Termo de Cooperação/Projeto:

Termo de Cooperação Técnica FUB/FUNAPE e PGFN

Desenvolvimento e inovação visam realizar um estudo de inovações tecnológicas, nas áreas de tecnologia da informação, com ênfase na interoperabilidade de sistemas, gestão corporativa, processos de gestão, gerência de redes e gestão de dados, para áreas estratégicas da Coordenação-Geral de Tecnologia da Informação (CGTI) da Procuradoria Geral da Fazenda Nacional – PGFN.

Documento:

RT de Arquitetura de integração e interoperação - Parcial

Data de Emissão:

01/02/2023

Elaborado por:

Universidade de Brasília – UnB
FUNAPE - Fundação de Apoio à Pesquisa - UFG
Laboratório de Tecnologias da Tomada de Decisão – LATITUDE.UnB

Digital Object Identifier:

10.5281/zenodo.7779002

PROCURADORIA-GERAL DA FAZENDA NACIONAL – PGFN

Ricardo Soriano de Alencar
Procurador-Geral da Fazenda Nacional

Daniel de Saboia Xavier
Procurador da Fazenda Nacional

José Renato Fragoso Lobo
Coordenador-Geral de Gestão de Pessoas e Desenvolvimento Institucional

Aleksey Lanter Cardoso
Diretor de Gestão Corporativa

Rodrigo Otávio Povoa Pullen Parente
Coordenador-Geral de TI

EQUIPE TÉCNICA

Alan Zuanella Vila Nova
Alex Aranda
Fernando Maciel Lima e Sousa
Hiago Pereira Kanashiro
Julio Sergio Fernandes Alves
Leandro Veloso Rodrigues
Priscila Fatima Pinheiro de Siqueira
Rosiane Aparecida Moreira
Thiago Oliveira Hoerlle
Equipe da Algar TI Consultoria S/A

UNIVERSIDADE DE BRASÍLIA

Márcia Abrahão Moura
Reitora

Prof.ª. Maria Emília Machado Telles Walter
Decana em exercício
Decanato de Pesquisa e Inovação – DPI

Rafael Timóteo de Sousa Júnior
Coordenador do Laboratório de Tecnologias da Tomada de Decisão – LATITUDE

EQUIPE TÉCNICA

Rafael Timóteo de Sousa Júnior (Pesquisador Sênior)
Fábio Lúcio Lopes de Mendonça (Pesquisador Sênior)
Georges Daniel Amvame Nze (Pesquisador Sênior)
Robson de Oliveira Albuquerque (Pesquisador Sênior)
William Ferreira Giozza (Pesquisador Sênior)
Alex Mendes Pacheco da Silva
Allan Filipe Almeida
Ana Beatrice Neubauer de Moura
Ana Paula Bernadi da Silva
Ana Paula Morais Vale
Ayra de Avila Almeida
Caio Henrique Caetano
Carlos Eduardo Ramalho de Souza
Cleuber Santos Silva
Diego Martins de Oliveira
Felipe Barreto de Oliveira
Flavio Garcia Praciano
Gabriel Ribeiro de Araújo
Guilherme Batista Meneses Alves
Hiago Pereira Kanashiro
Isaac Silva Martins
Ismael Ithalo Barbosa Pinto
João Paulo da Costa e Silva Garcia
Joao Paulo Pimentel
Johnan Nicholas Reed
Klainer Mateus Estrela Gomes
Kelly Santos de Oliveira Bezerra
Leonardo Jorge França Morais
Letícia Ramos Reinaldi
Lucas da Silva Barbosa
Luiz Augusto dos Santos Pires
Marcos Tércio Ramos
Maria Karoline Domingues
Maria Tereza Correa Pacheco Alves
Marilia do Nascimento T. Valentim
Orlando Werbeth dos Santos Gomes
Paulo Henrique Batista Rodrigues

Paulo Lima Machado
Phelipe Alan Almeida
Priscila Batista Rodrigues
Renato Jose da Silva Camoes
Thiago Leite de Souza
Valeria Simas Schultz
Wellington Domingos Neves

HISTÓRICO DE REVISÕES

Data	Versão	Descrição
31/01/2022	1.0	Criação do documento técnico.
15/06/2022	1.1	Atualização do documento técnico.
27/01/2022	1.2	Atualização do documento técnico.
01/02/2023	1.3	Atualização do documento técnico.

