**Plano de Implantação**

|  |  |
| --- | --- |
| ***importaSiSU*** | |
| **Gerente de Projeto** | |
| Pedro Rodolfo Gomes de Souza | |
| rodolfo@cin.ufpe.br | |
| +55 81 9 9706-8582 | |

|  |
| --- |
| ***Objetivo deste Documento*** |
| Este documento tem como objetivo descrever informações importantes, a serem consideradas, na implantação da API de integração entre o SiSU e o SIGAA. |

|  |  |  |  |
| --- | --- | --- | --- |
| ***Histórico de Revisão*** | | | |
| **Data** | **Autor** | **Descrição** | **Versão** |
| 17/06/2019 | Pedro Rodolfo | Início da realização do documento | Início 1.0 |
| 20/07/2019 | Pedro Rodolfo | Adições no documento | 1.1 |
|  |  |  |  |

Documento baseado no “*Metodologia de Desenvolvimento de Software - Versão 1.0”,* desenvolvido pelo **DATASUS.**

Sumário

1. **Introdução** 
   1. Motivação
   2. Escopo
   3. SISU
   4. SIGAA
   5. Processo de Matrícula de Ingressantes da UFPE
2. **Planejamento de implantação** 
   1. Fatores Críticos
   2. Papéis e Responsabilidades
   3. Órgãos Impactados
      1. NTI
      2. PROACAD
      3. PROCIT
   4. Possíveis Riscos na Implantação
3. **Cronograma**
4. **Recursos** 
   1. Facilidades
   2. Hardware
   3. Release de Implantação
      1. Software
      2. Documentação de Suporte
      3. Equipe e Suporte
5. **Recomendações**
6. **Treinamento**
   1. Etapas do Treinamento
   2. Material
7. **Aprovações do Cliente**
   1. Kick-off
   2. 1º Status Report
   3. 2º Status Report
8. **Conclusão**
9. **Apêndice: Manual de Implantação Técnico**
10. **Introdução**
    1. **Motivação**

Atualmente, segundo o cliente, existe um grande gasto de trabalho na integração dos dados do SiSU (providos pelo Ministério da Educação) com o atual sistema de matrículas da UFPE, o SIG@. E visando evitar esse gasto de trabalho após a implantação do novo sistema, o SIGAA, se faz necessário planos para a implantação de uma ferramenta que otimize/automatize esse processo.

* 1. **Escopo**

|  |  |  |  |
| --- | --- | --- | --- |
| Descrição Detalhada do Produto do Projeto | | | |
| O produto será um Plano de Implantação para o módulo de matrículas do SIGAA, que fará integração com o SiSU (Sistema de Seleção Unificada), e será usado pelo NTI e Corpo Discente, para realização das matrículas dos ingressantes da UFPE. | | | |
| Principais Entregas do Projeto / Critérios de Aceitação | | | |
| Entregas | | Critérios de Aceitação | |
| Modelagem BPMN, i\* e Ishikawa do AS-IS do atual processo de matrículas | | O BPMN, i\* e Ishikawa deve estar de acordo com o processo atual de matrícula dos ingressantes da UFPE | |
| Modelagem BPMN, i\* e Ishikawado TO-BE | | O BPMN, i\* e Ishikawa deve estar de acordo com o processo que será implementado | |
| Documentação formal e descritiva do Plano de Implantação | | Estar detalhado e com todas as informações necessárias | |
| Apresentação do Plano de Implantação | | Exposição e explicação concisa | |
| Escopo Não Contemplado | | | |
| A implantação e seus custos, além dos treinamentos para utilização do módulo de matrículas não estão dentro do escopo do projeto. Atualizações futuras e manutenção do sistema final não serão contemplados. | | | |
| Premissas | | | |
| * Os principais afetados e beneficiados serão o NTI e a Coordenação de Divisão de Corpo Discente da PROACAD. * Partimos do pressuposto que as informações coletadas em entrevistas, reuniões e questionários são completas e verídicas. * Os prazos estipulados serão cumpridos | | | |
| Restrições | | | |
| * Curto período para a realização do projeto (3 meses) * Os membros da equipe não podem dar total atenção a esse projeto * Dificuldade de acesso à informação, fazendo com que sejam necessárias visitações, reuniões, entrevistas, questionários e leitura do código fonte do SIGAA | | | |

* 1. **SiSU**

Segundo o Ministério da Educação, “O Sistema de Seleção Unificada (SiSU) é o sistema informatizado, gerenciado pelo Ministério da Educação (MEC), pelo qual instituições públicas de educação superior oferecem vagas a candidatos participantes do Exame Nacional do Ensino Médio (Enem).” É um sistema desenvolvido pelo Núcleo de Informações Executivas, da Secretaria Executiva do MEC. No ano de 2019, ofereceu 235.476 vagas em 129 instituições espalhadas pelo Brasil (apenas na sua edição do 1º semestre). E houveram 3.492.751 inscrições de 1.823.871 candidatos. Na Universidade Federal de Pernambuco, esse processo é gerenciado pela divisão de Corpo Discente (órgão da Pró-reitoria para Assuntos Acadêmicos, a PROACAD), gerido pela Cláudia Lucas. Na última edição (que ocorreu no primeiro semestre do ano de 2019), foram oferecidas 6.972 vagas para a UFPE, para os dois semestres. Houveram 3 listas de classificação (e 2 listas de convocação). O processo ocorreu de 30/01/2019 até 13/03/2019. No ano de 2019, houveram alguns problemas no processo de matrícula por conta da avaliação racial, agora realizada pela universidade, para a comprovação de direito à utilização das cotas raciais, atrasando ainda mais o processo de matrícula.

