas sem o uso de uma ferramenta de software específica nas atividades essenciais do dia a dia do funcionário. E que o computador encontrado é utilizado para controlar através de planilhas os trabalhos que são feitos manualmente.

Devido esse setor da Secretaria ser bastante requisitado e não ser “informatizado” observou-se a necessidade de automatização destas atividades, propiciando, futuramente, a comunicação entre os setores da instituição através de uma ferramenta de software.

O Software desenvolvido tem como o objetivo atender as necessidades do setor da Secretaria Geral da FAFIC, automatizando alguns processos que são feitos manualmente e armazenados em Banco de Dados dando mais segurança e organização nos trâmites processuais desse setor. Esse software tem o propósito de diminuir os problemas encontrados no setor, acelerando o tempo de busca dos documentos processados, além de facilitar a criação de novos documentos e gerar um controle e armazenamento desses dados.

Como falado anteriormente, os processos destinados à SG não possuem nenhum tipo de informatização, e a correlação entre os setores da FAFIC é imprescindível para o gerenciamento de tais processos, pois eles são responsáveis por enviar a solicitação da documentação ao setor para serem analisados e processados, e dependendo da quantidade de solicitações que serão feitas, os outros setores irão ter que aguardar até que toda a documentação seja processada.

Outro fato para ser destacado é que todos os processos citados anteriormente são feitos presencialmente, ou seja, o aluno que necessitar de alguma documentação, como: cancelamento de matrícula, regime domiciliar para atestado ou gestante e dispensa de disciplina terá que se deslocar até a faculdade para dar início ao processo. A ferramenta proposta disponibiliza ao solicitante do processo, acesso remoto, isto é, não será necessária a presença física do mesmo na instituição para protocolar e/ou acompanhar o andamento do processo.

Como todos esses documentos são feitos manualmente, torna-se difícil o gerenciamento e localização do documento anterior e do próximo já que essa “identificação” (numeração do processo) é necessária para compor a criação de um novo processo. Assim, com o aumento na quantidade dos processos, torna-se mais difícil fazer uma consulta ou criar um novo processo. Isso não é problema para o SGPI (Sistema Gestor de Processos Internos) já que o mesmo propõe gerenciar toda a parte de criação de processos, gerando um identificador automaticamente, como também retornar qualquer consulta que será feita sem a necessidade de conhecimento sobre outros documentos.

Outro ponto a ser destacado é o fato de que um documento físico tem mais chances de ser perdido com o tempo, devido a questões de durabilidade e de fatores externos já que ele é físico, diferentemente de dados salvos em um software que vão trazer integridade e controle desses dados.

Como a Secretaria trabalha com muitas informações vinda de outros setores, muitas dessas informações demandam certo tempo para serem analisadas e processadas devido à quantidade de pedidos, e isso gera um grande problema, já que outros setores da instituição dependem do encaminhamento desses processos. O software visa automatizar e gerenciar todos os processos trazendo mais agilidade, rapidez e segurança.

Assim, o objetivo deste artigo é propor uma ferramenta para informatizar o setor da Secretaria Geral da FAFIC com a finalidade de facilitar e agilizar os processos destinados a esse setor, com o intuito de deixar mais eficiente a criação e o pedido de um processo tanto para os funcionários quanto para os alunos. É objetivo dessa ferramenta de software trazer agilidade e praticidade no manuseio dessas informações, pois seus dados serão armazenados em um banco de dados com backups de segurança, onde seriam feitas consultas que deixariam as atividades dos funcionários mais práticas no quesito de consulta e controle de dados/informações.

Dessa forma, o artigo será apresentado da seguinte maneira: inicialmente vão ser mostradas no referencial teórico as técnicas de engenharia de software que são necessárias para a construção deste software seguindo das ferramentas e tecnologias dando uma breve explicação do porque utilizá-las.

2 REFERENCIAL TEÓRICO

Neste tópico será abordado um pouco da metodologia e ferramentas que são de grande importância para a construção de um software.

2.1 Engenharia de software

Durante a elaboração deste projeto foi aplicado conhecimentos de engenharia de software, disciplina que se encarrega da especificação, criação e desenvolvimento de software, garantindo que o produto final não contenha ambiguidades. Segundo o artigo de Cabral, Silva e Souza (2019), “a crise de software ocorre quando o sistema não satisfaz todos os envolvidos no projeto, sejam clientes, desenvolvedores e/ou usuários”. Por isso a engenharia de software é importante para a construção de um sistema, pois propicia o desenvolvimento de software planejado, evitando conflitos e problemas futuros.

Foram incorporadas ao projeto boas práticas da engenharia de requisitos, área da engenharia de software que se encarrega de toda a documentação de requisitos necessária para a construção de um software. Segundo (SILVA, 2019), a especificação de requisitos é o documento de trabalho que servirá como referências para as demais atividades da engenharia de software nele são mostradas as técnicas de como se deve fazer, quais as necessidades reais e quais restrições existem para que o software seja desenvolvido.

2.1.1 Metodologias ágeis de software

Todo processo de desenvolvimento de software precisa seguir um método ou metodologia para serem trabalhadas com o objetivo de facilitar a criação do software. Dentre as diversas abordagens existem as chamadas metodologias ágeis, que apresentam técnicas para iniciar um processo de desenvolvimento de software. Segundo Pressan (2007), as metodologias ágeis constituem parte da Engenharia de Software, a área de conhecimento voltada para a especificação, desenvolvimento e manutenção de sistemas de software.

2.1.2 *Scrum*

Uma das metodologias ágeis adotada foi o *scrum*, que Andrade, Oliveira e Barbosa (2019), “é um *framework*[[1]](#footnote-1)[[2]](#footnote-2) dentro do qual pode-se empregar diversos processos e técnicas para desenvolver produtos complexos.” O scrum funciona da seguinte forma: são definidos ciclos de atividades chamados de *sprints*, que deverão ser entregues até o prazo determinado e nesse tempo também acontecem reuniões para tratar o que foi trabalhado. Nos *sprints* são feitas reuniões com os *stakeholders*, que são todas as pessoas envolvidas no projeto, para definir o nível de prioridade das atividades, qual tipo de documentação que deverá ser empregada no projeto e qual funcionalidade deverá ser implementada e testada.

Para o desenvolvimento deste trabalho utilizaremos algumas instâncias do *Scrum*, de acordo com o contexto e a necessidade, como gerenciador do processo de desenvolvimento.

### 

2.1.2.1 Processo *scrum*

Segundo Larman (2003 *apud* SABABBAGH, 2013, p. 35), o *Scrum* é iterativo. O produto é desenvolvido em ciclos ou iterações sucessivas. Em cada um desses ciclos é gerado um incremento no produto, que se soma e modifica o que já se tem pronto até o momento. Cada ciclo, assim, é como se fosse um pequeno projeto, com todas as atividades necessárias para se gerar uma parte do produto estável e funcionando, ainda que incompleta e que não seja imediatamente entregue para seus usuários.