Universidade de Brasília – UnB
Campus Universitário Darcy Ribeiro - FT – ENE – Latitude
CEP 70.910-900 – Brasília-DF
Tel.: +55 61 3107-5598 – Fax: +55 61 3107-5590

LISTA DE SIGLAS

PGFN	Procuradoria Geral da Fazenda Nacional
TED	Termo de Execução Descentralizada
TIC	Tecnologias de informação e comunicação
RT	Relatórios técnicos
UnB	Universidade de Brasília
PPGEE	Programa de Pós-Graduação em Engenharia Elétrica
TED	Termo de Execução Descentralizada
FUB	Fundação Universidade de Brasília

SUMARIO

1. INTRODUÇÃO

Com base na Lei nº 2.642, de 9 de novembro de 1955, houve a criação da Procuradoria-Geral da Fazenda Nacional – PGFN, na forma atualmente conhecida, em substituição à Procuradoria-Geral da Fazenda Pública. Instituída como órgão de consultoria jurídica do Ministério da Fazenda, à PGFN era atribuída, principalmente, examinar e fiscalizar os contratos de interesse da União, apurar e inscrever a dívida ativa federal para fins de cobrança judicial e cooperar com o Ministério Público da União junto à justiça comum (art. 1º).

O Decreto-Lei nº 147, de 3 de fevereiro de 1967, estabeleceu a segunda lei orgânica da PGFN. Esse diploma legislativo fixou competências até hoje mantidas pelos demais atos normativos que o sucederam, na mesma direção do que previa a Lei nº 2.642, de 1955, estabelecendo o seguinte: a) a vinculação administrativa da PGFN como órgão do Ministério da Fazenda responsável pelo prestação de serviços jurídicos da Pasta; b) a atribuição de apurar e inscrever, para fins de cobrança judicial, a dívida ativa da União, tributária ou de qualquer outra natureza; c) e sua atuação nacional por força da descentralização do órgão.

Com a promulgação da Constituição da República de 1988, houve uma mudança significativa da Procuradoria-Geral da Fazenda Nacional quanto a sua vinculação exclusiva ao Ministério da Fazenda. A PGFN passou a integrar a nascente Advocacia-Geral da União, órgão criado para defender, judicial ou extrajudicialmente, os interesses da União.

A Lei Complementar nº 73, de 10 de fevereiro de 1993, que institui a Lei Orgânica da Advocacia-Geral da União, previu, expressamente, a subordinação técnica e jurídica da PGFN ao Advogado-Geral da União, confirmando a finalidade do legislador constituinte em vincular a Procuradoria como órgão da PGFN responsável pela atuação na área fiscal.

Com isso, a PGFN tornou-se órgão de direção superior da Advocacia-Geral da União e suas atribuições residem, principalmente, na representação da União em causas fiscais, na cobrança judicial e administrativa dos créditos tributários e não-tributários e no assessoramento e consultoria no âmbito do Ministério da Fazenda onde atualmente encontra-se no âmbito do Ministério da Economia.

De outro lado, temos o Laboratório de Tecnologias da Tomada de Decisão – LATITUDE, que foi criado em 2010 com recursos da Lei de Informática provenientes da DELL Computadores do Brasil, constituído como um ambiente de inovação e

desenvolvimento para projetos de pesquisa interdisciplinar entre as engenharias, a computação, a ciência da informação, bem como os demais domínios do conhecimento de interesse para a temática focal da tomada de decisão.

Criado no âmbito do Departamento de Engenharia Elétrica da Universidade de Brasília - UnB, onde tem seu espaço físico próprio, o Laboratório LATITUDE é vinculado ao Programa de Pós-Graduação em Engenharia Elétrica – PPGEE e ao Programa de Pós-Graduação Profissional em Engenharia Elétrica - Segurança Cibernética, contando também com professores, pesquisadores e estudantes de Programas de Pós-Graduação das áreas de Engenharia Civil, Engenharia Computação, Engenharia da Produção, Engenharia da Automação, Engenharia de Software, Ciência da Computação, Ciência da Informação, Administração, Direito, Educação, Psicologia, História, bem como dos respectivos cursos de graduação, nos Campi Darcy Ribeiro e Gama.