* 1. **SIGAA**

O SIG-UFRN, sigla para Sistemas Institucionais Integrados de Gestão da Universidade Federal do Rio Grande do Norte é uma suíte de aplicações de Sistemas de Informação, do tipo SIG (Sistemas de Informação Gerencial), que auxilia as organizações com todas as informações necessárias para a gestão eficiente da mesma. O SIG-UFRN surgiu no ano de 2004, através da SINFO, a Superintendência de Informática da Universidade Federal do Rio Grande do Norte. E um de seus módulos é o SIGAA, segundo a descrição da própria SINFO/UFRN, informatiza os procedimentos da área acadêmica através dos módulos de: graduação, pós-graduação (stricto e lato sensu), ensino técnico, ensino médio e infantil, submissão e controle de projetos e bolsistas de pesquisa, submissão e controle de ações de extensão, submissão e controle dos projetos de ensino (monitoria e inovações), registro e relatórios da produção acadêmica dos docentes, atividades de ensino a distância e um ambiente virtual de aprendizado denominado Turma Virtual. Também disponibiliza portais específicos para: reitoria, professores, alunos, tutores de ensino a distância, coordenações lato-sensu, stricto-sensu e de graduação e comissões de avaliação (institucional e docente).

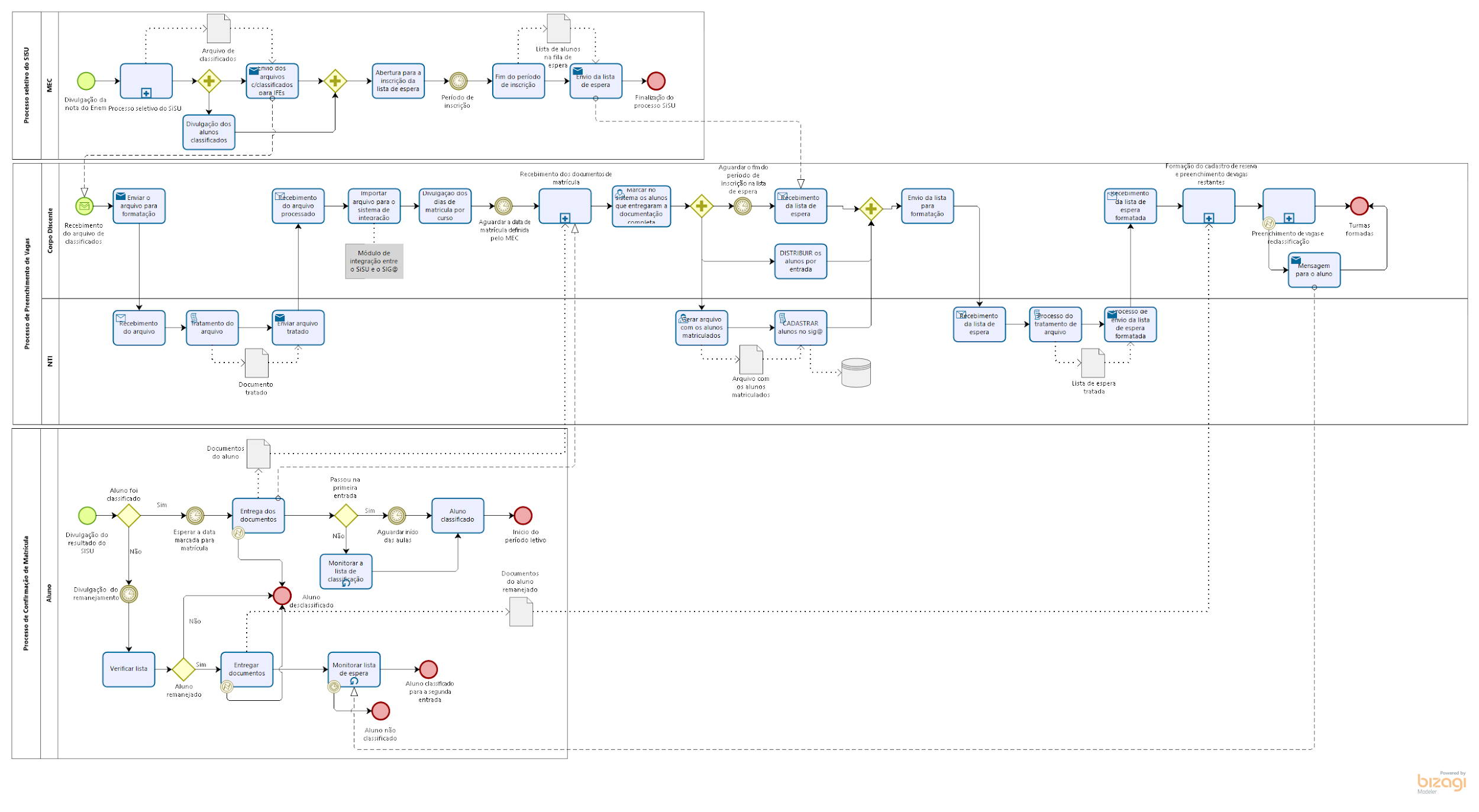
* 1. **Processo de Matrícula da UFPE**

Depois da divulgação das notas do ENEM, o MEC dá início ao processo seletivo do SISU. Ao fim deste processo os alunos são classificados, ou não, para os cursos em que se inscreveram de diversas universidades. Um arquivo contendo os alunos aprovados na primeira chamada de alunos é formada para cada uma das instituições envolvidas no processo e em seguida enviada para cada uma destas.

Uma vez que o arquivo da UFPE é recebida pelo Corpo Discente, esta é enviada para o NTI para receber o devido tratamento.