No *Scrum*, os ciclos são chamados de *Sprints* como falado anteriormente. Eles têm tamanho fixo e acontece um após o outro, sem intervalos, ou seja, toda a elaboração do projeto que utiliza *Scrum* acontece dentro dos *Sprints*. Em cada *Sprint* existem atividades a serem seguidas, então, dentro das limitações desse projeto, serão citadas apenas as práticas necessárias para sua elaboração, segue abaixo:

* Plano de *Release*, estabelece a meta da *Release*, as maiores prioridades do *Product Backlog,* os principais riscos, características gerais e funcionalidades que estarão contidas na *Release*, bem como inspecionar e fazer modificações nos *Sprints*;
* Reunião de *Sprint Planning,* onde se realiza o planejamento do *Sprint* (ciclo);
* Trabalho de desenvolvimento do produto, que inclui tudo que é necessário para se entregar uma parte do produto pronto ao final do *Sprint*;
* Reunião de *Sprint* *Review*, onde os resultados do trabalho realizado no *Sprint* são apresentados as partes interessadas, que oferecem seu *feedback*. Essas reuniões podem ser definidas com intervalos de 7 (sete), 15 (quinze) ou 30 (trinta) dias, de acordo com os participantes do projeto.

2.1.2.2 Papéis do *scrum*

Devido à grande quantidade de papéis disposta pelo *Scrum*, este trabalho irá se utilizar daqueles que se enquadram para o desenvolvimento do mesmo, já que o número de membros envolvidos neste projeto é bastante reduzido. Abaixo se descreve os papéis que serão utilizados da metodologia *Scrum*.

* *Product Owner* - responsável por definir, comunicar e manter a visão do produto ao longo do projeto. Também é de responsabilidade do *Product Owner* (PO)criar uma lista de prioridades a serem desenvolvidas durante cada ciclo do projeto, chamada de *Product Backlog.*
* *Scrum Master -* é encarregado da organização e manutenção do projeto, ou seja, é responsável por garantir que os impedimentos que comprometam a sequência do projeto sejam removidos. Também facilita as interações entre o PO e os desenvolvedores.
* *Desenvolvedor -* encarregado de, juntamente com o cliente, efetuar a elaboração do *Product Backlog,* além de produzir a documentação, definir arquitetura e tecnologias utilizadas no desenvolvimento do produto, realiza seu trabalho de acordo com as metas negociadas com o PO*.*

2.1.3 Metodologia XP

Os principais fundamentos do XP tiveram origem nas tradições do desenvolvimento em *Smalltalk* e datam de meados da década de 80, quando Kent Beck e Ward Cunningham trabalhavam na Tektronixs, Inc. Práticas, tais como, refatoração, programação em par, mudanças rápidas, *feedback* constante do cliente, desenvolvimento iterativo, testes automatizados, entre outras, são elementos centrais da cultura da comunidade *Smalltalk*. Olhando deste ponto de vista, o XP pode ser considerado o modo de agir do *Smalltalk* generalizado para outros ambientes (TELES, 2005, p.55).

Segundo Beck (2004) o XP baseia-se em valores e práticas, em que o objetivo principal desta metodologia é assegurar que o cliente receba o máximo de valor a cada dia de trabalho da equipe de desenvolvimento.

### 

2.1.3.1 Valores do XP

A *eXtreme Programming* estabelece a forma de desenvolvimento do processo que são conhecidas como valores do XP. Abaixo seguem descritos os valores utilizados neste trabalho.

**Comunicação:** o XP promove uma maior comunicação entre os membros da equipe, pois, não é limitada por procedimentos formais. Utiliza o melhor meio possível, que pode ser: uma conversa ou reunião informal, correio eletrônico, telefonema ou qualquer tipo de comunicação desde que seja de caráter ágil.

**Simplicidade:** o XP incentiva práticas que reduzam a complexidade do sistema; a solução adotada deve ser sempre a mais simples e que alcance os objetivos.

***Feedback*:** várias práticas do XP garantem um rápido *feedback* sobre várias etapas/partes do processo, como: qualidade do código (testes unitários), estado do desenvolvimento (*user* *stories* e integração contínua).

**Coragem:** algumas práticas do XP aumentam a confiança do programador e ajudam-no a ter coragem para melhorar o código a fim de torná-lo mais simples, e investir tempo no desenvolvimento de testes.

**Respeito:** a falta de respeito entre os participantes do projeto pode influenciar negativamente no seu desenvolvimento, já que é necessária a harmonia e bom relacionamento entre os membros da equipe.

### 

2.1.3.2 Práticas do XP

De acordo com Beck K. (2004), a metodologia XP envolve 12 (doze) práticas que orientam a equipe do projeto, a seguir serão descritas algumas dessas práticas utilizadas no desenvolvimento deste trabalho.

**Entregas Frequentes** – este procedimento serve para manter o cliente informado sobre a produção, como também, se o mesmo está de acordo com o que já foi implementado. O período de entregas pode ser definido entre o cliente e a equipe, com prazo semanal, quinzenal ou mensal.

**Projeto Simples** – o sistema deve ser feito de forma simples e objetiva, com isso, o projeto torna-se de mais fácil compreensão e não há perda de tempo com algo que seja complexo.

**Testes Constantes** – o XP adota dois tipos de teste: o teste unitário automatizado, aquele em que o programador executa sempre que escrever código, e o teste funcionais, que são usados junto ao cliente para a verificação do sistema como um todo.

**Refatoramento** – consiste em aplicar melhorias no código existente, tornando-o mais simples e melhor estruturado, sem afetar sua funcionalidade.

2.1.4 *Trello*

O *Trello* é uma ferramenta on-line e pode ser utilizada para o gerenciamento de atividades do processo de desenvolvimento, nessa ferramenta são definidos quadros com as tarefas que serão executadas, o nível de prioridade de cada tarefa e a data de início e fim. O *Trello* é muito útil para organizar tarefas e definir quais são as atividades que serão desenvolvidas de acordo com os *sprints* definidos. Essa ferramenta é online e gratuita, bastante simples de ser utilizada em qualquer projeto.

2.1.5 Diagrama UML

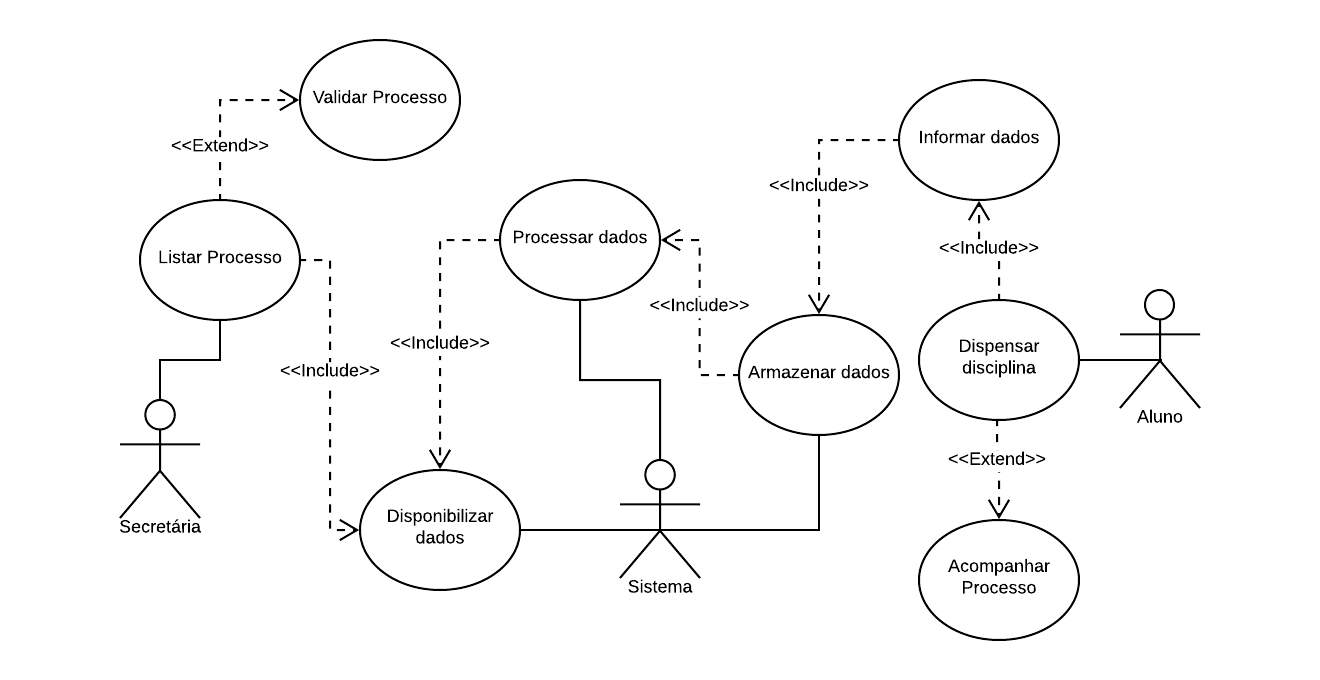
UML (Unified Modeling Language) ou Linguagem de Modelagem Unificada segundo Lucidchart (2019) é uma linguagem de modelagem unificada semântica e sintaticamente rica para arquitetura, design e implementação de sistemas de software complexos, tanto estruturalmente quanto para comportamentos. De modo geral, diagramas UML descrevem o limite, a estrutura e o comportamento do sistema e os objetos nele contidos.

O Diagrama UML faz parte da engenharia de requisitos e ela é responsável por demonstrar através de diagramas, modelos para serem compreendidos de forma fácil para pessoas de qualquer área de atuação.

2.1.6.1 Diagrama de caso de uso

Os diagramas de caso de uso têm a função de comunicar a funcionalidade e o comportamento do sistema para o usuário. Segundo Larman (2000, p.115), pode-se dizer que um caso de uso é um documento que descreve a sequência de eventos de um ator que usa o sistema para completar um processo. Os diagramas de caso de uso também demonstram como o usuário se comunica com o sistema. Como demonstrado na figura a seguir.

Figura 1 - Diagrama de Caso de Uso



Fonte: Elaboração do autor

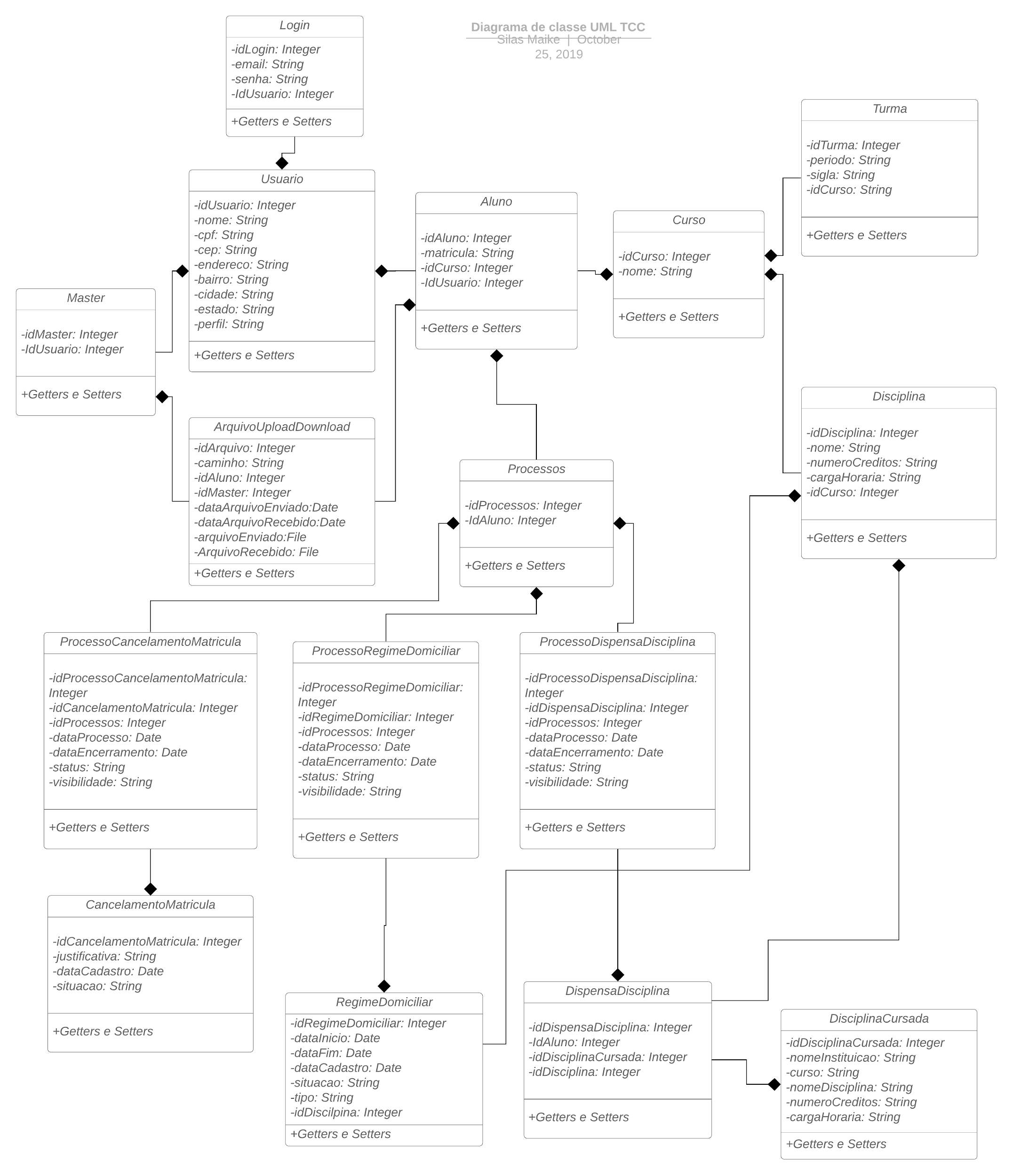
2.1.6.2 Diagrama de classe

Uma classe é uma representação de um objeto e nela estão definidos todos os seus atributos e funcionalidades.