Após a divulgação da nota do ENEM, o SiSU efetua um processo seletivo, que gera um arquivo de alunos classificados, então, em paralelo com a divulgação dos alunos que passaram no vestibular, faz o envio do arquivo de classificados para o corpo discente da UFPE. Após finalizados o envio e a divulgação, é esperado o tempo de inscrição dos alunos, o SiSU gera a lista de alunos na fila de espera e faz o envio dessa lista para o corpo discente. Após o envio, o processo por parte do SiSU é finalizado.

Recebido o arquivo dos alunos classificados pelo corpo discente, é iniciado o processo por parte da UFPE. Após o recebimento desse documento, a lista dos alunos classificados é enviada para o NTI, que, após recebimento, faz o tratamento do arquivo para que se adeque ao processo de preenchimento de vagas. Após o arquivo ser tratado, é enviado novamente ao corpo discente para que seja feita a continuação do processo. Após receber o arquivo, o corpo discente envia os dados para a ferramenta de Integração com o SiSU, aguarda a data de matrícula definida pelo MEC, então, divulga os dias de matrícula de cada curso. Depois de serem divulgados os dias de matrícula é efetuado o recebimento dos documentos de matrícula. Um subprocesso que consiste em: Após o início do processo de matrícula, receber os documentos dos alunos. Se os documentos não estiverem autenticados, o aluno deve ser enviado para autenticar os documentos. Após autenticados, recebidos, se o aluno possui documentos faltando, deve ser desclassificado, perdendo sua vaga. O aluno sendo classificado, é emitido o seu comprovante de matrícula, e o subprocesso é finalizado. Após o recebimento dos documentos, o corpo discente classifica os alunos que se matricularam e elimina os alunos que não compareceram, então, em paralelo, faz a distribuição dos alunos por entrada. O NTI gera o arquivo com os alunos matriculados, faz o cadastro destes no SIG@ e o corpo discente aguarda o fim do período de inscrição da lista de espera e efetua o recebimento dos alunos na lista de espera. Depois de ter feito essas três atividades em paralelo, o corpo discente envia para o NTI a lista de espera para ser formatada. Recebendo essa lista, o NTI faz o processo de tratamento do arquivo e envia novamente para o corpo discente, que faz o preenchimento das vagas e reclassificação, finalizando assim o processo.



*Figura 1 Modelagem em BPMN do atual processo de matrícula. (A figura encontra-se em melhor resolução no Github do projeto, através do seguinte link:* [*https://github.com/prgs1/importasisu/blob/master/Entregas/GPN/Entrega%201/Modelagem%20em%20BPMN%20do%20AS-IS%20do%20processo%20de%20matr%C3%ADcula.png*](https://github.com/prgs1/importasisu/blob/master/Entregas/GPN/Entrega%201/Modelagem%20em%20BPMN%20do%20AS-IS%20do%20processo%20de%20matr%C3%ADcula.png)*)*

|  |
| --- |
|  |

*Figura 2 Processo de recebimento dos documentos de matrícula*

|  |
| --- |
|  |

*Figura 3 Processo de formação do cadastro de reserva e preenchimento de vagas restantes*

|  |
| --- |
|  |

*Figura 4 Preenchimento de vagas e reclassificação.*

1. **Planejamento de Implantação**
   1. **Fatores Críticos**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Descrição | What? | Why? | Who? | Where? | When? | How? | How Much? |
| Cooperação Departamental | Os diferentes sistemas usados (pelos diferentes departamentos) | Há vários departamentos envolvidos e há um prazo para finalização do processo. | Diretor de Projetos do NTI e diretora do Corpo Discente | Corpo discente, NTI e MEC | Durante todo o processo de matrícula | Automação do processo reduzirá os possíveis erros de comunicação entre os departamentos | 2 |
| Gestão do Projeto | Processo de negócio, sistema e cronograma | Para evitar erros e atrasos na implementação da solução | Gestor e equipe de desenvolvimento | PROCIT e DevTeam | Durante todo o processo de desenvolvimento da solução | Comunicação da equipe, modelagem adequada do processo e dos problemas | 3 |
| Gestão das Expectativas | O escopo do projeto | É importante deixar o usuário ciente do que está dentro do escopo do projeto | O gerente do projeto e o usuário | Onde o projeto será implementado (SIGAA/NTI) | Nas reuniões com os stakeholders | Através de diálogo e negociação com o cliente | 2 |
| Análise e Conversão de Dados | As planilhas do MEC e dados gerados pelo sistema de importação | O sucesso do processo depende da troca de dados entre diversas plataformas | O NTI e o sistema depois de automatizado | Dos terminais dos usuários do sistema de importação, no NTI | Durante as três distribuições de ingressantes | Através do correto tratamento das planilhas e planejamento do formato dos arquivos do sistema de importação | 2 |
| Reengenharia do Processo | Análise dos pontos de gargalo do processo e melhorias destes | O processo de matrícula, da forma como se encontra, é lento e problemático | A equipe de desenvolvimento | NTI / PROCIT / PROACAD | Nos pontos onde o gargalo acontece | Através do planejamento cuidadoso do novo processo | 3 |

* 1. **Papéis e Responsabilidades**

|  |  |
| --- | --- |
| **Stakeholder** | **Função** |
| Marlos Ribeiro | Coordenador de Projetos - NTI/UFPE |
| Equipe INTEGRAÇÃO / SIGAA | Equipe do NTI que realizará a integração entre o SiSU e o SIGAA |
| Cláudia Lucas | Coordenadora do Corpo Discente da Proacad (Pró-reitoria para Assuntos Acadêmicos) |

* 1. **Órgãos Impactados**
     1. **NTI**

Segundo a descrição no seu site oficial, O Núcleo de Tecnologia da Informação (NTI) é o órgão suplementar da UFPE responsável por realizar a gestão de infraestrutura de software e hardware da UFPE e o planejamento e execução da política de informática da universidade. O NTI tem também a responsabilidade de pesquisar, desenvolver, executar e participar de projetos em Tecnologia de Informação e serviços de informática, bem como de captar recursos através de projetos, consultorias e serviços.