O diagrama de classe é a representação das classes do software e a relação que essas classes possuem umas com as outras. Através do diagrama de classe é possível identificar as relações entre objetos do sistema facilitando o entendimento do código.

Existem diversas ferramentas que auxiliam na modelagem do sistema, esta pesquisa utilizou a ferramenta online L[ucidchart](https://www.lucidchart.com/), a mesma permite que os usuários criem desenhos, gráficos, diagramas, entre outros modelos para a elaboração de projetos. Por se tratar de uma ferramenta online e gratuita funciona em qualquer computador, não requer instalação, e todos os seus projetos ficam salvos em nuvem, promovendo mobilidade, desde que o dispositivo possua conexão com a internet (LUCIDCHART, 2019).

Figura 2 - Diagrama de Classes



Fonte: Elaboração do autor

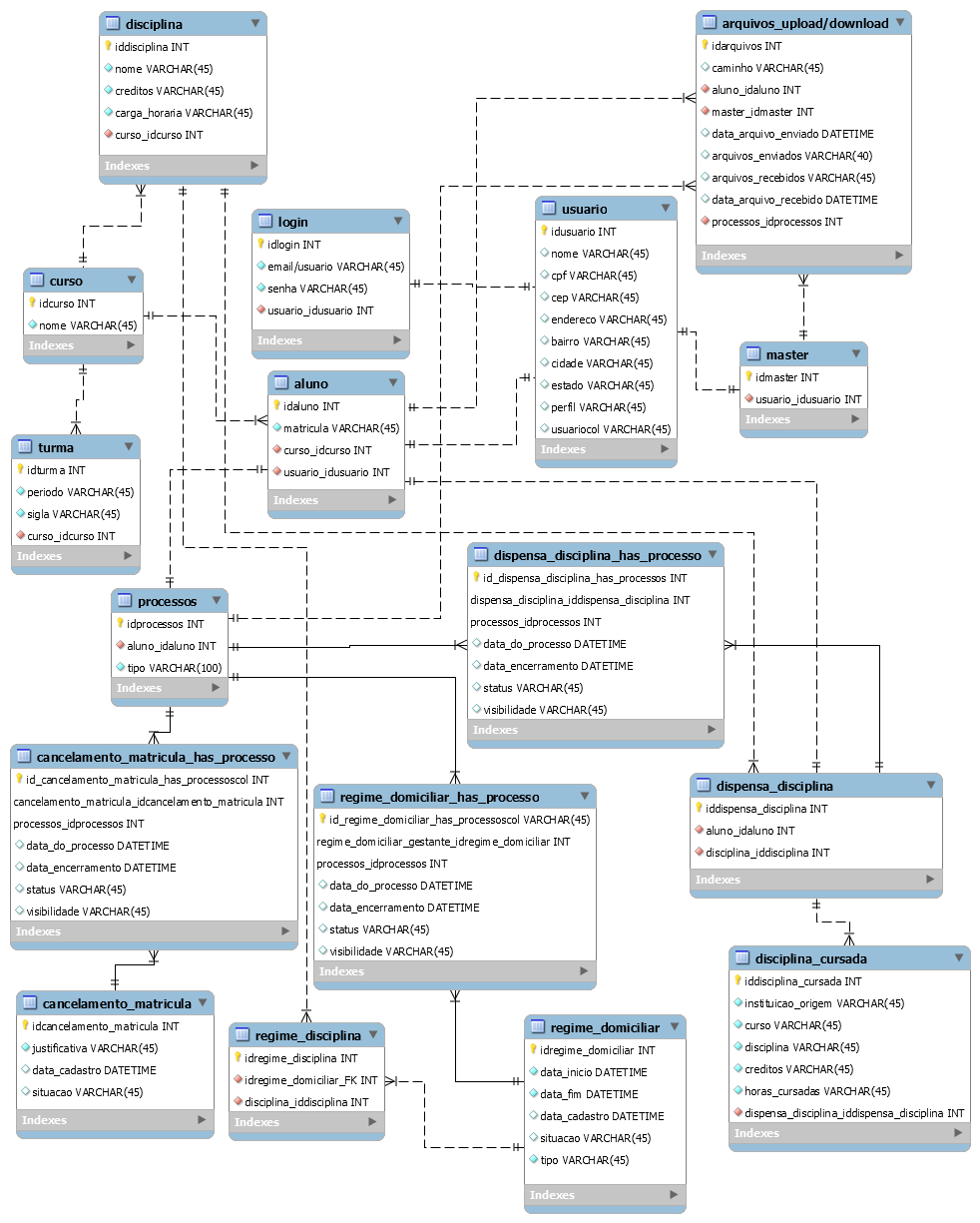
A diagramação é importante devido à possibilidade de alterações ou correções no código fonte da aplicação, e com a documentação é possível identificar e analisar determinadas funções da aplicação sem interferir em todo o sistema. Além disso, através da documentação, impactos que podem acontecer se alterar determinada linha do código fonte do sistema serão previamente detectados.

2.2 Modelagem do banco de dados

A construção do sistema iniciou-se com a criação do modelo do banco de dados, através dessa modelagem é possível visualizar, mesmo que de forma inicial, como serão as regras de negócio, e segundo Costa (2011) “DATE (2004, p. 10) Um banco de dados é uma coleção de dados persistentes, usada pelos sistemas de aplicação de uma determinada empresa”.

Para a criação da modelagem do banco de dados foi utilizada a ferramenta *MySQL Workbench 8.0 CE* devido ela ser de código aberto e possuir facilidades na criação de tabelas e relacionamentos tornando mais ágil a elaboração da documentação do banco de dados. Essa ferramenta é bastante utilizada pela sua facilidade e vasto material que orientam o uso da mesma.