* + 1. **PROACAD**

Segundo a descrição no seu site oficial, A Pró-Reitoria de Assuntos Acadêmicos (Proacad) é responsável pelos assuntos referentes ao ensino de graduação da Universidade Federal de Pernambuco. Cabe a ela atuar junto aos estudantes, docentes, coordenadores de cursos, coordenadores das áreas básicas, chefes de departamentos e diretores de centro; garantir a qualidade dos cursos de graduação oferecidos e reafirmar seu compromisso social, articulando o ensino com as áreas de pesquisa e extensão; a organização, o funcionamento e a política didático-pedagógica da graduação e a coordenação do processo de avaliação da qualidade do ensino; coordenação da qualificação dos docentes dos cursos de graduação e o controle acadêmico dos estudantes no que diz respeito aos aspectos legais e documentação.

* + 1. **PROCIT**

Segundo a descrição no seu site oficial, a Pró-Reitoria de Comunicação, Informação e Tecnologia da Informação (PROCIT) tem por missão direcionar, articular, executar e monitorar, com transparência e responsabilidade, as políticas e ações com as unidades de comunicação, informação e tecnologia da informação, atuando transversalmente na UFPE, visando maior integração e interação com as comunidades interna e externa.

* 1. **Possíveis Riscos na Implantação**

Dificuldade no desenvolvimento da API e entendimento do código fonte do SIGAA, uma vez que o desenvolvimento da API depende não só de conhecimento nas ferramentas escolhidas, mas também do código fonte do SIGAA.

1. **Cronograma**

|  |  |  |
| --- | --- | --- |
| **Tarefa** | **Prazo** | **Responsável** |
| Desenvolvimento da API | 6 meses | Equipe de Desenvolvimento - NTI |
| Implantação parcial e customização | 6 meses | Equipes de Desenvolvimento e Integração - NTI |
| Implantação Final | 6 meses | Equipe de Integração - NTI |
| Treinamento e Capacitação | 6 meses | Marlos - NTI |
| Utilização do sistema e geração de Feedbacks | 12 meses | NTI e Corpo Discente |
| Customizações pelos Feedbacks | 6 meses | NTI |

1. **Recursos**

Esta seção tem como objetivo definir os recursos necessários ao desenvolvimento e implantação da API de integração.

* 1. **Instalações**

As instalações atuais do NTI e UFPE são suficientes para o desenvolvimento e implantação da API.

* 1. **Hardware**

As necessidades de Hardware para desenvolvimento da API são atualmente atendidas pelos computadores e dispositivos pertencentes ao inventário da UFPE.

Dependendo da escolha de como a plataforma será implantada, pode ser necessário um servidor dedicado para hospedagem da API. No entanto, existe a opção de implementação da API como módulo isolado do SIGAA, eliminando essa necessidade.

* 1. **Release de Implantação**
     1. **Software**

A seguir as tecnologias sugeridas para desenvolvimento e implantação da ferramenta:

* Linguagem de programação Java: linguagem de programação orientada a objetos desenvolvida na década de 90 por uma equipe de programadores chefiada por James Gosling, na empresa Sun Microsystems. Em 2008 o Java foi adquirido pela empresa Oracle Corporation. Diferente das linguagens de programação modernas, que são compiladas para código nativo, a linguagem Java é compilada para um bytecode que é interpretado por uma máquina virtual (Java Virtual Machine, mais conhecida pela sua abreviação JVM).
* Framework Java Server Faces 2: é uma especificação Java para a construção de interfaces de usuário baseadas em componentes para aplicações web. Possui um modelo de programação dirigido a eventos, abstraindo os detalhes da manipulação dos eventos e organização dos componentes, permitindo que o programador se concentre na lógica da aplicação.
* Java Architecture for XML Binding: é um framework Java que permite o desenvolvimento de classes representados por XML.
* JavaScript: é uma linguagem de programação interpretada de alto nível, caracterizada também, como dinâmica, fracamente tipada, prototype-based e multi-paradigma.
  + 1. **Documentação de Suporte**

A documentação de apoio ao desenvolvimento da API segue anexa a este documento, contendo assim as informações mais gerais sobre o desenvolvimento e ferramentas necessárias para tal.

Há a necessidade também da documentação das ferramentas utilizadas, tais como a linguagem de programação, framework e ferramentas de apoio escolhidas.

É necessário, por fim, o conhecimento do framework Java Server Faces 2, utilizado para redação do código fonte do SIGAA.

Toda documentação necessária está inclusa nas referências do Manual de Elaboração da API.

* + 1. **Equipe e Suporte**

Após desenvolvida a API, a equipe de desenvolvedores do NTI pode prestar suporte e manutenção à API e ao Corpo Discente.

Quanto ao time de desenvolvimento, pode ser necessário suporte e apoio nas modificações a serem feitas no código fonte do SIGAA. Para tal, a UFRN pode prestar suporte ao NTI a depender do Termo de Execução Descentralizada firmado entre a UFPE e a UFRN.

1. **Recomendações**

Embora qualquer linguagem de programação possa ser para desenvolvimento da API, recomenda-se o uso da linguagem Java e do Framework JAXB para tal. O processo de escrita de serviços web RESTful através do dito framework possui um grande conjunto de ferramentas de suporte na IDE Netbeans.

1. **Treinamento**
   1. **Treinamento da Equipe de Desenvolvimento**

Se necessário, a equipe de desenvolvimento pode ser instruída pela UFRN quanto às mudanças a serem feitas no SIGAA.

Pode ser necessário, também, treinamento nas linguagens e frameworks escolhidos e, como suporte nesse processo, pode ser usada a documentação destas ferramentas, anexa ao Manual de Elaboração da API.

* 1. **Treinamento do Usuário**

Uma vez que a API será um serviço isolado da plataforma de importação, sendo um serviço de puro back-end, não há a necessidade de treinamento tão aprofundado por parte do usuário final. Todo treinamento necessário refletirá apenas nas mudanças feitas na plataforma de importação atual para adequar-se a implementação da API.

1. **Aprovações do Cliente**
   1. **Kick-off**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Desenvoltura nas interações** | **Entendimento dos Problemas** | **Clareza na apresentação** | **Qualidade das soluções propostas** | **Nível de Planejamento** | **Total** | **Comentários** |
| 5 | 5 | 5 | 4 | 5 | 4,8 | Delineamento melhor da solução |

* 1. **1º Status Report**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Desenvoltura nas interações** | **Entendimento dos Problemas** | **Clareza na apresentação** | **Qualidade das soluções propostas** | **Nível de Planejamento** | **Total** | **Comentários** |
| 5 | 5 | 3,5 | 3 | 4 | 4,1 | o REBOOT da solução não ficou claro. Se não vai ser um produto de software que vai ser entregue, o que vai ser? Quais as características? Que formato esta solução terá? Quais ganhos esta trará para a UFPE? |

* 1. **2º Status Report**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Desenvoltura nas interações** | **Entendimento dos Problemas** | **Clareza na apresentação** | **Qualidade das soluções propostas** | **Nível de Planejamento** | **Total** | **Comentários** |
| 5 | 5 | 5 | 4,5 | 5 | 4,9 | Senti falta do apontamento da solução que conversamos. a proposta de implementação de API de integração no SIGAA |

1. **Conclusão**

**Contatos:**

**Antônio Paulino -** [**apln2@cin.ufpe.br**](mailto:apln2@cin.ufpe.br)

**Gabriel Cavacalnti -** [**gcm2@cin.ufpe.br**](mailto:gcm2@cin.ufpe.br)

**Isac Tomaz -** [**its@cin.ufpe.br**](mailto:its@cin.ufpe.br)

**Pedro Rodolfo -** [**rodolfo@cin.ufpe.br**](mailto:rodolfo@cin.ufpe.br)

**Tiago Oliveira - tos@cin.ufpe.br**

1. **APÊNDICE: Manual de Implantação Técnico**

**CENTRO DE INFORMÁTICA / UNIVERSIDADE FEDERAL DE PERNAMBUCO**

**BACHARELADO EM SISTEMAS DE INFORMAÇÃO**

ANTÔNIO PAULINO DE LIMA NETO

GABRIEL CAVALCANTI DE MELO

ISAC TOMAZ DA SILVA

PEDRO RODOLFO GOMES DE SOUZA

**MANUAL DA API DE INTEGRAÇÃO**

SiSU - SIGAA

RECIFE

2019ÍNDICE

[**INTRODUÇÃO**](#_9ftpcocxndes) **2**

[OBJETIVOS](#_mqfo92hd7p1g) 2

[**SIGAA**](#_u5taflem7z8v) **3**

[ARQUITETURA MVC](#_s7130y8a0q1w) 3

[MANAGED BEANS](#_z8wzq6mtqtuq) 3

[MÉTODOS DE IMPORTAÇÃO](#_798db017mh38) 4

[**PLATAFORMA DE IMPORTAÇÃO**](#_no8z6fg2ioet) **4**

[DOCUMENTO DE SAÍDA](#_jjoj969lkd33) 4

[**FUNCIONAMENTO DA API COMO SOFTWARE ISOLADO**](#_nqjlmngdvlcj) **5**

[VIEWS](#_742v6o58z2f0) 5

[TRATAMENTOS](#_f3a0db7kcf20) 6

[RECOMENDAÇÕES](#_ees5pscad41z) 6

[VANTAGENS E DESVANTAGENS](#_ccduv8eaglkx) 6

[**FUNCIONAMENTO DA API COMO SERVIÇO WEB RESTFUL**](#_oyz7pixk66hb) **6**

[INTEGRAÇÃO COM O SIGAA](#_gmru8xrvyd22) 7

[INTEGRAÇÃO COM A PLATAFORMA DE IMPORTAÇÃO](#_ap1bqoph9qfk) 7

[SEM-ESTADO](#_hyunf2ocld0j) 7

[CACHE](#_uakjdalmuck4) 7

[DIVISÃO EM CAMADAS](#_hmk5hnuc21hh) 7

[CÓDIGO EM DEMANDA](#_l2bf8mum2tb8) 8

[OPERAÇÕES BÁSICAS](#_hij7i4y85tqe) 8

[JAVASCRIPT OBJECT NOTATION](#_lymveifm6ipu) 8

[MÉTODOS RECOMENDADOS](#_ehbr3li689cy) 9

[matricular\_aluno](#_d9f5ugpnsh9x) 9

[comprovante\_de\_matricula](#_ir7yevl7yns5) 9

[reclassificar\_aluno](#_ir7yevl7yns5) 9

[cancelar\_matricula](#_ir7yevl7yns5) 9

[**MÉTODOS AUXILIARES**](#_l4bm8268b5rs) **9**

[convocar\_aluno](#_8wlqomxeov81) 9

[associar\_matriz\_curricular](#_8wlqomxeov81) 10

[RECOMENDAÇÕES](#_li7l3y7wodtp) 10

[REQUISITOS](#_k4rl6saktqcg) 10

[**REFERÊNCIAS**](#_9vm2j5e8n8m) **11**

# INTRODUÇÃO

Este documento tem por objetivo facilitar o processo de integração entre a plataforma de importação do SiSU e do SIGAA.