Figura 3 - Modelagem do Banco de Dados



Fonte: Elaboração do autor

2.2.1 Segurança da aplicação

Para garantir a segurança das informações do banco de dados, é necessário utilizar criptografia de dados, dessa maneira as informações tornam-se ilegíveis caso sejam visualizadas por pessoas não autorizadas. Criptografia segundo Bellare, Mihir; Rogaway, Phillip (2005) refere-se à construção e análise de protocolos que impedem terceiros, ou o público de lerem mensagens privadas. “Segundo Sêmola (2003, p. 45), toda informação deve ser protegida de acordo com o grau de sigilo de seu conteúdo, visando à limitação de seu acesso e uso apenas às pessoas para quem elas são destinadas” (SILVA; ROSA, 2017).

2.3 Sistema de controle de versões

Para ter controle do código, tanto para segurança quanto para manutenção através de correções e atualizações se faz necessário o uso de um sistema de controle de versões. Esses sistemas têm a finalidade de gerenciar todo código em repositórios (todos os locais que ficam armazenados o código fonte do sistema tanto local quanto em nuvem).

Todo código que é desenvolvido fica armazenado em um repositório local - que no caso é a máquina em que se encontra esse código. Esses códigos são enviados para um repositório central, enviados para um sistema de armazenamento em nuvem, que ficarão disponíveis para atualizações e correções.

Esses sistemas também fazem o uso de versionamento que é o gerenciamento de controle de versão do código que vai ser adicionado ou modificado. Para o gerenciamento e versionamento do código fonte da aplicação foi utilizado *o git.* Segundo o site do *git* (2019), ele é um sistema de controle de versão distribuído de [código aberto e gratuito](https://git-scm.com/about/free-and-open-source) , projetado para lidar com tudo, de projetos pequenos a grandes, com velocidade e eficiência. Além disso, é fácil de aprender e tem uma pegada minúscula com desempenho extremamente rápido.

A ferramenta usada para hospedar (armazenar o código-fonte do software) foi o *Github.* Segundo o site oficial, o *Github* (2019) foi criado para que os projetos sejam compartilhados com outros desenvolvedores com o intuito que o código disponibilizado seja revisado e implementados por todos.

2.4 Java

Todo software precisa de uma linguagem de programação para que as instruções (os códigos) que forem digitadas sejam entendidas por uma máquina e executadas. A linguagem utilizada para a implementação do SGPI foi o Java, pelo fato dela estar muito presente no âmbito acadêmico além de possuir um vasto conteúdo para consulta, seja através da documentação ou na internet. Segundo a *Oracle* (2019), o Java oferece a rica interface do usuário, desempenho, versatilidade, portabilidade e segurança que os aplicativos de hoje exigem.

2.5 Plataforma de Desenvolvimento

Dentre as diversas características do Java, destaca-se a portabilidade, ou seja, pode operar em qualquer plataforma (em qualquer sistema operacional que tenha o Java instalado). Isso traz grande vantagem para o desenvolvimento pelo fato que o desenvolvedor não vai se preocupar com que sistema operacional o cliente utiliza, porque o Java traz essa vantagem.

2.6 *Visual Studio Code*

Para ajudar no desenvolvimento de qualquer aplicação, faz-se necessário uma ferramenta para auxiliar na criação de seus códigos, essas ferramentas otimizam o tempo de desenvolvimento, são muito utilizadas pelo fato de possuírem diversas funções, dentre elas podemos citar o auto completar, em que palavras reservadas da linguagem de programação são disponibilizadas na tela para o desenvolver selecionar de acordo com sua necessidade, outro ponto importante é a função de demonstrar erros de escrita no código, dessa forma a ferramenta aponta diretamente ao erro, sem a necessidade de ler diversos trechos de código.

Foi utilizado como ferramenta de desenvolvimento o editor de código fonte *Visual* *Studio Code* da microsoft, que é de código aberto e oferece recursos para o desenvolvimento de software. Segundo a documentação da ferramenta, o *Visual Studio Code* é um editor de código-fonte leve, mas poderoso, que roda na sua área de trabalho e está disponível para Windows, macOS e Linux. Ele vem com suporte interno para outras linguagens de programação como: *JavaScript, TypeScrip*t e *Node.js* e possui um rico ecossistema de extensões para outros idiomas (como C ++, C #, Java, Python, PHP, Go). (CODE, 2019)

2.7 *PostgreSQL*

Os dados manipulados pelas aplicações, de uma maneira geral, devem estar seguros, disponíveis e íntegros, para isso, utiliza-se banco de dados, onde eles ficam armazenados e disponíveis para consultas de acordo com as solicitações.

Um banco de dados é gerenciado por um Sistema Gerenciador de Bancos de Dados (SGBD) e esse gerenciador possui uma linguagem própria chamada de *Structured Query Language* (SQL). Essa linguagem é responsável por fazer todas as operações necessárias no banco de dados.

Um banco de dados é importante pelo fato dele organizar todos os dados por entidades (grupos com várias semelhanças) e armazená-los de forma segura, independente da quantidade de dados que vai ser processado, trazendo grande escalabilidade e garantia em armazenamento e segurança.

Na construção do produto de software proposto, foi escolhido o SGBD *PostgreSQL* 9.6, pois dentre algumas vantagens para esta aplicação tem-se: software livre, ou seja, não é necessário pagar por licença para utilizá-lo; boa performance; e altamente escalável.

2.8 *Apache Tomcat*

Quando se acessa uma página da WEB, existe um intermediário que disponibiliza aquelas páginas para que se possa ver seu conteúdo no navegador, esse intermediário é chamado de servidor. Uma página da internet é nada mais, nada menos que vários códigos que são interpretados pelo navegador para que seja possível acessar seu conteúdo visual. Quando um software é finalizado, ele é implantado em um servidor e o mesmo se encarrega de enviar as informações para o computador.

Por ser uma aplicação WEB se fará necessário a utilização de um servidor, que é o responsável por tratar de requisições do cliente e devolvê-las como resposta, como falado anteriormente. Dentre os servidores disponíveis para utilização, foi utilizado o *TomCat*, por ser gratuito e possuir diversos recursos que auxiliam e facilitam o desenvolvimento do sistema.

2.9 *HTML*

A tecnologia *Hyper Text* *Markup* *Language* (HTML) é a linguagem utilizada para o desenvolvimento de *web pages.* O HTML proporciona diversos artifícios para a criação de páginas estáticas, é basicamente formado por *tags* que estruturam todo o texto ou referência externa. Segundo Deitel *et al* (2008), o HTML não é uma linguagem de programação, mas de marcação que serve para identificar elementos de uma página de Internet e mostrá-la na tela do navegador.

2.10 *CSS*

O *Cascading* *Style* *Sheets* (CSS) é uma "folha de estilo" composta por “camadas” e utilizada para definir a apresentação (aparência) em páginas da internet que adotam para o seu desenvolvimento linguagens de marcação (como XML, HTML e XHTML). O CSS define como serão exibidos os elementos contidos no código de uma página da interne*t* e sua maior vantagem é efetuar a separação entre o formato e o conteúdo de um documento.