## OBJETIVOS

A API de integração SiSU-SIGAA tem por objetivo facilitar a matrícula dos alunos ingressantes, automatizando este processo através de criação direta entre as plataformas de controle de candidatos e o portal SIGAA.

# SIGAA

O SIGAA é uma plataforma web que funciona a partir das especificações do JSF e, por tanto, uma aplicação MVC. Embora as especificações desta aplicação tenham grande influência na elaboração da API, este documento não tem por objetivo entrar em detalhes sobre estas especificações, sendo o foco principal a elaboração da API de comunicação entre o SIGAA e a plataforma de Importação do SiSU.

## ARQUITETURA MVC

O modelo MVC é um modelo de arquitetura de software de três camadas que divide o software entre Modelos, Visões e Controladores.

O papel do modelo é a manipulação do banco de dados da aplicação, realizando as operações de CRUD e gerindo o formato geral da informação que compõe a aplicação.

O controlador, por sua vez, implementa as regras de negócio que regem a aplicação, controlando o acesso das informações e operações a pessoas autorizadas. A plataforma em questão (SIGAA) usa controladores Managed Bean.

Já a Visão tem como papel apresentar a informação e toda a plataforma de interação com o usuário.

## MANAGED BEANS

Managed Beans funcionam como sementes do modelo que podem ser manipulados pelos controladores através das visões da aplicação. Isto é, ao usar um Managed Bean, o JSF aloca na memória do usuário um modelo essencialmente igual ao utilizado pela aplicação e, a partir deste modelo, realiza as operações implementadas no próprio Managed Bean como regras de negócios, tais como buscas e consultas, assim como as operações de CRUD, implementadas no modelo.

Managed Beans são classes Java que devem seguir duas regras especiais para seu correto funcionamento:

1. O método construtor do Managed Bean não pode receber argumentos;
2. O Managed Bean precisa expor seus atributos através de métodos *getters* e *setters*. Sendo estes declarados como privados (usando a keyword *private*).

Ao usar um formulário numa visão, por exemplo, o JSF usa o método *setter* em um campo de input associado a um dos atributos do Managed Bean e o método getter para recuperar e exibir informações salvas.

Um detalhe sintático a ser levado em consideração na manipulação de Managed Beans é que, embora o nome da classe deva ser escrito em CamelCase, as instâncias de Managed Beans precisam ser referidas em lowercase.

## MÉTODOS DE IMPORTAÇÃO

Após análise primária do trecho de código fonte referente ao módulo de graduação do SIGAA, percebe-se que a plataforma já possui os métodos e ferramentas necessárias à realização tanto de matrículas em massa de ingressantes na instituição, quanto à chamada e reclassificação de de alunos candidatos para a lista de espera.

Dito isto, cabe aos desenvolvedores da API conhecer e utilizar os métodos necessários e classes responsáveis pela execução desse processo. A API terá por objetivo, então, conectar a atual plataforma de importação do SiSU e a plataforma do SIGAA.

# PLATAFORMA DE IMPORTAÇÃO

Por já existir uma plataforma de importação onde o corpo discente realiza o controle e confirmação das matrículas de ingressantes e candidatos, este documento não entra em maiores detalhes, como necessidades técnicas da plataforma de importação. É feita apenas uma análise superficial do documento de saída desta plataforma, sendo este o necessário para as chamadas de métodos do módulo de graduação do SIGAA.

A partir desta plataforma há duas opções para implantação da API.

1. A API pode ser implementada como parte da plataforma de importação.

A plataforma de importação se torna parte do módulo de graduação do SIGAA, sendo a API parte desta plataforma.

1. A API como software isolado de integração entre as plataformas de importação e do SIGAA.

Com essa implementação a API funcionará como um serviço isolado, recebendo as saídas da plataforma de importação e fazendo chamadas aos métodos do módulo de graduação do SIGAA.

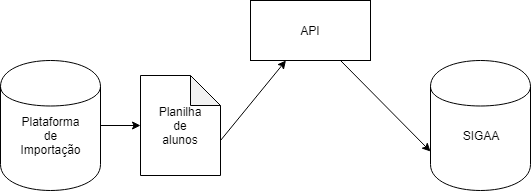
Em ambos os casos, os métodos da API podem seguir as especificações descritas neste documento.

## DOCUMENTO DE SAÍDA

O formato do documento de saída da plataforma de importação é importante a ser considerado. Este será o documento usado pela API para realizar os cadastros automáticos.

# FUNCIONAMENTO DA API COMO SOFTWARE ISOLADO

O objetivo da API é realizar as matrículas dos alunos no SIGAA a partir das saídas da plataforma de implantação e dos controladores do SIGAA. Uma vez que os ManagedBeans do JavaServerFace controlam a aplicação a partir da janela do navegador do usuário, fica inviável a implementação de uma API totalmente externa a plataforma do SIGAA, isto é, uma API independente ao SIGAA deve simular o acesso via navegador e acessar diretamente os campos gerados através da View dos ManagedBeans responsáveis pelo módulo de matrícula do SIGAA.

Esse tipo de comportamento pode ser alcançado através do uso de ferramentas de automação de browser, como Selenium, CasperJS e UniBot. Estas ferramentas simulam o navegador, permitindo o acesso aos campos gerados pelo ManagedBean para operação das regras de negócio pelas quais ele é responsável.

A figura acima demonstra o funcionamento da API na prática: os dados gerados pela plataforma de importação são tratados pela API que, usando um browser automático realiza o acesso ao SIGAA cadastra, convoca e reclasssifica os alunos.

## VIEWS

O trabalho da API pode ser resumido na visitação das páginas (views) geradas pelos ManagedBeans responsáveis pelas regras a serem implementadas em cada ponto. Isto é:

* Matrícula de Alunos (CadastramentoDiscenteMBean.java);
* Convocação de Alunos aprovados no SiSU (ConvocacaoAprovadosVestibularExternoMBean.java);
* Requalificação de Alunos (ConvocacaoVagasRemanescentesVestibularMBean.java e ConvocacaoMBean.java);
* E eventual declaração de matrícula (DeclaracaoVinculoMBean.java).

Cada ManagedBean é responsável por uma ou mais views e, por tanto, a API deve ser capaz de acessar todas as views por quais os ManagedBeans supracitados são responsáveis.

## TRATAMENTOS

Por fim, a API deve ser capaz de receber o número correto de parâmetros especificados e preencher os campos corretamente na interface JSP gerada por cada um dos Beans.

## RECOMENDAÇÕES

Embora a API possa ser escrita em qualquer linguagem, recomenda-se o uso da ferramenta CasperJS, utilizando a linguagem JavaScript. Escrita em JavaScript, a API pode ser adicionada como um módulo da plataforma de importação e ser ativada através de comandos na própria plataforma, uma vez que a API será também parte desta.

A escrita em uma outra linguagem, como Python ou Java, implica na implementação da API como serviço isolado. Isto é, a API deve ser acessada separadamente da plataforma de implantação, utilizando do documento resultante da plataforma de importação do SiSU.

## VANTAGENS E DESVANTAGENS

Como um software isolado da plataforma do SIGAA a API não não precisa interagir com o código fonte do SIGAA de nenhuma forma, as requisições são feitas a partir da simulação do browser, o que não necessita de mudanças no SIGAA. Essa implementação pode ser adotada, também, como solução paliativa para a integração da plataforma de importação com o SIG@, sendo de implementação mais simples e fácil.

Esta implementação, no entanto, não atende aos princípios de uma API RESTful e acaba por ter uma implementação que demande maior poder computacional.

# FUNCIONAMENTO DA API COMO SERVIÇO WEB RESTFUL

A alternativa da implementação da API como um sistema RESTful demanda algumas modificações no código fonte do SIGAA. Uma vez que o SIGAA é construído usando controladores Managed Bean do Java Server Face, a integração do sistema com uma API RESTful não poderá depender dos controladores. As requisições da plataforma devem ser feitas diretamente aos DAOs do SIGAA.

Essa implementação demanda um cuidado maior na construção da API, uma vez que os DAOs são responsáveis diretos pela manipulação do banco de dados do SIGAA. Escrita desta forma, fica a recomendação de que a API seja um serviço isolado da plataforma de importação.

Uma vez finalizada uma matrícula na plataforma de importação, uma requisição POST é feita pela plataforma para a API, contendo os dados referentes ao estudante matriculado. Estes dados são então enviados da API para o DAO responsável pelo cadastro de um novo aluno no SIGAA. Por sua vez o SIGAA responde a API com o estado da solicitação, isto é, uma resposta HTTP de status sobre o sucesso da operação realizada. A API então encaminha essa resposta para a plataforma de importação que deve, então, disponibilizar a resposta para o usuário.

## INTEGRAÇÃO COM O SIGAA

Para uma correta integração com o SIGAA, a API deve ser capaz de realizar requisições HTTP para a plataforma que, por sua vez, deve ser capaz de responder a essas requisições através dos códigos de status.

De acordo com a documentação do Java Server Face 2 e da NetBeans IDE, recomenda-se o uso do framework JAXB para criação das classes abstratas a serem estendidas pelos DAOs responsáveis pelo módulo de graduação.

## INTEGRAÇÃO COM A PLATAFORMA DE IMPORTAÇÃO

A comunicação entre a plataforma e a API se dá através de protocolo de requisição e resposta HTTP, atendendo aos princípios arquiteturais REST.

### SEM-ESTADO

A API não precisará guardar informações sobre requisições HTTP anteriores, tratando cada requisição feita como nova. Por se tratar do módulo de matrículas, é necessária confirmação da identidade e autorização do funcionário responsável por efetuar a matrícula em questão no SIGAA e, por tanto, cada requisição deve conter as informações necessárias à autenticação autorização do funcionário.

### CACHE

Uma vez que a API interage com a plataforma de importação, há pouca, se alguma, necessidade de armazenamento de recursos em cache do usuário por parte da API.

### DIVISÃO EM CAMADAS

Pela natureza da API a divisão em camadas acontece com a implementação da mesma. Isto é, a API passa a se comportar como a segunda camada de uma arquitetura com três camadas de sistema, onde a primeira é a plataforma de importação, a segunda é a própria API e a terceira é o SIGAA.

Tanto a autenticação quanto persistências dos dados enviados através da API são feitas na plataforma do SIGAA, sendo esta a camada final e mais importante do sistema.

### CÓDIGO EM DEMANDA

Por fim, não é necessário que a API seja capaz de retornar ao cliente trechos de código, uma vez que a função principal da API é de manipular dados estáticos.

## OPERAÇÕES BÁSICAS

As operações básicas de matrícula, leitura, exclusão e atualização são feitas de acordo com a convenção de APIs REST e métodos HTTP, assim como as suas respostas.