O CSS proporciona o código HTML “mais limpo”, já que a configuração dos elementos que serão exibidos na página pode ser feita em um arquivo CSS externo, dessa forma também é possível reutilizar definições de estilo em diversas páginas do sistema.

2.11 *Bootstrap*

O *Bootstrap* é um *framework* de desenvolvimento web que é essencial para qualquer desenvolvedor porque ele traz modelos prontos de componentes para serem reutilizados, e através dessa estratégia os softwares poderão ser produzidos em menor quantidade de tempo. Segundo Kronig, Costa e Spínola (2005) Muitos dos sistemas de software construídos são semelhantes. Portanto, devem-se considerar estes sistemas, avaliar seus componentes e determinar suas adaptações ou até mesmo reutilizá-los por completo na construção de um novo sistema.

2.12 *Apache Maven*

O *Apache* *Maven* ou *Maven* é uma ferramenta de gerenciamento de bibliotecas para projetos *Java*. O *Maven* gerencia bibliotecas *Java* e seus *plugins* dinamicamente de um ou mais repositórios, pois possui um repositório local, ou seja, no próprio computador do usuário, como também um repositório remoto, que pode ser acessado de qualquer ponto.

Pela necessidade da utilização de diversas bibliotecas *Java*, o *Apache* *Maven* foi usado para resolver o problema de gerenciamento dessas bibliotecas, facilitando a implementação e organizando melhor o sistema.

2.13 *Jaspersoft Studio*

Uma funcionalidade bastante importante para qualquer sistema é a visualização dos dados gerados de forma organizada através de relatórios. Os relatórios são consultas feitas pelos usuários para obterem um resultado preciso e estatístico. É através desses dados agrupados no relatório, que os usuários poderão observar e/ou tomarem decisões em suas atividades, facilitando muito o seu trabalho e trazendo valor ao que ele pretende buscar, disponibilizando consultas detalhadas.

O SGPI disponibiliza geração de relatórios através da ferramenta *Jaspersoft* *Studio* que é uma ferramenta de criação e modelagem de relatórios. *Jaspersoft* (2019) com essa ferramenta pode-se criar documentos de qualquer complexidade a partir dos seus dados - desde PDFs prontos para impressão até obras-primas interativas em HTML dinâmico, com navegação para componentes interativos, como mapas e gráficos que podem estar dentro ou fora do relatório.

2.14 *JUnit*

Para garantir que o software atenda o que foi definido nos requisitos se faz necessário o uso de testes, eles são muito importantes para que o software atinja a qualidade necessária e o produto atenda às necessidades do cliente. Nesta etapa, todas as funcionalidades do software devem ser testadas evitando problemas futuros com o software.

O *JUnit* é um *framework* de código aberto, com suporte à criação de testes automatizados na linguagem de programação *Java*. Esse *framework* facilita a criação de código para a automação de testes com apresentação dos resultados. O *JUnit* permite ao desenvolvedor fazer testes unitários, ou seja, pode ser verificado se cada método de uma classe funciona da forma esperada, exibindo possíveis erros ou falhas.

2.15 Cenário de Testes

O plano de testes funciona como um guia sobre as funcionalidades implementadas e testadas. Um projeto de software pode ter vários planos de teste; estes cenários podem também instruir os testadores do software na produção de novos testes. Além de verificar as funcionalidades do sistema os cenários de teste também são importantes para a documentação do sistema.

Tabela 1 - Cenário de Testes

|  |  |
| --- | --- |
| **Cenários de Testes** | |
| **Descrição** | Autenticação do Usuário |
| **Objetivo** | Fazer autenticação do usuário acessando os logins cadastrados no banco de dados |
| **Cenários** | |
| **Item** | Descrição |
| **01** | **Cenário:** O usuário não possui login e senha para acessar o sistema.  **Resultado:** O sistema não permitirá efetuar o acesso deste usuário. |
| **02** | **Cenário:** O usuário digita o login ou senha inválidos  **Resultado:** O sistema não permitirá a autenticação deste usuário. |
| **03** | **Cenário:** O usuário informa tanto o login quanto a senha válidos  **Resultado:** O sistema permitirá a autenticação deste usuário |

Fonte: Elaboração do autor

2.15.1 Funcionalidades implementadas e testadas

Até o momento, para este sistema, foram implementadas as seguintes funcionalidades e na mesma sequencia se ela foi testada:

Tabela 2 - Funcionalidades Implementadas

|  |  |  |
| --- | --- | --- |
| Funcionalidade | Implementação | Teste |
| Gerenciamento de dados dos usuários (cadastro, remoção, atualização e visualização) | Feita | Feito |
| Gerenciamento de dados dos alunos (cadastro, remoção, atualização e visualização) | Feita | Feito |
| Gerenciamento de dados dos cursos (cadastro, remoção, atualização e visualização) | Feita | Feito |
| Gerenciamento de dados das turmas (cadastro, remoção, atualização e visualização) | Feita | Feito |
| Gerenciamento de dados das disciplinas (cadastro, remoção, atualização e visualização) | Feita | Feito |
| Gerenciamento dos dados de cancelamento de matrícula (cadastro, remoção, atualização e visualização) | Feita | Feito |
| Gerenciamento dos dados de regime domiciliar (cadastro, remoção, atualização e visualização) | Feita | Feito |
| Gerenciamento de dados da dispensa de disciplina (cadastro, remoção, atualização e visualização) | Feita | Feito |
| Gerenciamento de dados dos processos (cadastro, remoção, atualização e visualização) | Feita | Feito |
| Autenticação dos usuários | Feita | Feito |
| Relatório dos processos | Não | Não |
| Relatório dos usuários | Não | Não |
| Relatório dos alunos | Não | Não |
| Relatório dos processos de cancelamento de matrícula | Não | Não |
| Relatório dos processos de regime domiciliar | Não | Não |
| Relatório dos processos de dispensa de disciplina | Não | Não |

Fonte: Elaboração do autor

2.16 Descrição dos Requisitos

Nesta seção aborda-se sobre os requisitos do sistema, entre funcionais e não funcionais, estabelecendo um conjunto de objetivos gerais que o mesmo deve cumprir, ou seja, características do que o sistema deve fazer.

2.16.1 Requisitos funcionais

De acordo com Sommerville (2003), requisitos funcionais são declarações de funções sobre como o sistema deve reagir a entradas específicas e como deve se comportar em determinadas situações. Ou seja, requisitos funcionais são todas as funcionalidades que são implementadas em um sistema, esses requisitos são importantes para dar um detalhamento claro sobre o que determinada função faz. Seguindo a metodologia *Scrum,* descreve-se abaixo o *Product Backlog*, que é formado por uma lista de requisitos que foram definidos com o cliente. No quadro abaixo, pode-se ver o detalhamento das tarefas realizadas.

Tabela 3 - Requisitos Funcionais

|  |  |  |
| --- | --- | --- |
| **#** | **Tarefa** | **Descrição do Item *Backlog*** |
| **01** | Gerenciamento de dados dos usuários | O sistema disponibilizará dos cadastros dos usuários que serão armazenados em um banco de dados que poderão fazer consultas, alterações e remoção desses dados. |
| **02** | Documentação | Produzir em paralelo todas as funções implementadas no software, fazendo qualquer alteração quando for necessário. |
| **03** | Controle de acesso | Quando o usuário informar seu login e senha, o sistema vai retornar se ele vai ser autenticado ou não. Caso sua autenticação seja aceita de acordo com os dados do banco de dados, ele vai ser redirecionado para seu página de acordo com o seu perfil; se caso não for aceita, ele vai mostrar que o login ou senha são inválidos. |
| **04** | Upload de arquivos | O sistema disponibilizará para que seja feito upload de arquivos para que tanto os alunos quanto os funcionários possam utilizar esta funcionalidade. O aluno pode utilizar esta função para enviar qualquer documentação que seja necessária para abrir um processo. O funcionário pode enviar qualquer documentação que precise ser aprovada pela direção para os alunos. |
| **05** | Download de arquivos | O sistema disponibilizará que tanto alunos quanto funcionários possam baixar arquivos. |
| **06** | Gerenciamento de dados dos alunos | Caso o perfil do usuário seja o funcionário, ele terá permissão de adicionar, alterar, visualizar e remover os alunos. |
| **07** | Gerenciamento de dados dos cursos | Caso o perfil do usuário seja o funcionário, ele terá permissão de adicionar, alterar, visualizar e remover os cursos. |
| **08** | Gerenciamento de dados das disciplinas | Caso o perfil do usuário seja o funcionário, ele terá permissão de adicionar, alterar, visualizar e remover as disciplinas. |
| **09** | Gerenciamento de dados das turmas | Caso o perfil do usuário seja o funcionário, ele terá permissão de adicionar, alterar, visualizar e remover as turmas. |
| **10** | Gerenciamento dos dados de cancelamento de matrícula | Caso o perfil do usuário seja o funcionário, ele terá permissão de adicionar, alterar, visualizar e remover os processos de cancelamento de matrícula. Caso o perfil do usuário seja aluno, ele poderá fazer a solicitação de cancelamento de matrícula. |
| **11** | Gerenciamento dos dados de regime domiciliar | Caso o perfil do usuário seja o funcionário, ele terá permissão de adicionar, alterar, visualizar e remover os processos de regime domiciliar. Caso o perfil do usuário seja o aluno ele poderá fazer a solicitação de regime domiciliar. |
| **12** | Gerenciamento de dados da dispensa de disciplina | Caso o perfil do usuário seja o funcionário, ele terá permissão de adicionar, alterar, visualizar e remover os processos de dispensa de disciplina. Caso o perfil do usuário seja o aluno, ele poderá fazer a solicitação de dispensa de disciplina. |
| **13** | Gerenciamento de dados dos processos | Apenas os usuários com perfil funcionário poderão executar essas funcionalidades de adicionar, alterar, remover e listar esses processos. |
| **14** | Relatórios | O sistema irá disponibilizará os relatórios de usuários, alunos, curso, disciplinas, turmas, cancelamento de matrícula, regime domiciliar, dispensa de disciplina e processos, para que esses dados sejam trabalhados e impressos. |

Fonte: Elaboração do autor

2.16.2 Requisitos não funcionais

De acordo com Sommerville (2007, p.77), requisitos não funcionais são relacionados ao uso da aplicação em termos de desempenho, usabilidade, confiabilidade, disponibilidade, segurança e tecnologias envolvidas.

Tabela 4 - Requisitos Não Funcionais

|  |  |  |  |
| --- | --- | --- | --- |
| **Item** | **Atributo** | **Detalhes** | **Prioridade** |
| **RNF01** | Interface | Interface com boa usabilidade, utilizando menus laterais, formulários para entrada de dados e alertas para identificação de erros. | Importante |
| **RNF02** | Ambiente | O sistema deve ser desenvolvido em ambiente *web*. | Essencial |
| **RNF03** | Segurança | Somente usuários cadastrados terão acesso ao sistema. | Essencial |

Fonte: Elaboração do autor

2.17 Processos Empregados

### Neste tópico serão mostrados quais foram às funcionalidades encarregadas para o desenvolvimento do software e qual papel que cada pessoa ficou para o desenvolvimento do mesmo.

2.17.1 Definição de papéis

**Cliente**: Devido à pequena quantidade de pessoas envolvidas neste projeto, o papel do cliente ficou nas mãos do orientador (deste trabalho de pesquisa) juntamente com as funcionárias da Secretaria Geral que participaram das reuniões e relataram quais os problemas enfrentados no setor. O cliente tem o papel de descrever as funcionalidades do sistema, acompanhar o andamento do projeto, perceber qual o nível de prioridade de cada funcionalidade.

**Gerente de Projeto:** Quem ficou encarregado de ser o gerente de projeto também foi o orientador. O gerente de projeto tem o papel de definir as *releases* e *sprints*, reunir informações necessárias para a elaboração do sistema, garantir que o produto seja entregue conforme o planejado, além de analisar e gerenciar o andamento do projeto.

**Desenvolvedor:** Coube ao orientando assumir o papel do desenvolvedor, assim a equipe de desenvolvimento é formada por apenas uma pessoa. Dentre suas atribuições, o desenvolvedor deve auxiliar no levantamento de requisitos, implementar os itens definidos no *Product Backlog,* realizar testes e atualizar a documentação sempre que for necessário.

2.18 Processos internos

Os documentos que fazem parte das regras de negócio do software são: declaração, aproveitamento de disciplinas, cancelamento de matrícula e o regime de trabalhos domiciliares.

Atualmente um processo é criado a partir da solicitação do aluno, em que o mesmo é encarregado de preencher manualmente os formulários necessários. Em seguida a documentação é protocolada junto à Secretaria Geral onde é gerado um código numérico que se torna em processo; após o trâmite, o processo aguarda movimentação na Secretaria Geral até ser concluído; e por fim armazenado de forma física (em papel) nos arquivos do setor.

O tipo de pesquisa utilizado neste projeto foi o de estudo de caso, pelo fato de ter sido feita o levantamento de informações teóricas, observações e experiências obteve-se um conhecimento detalhado sobre o funcionamento destes processos. Foi necessário ter um contato direto com todas as pessoas envolvidas para obter o máximo de detalhes possíveis para a elaboração dos requisitos.