* **POST:** Ao solicitar uma matrícula, a plataforma de importação deve se comunicar com a API através de um JSON contendo as informações necessárias para a efetiva matrícula do aluno no SIGAA em uma requisição POST.
* **PATCH:** Ao solicitar a alteração de um cadastro previamente efetuado, a plataforma envia para a API uma requisição PATCH contendo um JSON com as informações a serem atualizadas e identificação do cadastro a sofrer alterações.
* **GET**: Ao solicitar algum dado, como por exemplo o comprovante de matrícula do aluno, a requisição é do tipo GET e o JSON tem a identificação do aluno cujo comprovante de matrícula fora solicitado.
* **DELETE**: Por fim, o cancelamento da matrícula de um aluno desistente é feito através de uma requisição do tipo DELETE com o JSON com a identificação do aluno cuja matrícula será cancelada.

A depender da tecnologia utilizada na construção da API pode ser necessário sobrescrever métodos HTTP. Para isso, a API pode ter acesso ao cabeçalho customizado *X-HTTP-Method-Override*.

## JAVASCRIPT OBJECT NOTATION

As informações necessárias para o processamento de qualquer operação são providas a API pela plataforma de importação através de JSONs. A API considera apenas os campos do objeto relevantes a operação solicitada, ignorando demais propriedades e dados possivelmente fornecidos.

Não é necessário identificar no cabeçalho do JSON o tipo de operação executada. A ação é definida pelo método HTTP associado à tarefa.

Por fim, as respostas da API para o usuário são também enviadas através de JSON. Essas respostas podem ser de dois tipos:

* **Sucesso**: em caso de sucesso o cabeçalho do JSON de resposta deve conter o código de sucesso específico a operação no formato 2xx.
* **Falha**: em caso de falha, o cabeçalho contém o código de erro 4xx e 5xx associado ao impedimento da operação.

## MÉTODOS RECOMENDADOS

Como sugestão a nomenclatura dos métodos da API estão as regras de boas práticas de API RESTful anexas a este documento. No entanto, estão abaixo as sugestões e descrição de métodos essenciais ao funcionamento da API proposta.

|  |
| --- |
| *matricular\_aluno* Parâmetros: \*Dados do aluno  Método HTTP: POST  Esse é o principal método para o funcionamento da API. Os parâmetros fornecidos para a correta execução desse método são os dados necessários à matrícula do aluno no SIGAA. Através deste, a API se comunica com o DAO associado ao aluno, criando um novo registro de *discente\_graduacao* no banco de dados do SIGAA.  Resposta: a API comunica ao usuário o estado da requisição (sucesso ou falha) com seus respectivos códigos associados e mensagens do sistema. |
| *comprovante\_de\_matricula* Parâmetros: identificação do aluno  Método HTTP: GET  Esse método retorna o link para o comprovante de matrícula ou grade de horários do aluno.  Resposta: link ou arquivo da grade de horários, quando bem sucedida. |
| *reclassificar\_aluno* Parâmetros: identificação do aluno  Método HTTP: PATCH  Esse método é chamado quando um aluno da segunda entrada é reclassificado para a primeira. |
| *cancelar\_matricula* Parâmetros: identificação do aluno  Método HTTP: DELETE  Esse é o método responsável pelo cancelamento da matrícula de um aluno desistente. |

## MÉTODOS AUXILIARES

|  |
| --- |
| *convocar\_aluno* Parâmetros: identificação do aluno  Método HTTP: GET  Esse é o método auxiliar que realiza a convocação do aluno aprovado, isto é, a partir desse método, quando o aluno é matriculado ou reclassificado, um e-mail é enviado para o mesmo informando de sua situação.  O SIGAA fornece os métodos necessários para convocação via e-mail, bastando a chamada de método a seguir.  Respostas: A resposta vai depender do estado do servidor e da identificação do aluno. Sucesso no caso do servidor ter respondido e o aluno ter sido encontrado, logo o método de envio de e-mail do SIGAA foi chamado apropriadamente. Falha em qualquer outro caso. |
| *associar\_matriz\_curricular* Parâmetros: Identificação do aluno, identificação da matriz curricular  Método HTTP: POST  O SIGAA já possui o método necessário a associação de matriz curricular a um discente, gerando uma grade de horários de primeiro período. A API precisa apenas solicitar a execução do método responsável, informando ao DAO os parâmetros necessários. |

## RECOMENDAÇÕES

Embora qualquer linguagem de programação possa ser para desenvolvimento da API, recomenda-se o uso da linguagem Java e do Framework JAXB para tal. O processo de escrita de serviços web RESTful através do dito framework possui um grande conjunto de ferramentas de suporte na IDE Netbeans.

## REQUISITOS

Embora possa, a princípio, parecer necessária a hospedagem da API em um servidor externo para solicitação através da plataforma de importação, existe a possibilidade da API funcionar como um módulo isolado do SIGAA, não interferindo no código fonte do mesmo. Dessa forma, descarta-se a necessidade de hardware isolado para hospedagem da API uma vez que a demanda pela API é ainda menor que a demanda de acesso do SIGAA e que, tal demanda, acontece apenas durante o período de matrícula e modificação de matrículas.

## 

# REFERÊNCIAS

<https://docs.oracle.com/cd/E17802_01/j2ee/javaee/javaserverfaces/2.0/docs/api/>

<https://netbeans.org/kb/docs/web/jsf20-intro_pt_BR.html>

<https://javabeat.net/jsf-2-managed-beans/>

<http://casperjs.org/>

<https://www.seleniumhq.org/>

<https://unibot.ai/>

<https://restfulapi.net/rest-architectural-constraints/>

<https://www.ibm.com/support/knowledgecenter/pt-br/SS4GCC_6.1.1/com.ibm.urelease.doc/topics/rest_api_ref_conventions.html>

<https://www.w3.org/Protocols/HTTP/HTRESP.html>

<https://blog.mwaysolutions.com/2014/06/05/10-best-practices-for-better-restful-api/>