Para o levantamento dos requisitos, entendimento do processo e desenvolvimento do software, foram necessários questionários e diversos encontros com todos os funcionários da Secretaria Geral. Foram discutidos os principais pontos para a implementação deste software, bem como os problemas levantados durante essas reuniões, além disso, foi necessária a obtenção dos formulários de cada processo citado anteriormente para dar início à análise de requisitos tanto do sistema quanto do banco de dados.

3 CONSIDERAÇÕES FINAIS

Na construção desta documentação assim como o software foram utilizados alguns conhecimentos de engenharia de software, conhecimentos de técnicas e habilidades necessárias para a construção do mesmo.

O Sistema Gestor de Processos Internos (SGPI) pretende auxiliar no gerenciamento de processos da Secretaria Geral da FAFIC automatizando essas atividades para que sejam executadas da maneira mais fácil e rápida.

Para a criação desse software não foi possível existir integração com o sistema principal da faculdade que é o SOPHIA porque não existiam meios (até o momento da criação desta documentação) para que os dados do banco de dados desse sistema fossem integrados ao SGPI, e devido à limitação do tempo foi necessário criar um banco independente.

O software atuando independente de qualquer sistema existente na faculdade não permitiu o reaproveitamento dos dados já existentes em outros sistemas. Por isso, os dados de alunos, cursos, disciplinas e turmas tiveram que ser recriadas e colocadas em um banco exclusivo para o SGPI.

3.1 Trabalhos futuros

Como se trata de um trabalho acadêmico segue algumas considerações para futuras implementações. Como esses processos necessitam da assinatura do coordenador, seria interessante a criação de uma funcionalidade envolvendo a assinatura digital para validar a documentação, sugerimos também à integração com outros sistemas da faculdade para sincronizar e manterem os dados atualizados, reaproveitando dados já cadastrados no sistema utilizado pela FAFIC.

**REFERÊNCIAS**

ANDRADE, Antonio José F.; OLIVEIRA, Jany C. de; BARBOSA, Paulo Alberto M.. **Gestão de Projeto com Scrum: Um Estudo de Caso.**Disponível em: <https://www.enucomp.com.br/2012/conteudos/artigos/scrum.pdf>. Acesso em: 19 out. 2019.

BECK, K**. Programação extrema explicada: acolha as mudanças.** Porto Alegre, RS**:** Bookman**,** 2004**.**

CABRAL, Ademilson Angelo; SILVA, Diego Bezerra da; SOUZA, Alan Pinheiro de. **A PROBLEMÁTICA DO DESENVOLVIMENTO DE SOFTWARE: CRISE OU CALAMIDADE CRÔNICA?**Disponível em: <http://revistaconexao.aems.edu.br/wp-content/plugins/download-attachments/includes/download.php?id=1614>. Acesso em: 18 out. 2019.

CODE, Visual Studio. **Code editing. Redefined.**Disponível em: <https://code.visualstudio.com/>. Acesso em: 19 out. 2019.

COSTA, Elisângela Rocha da. **BANCOS DE DADOS RELACIONAIS.**2011. 63 f. TCC (Graduação) - Curso de Tecnólogo em Processamento de Dados, Faculdade de Tecnologia de São Paulo, São Paulo, 2011. Disponível em: <http://www.fatecsp.br/dti/tcc/tcc0025.pdf>. Acesso em: 24 out. 2019.

DEITEL, H. M., DEITEL P.J., & NIETO, T.R.. **Internet & World Wide Web: como programar.** 2ed. Porto Alegre-RS: Bookman,2003.

GIT. Git - **Tudo é Local**. Disponível em: <https://git-scm.com/>. Acesso em: 25 out. 2019.

GITHUB. **Built for developers.**Disponível em: <https://github.com/>. Acesso em: 25 out. 2019.

JASPERSOFT. **Jaspersoft® Studio.**Disponível em: <https://community.jaspersoft.com/project/jaspersoft-studio>. Acesso em: 24 out. 2019.

KRONIG, Rosangela; COSTA, Ivanir; SPÍNOLA, Mauro. **Uma proposta de um processo prático para apoiar o reuso de software.**2005. Disponível em: <http://www.abepro.org.br/biblioteca/enegep2005\_enegep0801\_1233.pdf>. Acesso em: 23 out. 2019.

LARMAN, C. Utilizando UML e padrões: uma introdução a analise e ao projeto orientado a objetos. Porto Alegre: Bookman, 2000.

LUCIDCHART. **Finalmente, diagramas bem feitos.**Disponível em: <https://www.lucidchart.com/pages/pt/tour-do-produto-lucidchart>. Acesso em: 21 out. 2019.

LUCIDCHART. **O que é um diagrama UML?**Disponível em: <https://www.lucidchart.com/pages/pt/o-que-e-uml>. Acesso em: 21 out. 2019.

MANHÃES, V**. Um Estudo de Caso da Adoção das Práticas e Valores do *EXtreme Programming*.** 2005. 181 f. Dissertação (Mestrado em Informática) – Núcleo de Computação Eletrônica, Universidade Federal do Rio de Janeiro. Rio de Janeiro. 2005.

ORACLE. **Visão geral do Java SE**. Disponível em: <[https://www.oracle.com/technetwork/java/javase/overview/index.html](https://www.google.com/url?q=https://www.oracle.com/technetwork/java/javase/overview/index.html&sa=D&ust=1574341837093000&usg=AFQjCNFly6bbKj5ASL9J_CwGP1o0RstxZQ)>. Acesso em: 23 out. 2019.

SILVA, Samuel Fabiano Barbosa. **ENGENHARIA DE REQUISITOS:**Uma análise das técnicas de levantamento de requisitos. Disponível em: <http://professores.dcc.ufla.br/~terra/publications\_files/students/2012\_fumec\_silva.pdf>. Acesso em: 19 out. 2019.

SILVA, Thaynára da; ROSA, Paulo Roberto. **SEGURANÇA EM BANCO DE DADOS.**2017. Disponível em: <http://www.unoeste.br/site/enepe/2017/suplementos/area/Exactarum/2%20-%20Ci%C3%AAncias%20da%20Computa%C3%A7%C3%A3o/SEGURAN%C3%87A%20EM%20BANCO%20DE%20DADOS.pdf>. Acesso em: 20 out. 2019.

SOMMERVILLE, I. **Engenharia de *software***. 6ª ed. São Paulo: Pearson Education Companion, 2003.

SOMMERVILLE, I. **Engenharia de *software***. 8ª ed. São Paulo: Pearson Addison-Wesley, 2007.

1. <http://www.martinsfontespaulista.com.br/anexos/produtos/capitulos/243418.pdf> [↑](#footnote-ref-1)
2. um framework de desenvolvimento é uma “base” de onde se pode desenvolver algo maior ou mais específico. É uma coleção de códigos-fonte, classes, funções, técnicas e metodologias que facilitam o desenvolvimento de novos softwares. [↑](#footnote-ref-2)