A PGFN vem buscando, gradativamente, desenvolver soluções para sanar parte dos problemas existentes nas áreas de tecnologia da informação, administração interna, gestão corporativa, pessoal e documental. Para isso foram realizados estudos em diversas áreas da PGFN, realizando um levantamento dos problemas e sua descrição.

No que se relaciona às tecnologias da informação e das comunicações, é necessário trazer inovações tecnológicas aos processos e sistemas administrativos que se integram aos sistemas estruturantes (Sistemas SIDA, SAJ, Flexa, SISPAR, Regularize, Dívida, FGTS) de forma que a PGFN consiga atender às suas demandas de forma rápida e eficiente, fornecendo uma plataforma inteligente que auxilie tanto o acesso às diversas pesquisas internas e externas realizadas como à tomada de decisão. Entende-se que esse aprimoramento envolve o sistema informacional, que organize a informação e banco de dados customizado, de acordo com necessidades das áreas de negócio envolvidas, procurando as seguintes resultantes: identificar, mitigar e tratar riscos, quantificar a ocorrência de fenômenos, mapear correspondentes processos. Além do acompanhamento contínuo das informações, com aprimoramento das análises administrativas e das defesas apresentadas. Ante o exposto, complementa-se que a informação é hoje um dos patrimônios mais importantes de uma organização, seja ela pública ou privada. As tecnologias da informação e comunicações (TIC) se consolidam como ativo estratégico, onde integra recursos, processos, métodos, técnicas para obter, processar, armazenar, disseminar e fazer uso da informação. Sob essa ótica, a governança de TIC garante a boa e regular gestão dos serviços de TIC, que se desdobram ao encontro da estratégia

corporativa do órgão. Entretanto, esse alinhamento só é viável com a estruturação de um planejamento que reflita como a TIC contribuirá, através do alcance das suas metas e ações, para o alcance dos objetivos organizacionais.

Assim, é pertinente e relevante evoluir a maturidade das tecnologias utilizadas em tais sistemas de informação, assim como dos processos de gestão e governança associados. Em especial, coloca-se a necessidade de interoperação entre sistemas, assim como a preparação para a interoperação com outros sistemas que deverão ser concebidos, desenvolvidos e operacionalizados. Tal necessidade precisa de soluções inovadoras no que se refere à semântica da informação e à algorítmica de operação simultânea e paralela de módulos de sistemas interdependentes, sejam eles internos à PGFN, sejam sistemas externos que necessitem de serviços ou informações dos sistemas da PGFN. Colocam-se nesse contexto as questões de confidencialidade, integridade e disponibilidade das informações e serviços, o que implica no requisito de agregar uma abordagem de segurança da informação às atividades de gestão de sistemas de informação da PGFN. O próprio planejamento diretor desses sistemas e das respectivas tecnologias de suporte (armazenamento, processamento, rede, acesso etc.) merece estudos que levem a seu aprimoramento e sua colocação em um processo de governança que contribuam para uma melhoria continuada da maturidade da PGFN em tais domínios de tecnologias da informação.

2. VERSIONAMENTO DE PROJETOS

O versionamento de projeto é uma prática essencial quando o desenvolvimento envolve mais de uma pessoa. Isso permite que as contribuições de cada colaborador sejam registradas e gerenciadas de maneira eficiente.

Além disso, o versionamento permite que o projeto seja trabalhado de forma colaborativa, ou seja, as pessoas podem trabalhar de forma simultânea no projeto, sem prejudicar o trabalho uns dos outros. Isso aumenta a produtividade e garante a qualidade do projeto final.

3. CONTROLE DE VERSÃO

O Controle de versão é uma ferramenta fundamental para equipes de desenvolvimento de software, pois permite gerenciar e rastrear as alterações em um projeto. Além disso, ele também permite trabalhar em equipe de forma colaborativa, permitindo que múltiplos desenvolvedores possam trabalhar no mesmo arquivo ou conjunto de arquivos ao mesmo tempo, sem correr o risco de sobrepor alterações uns dos outros.

Com um controle de versão, é possível manter uma história completa do desenvolvimento do projeto, incluindo todas as modificações realizadas em cada arquivo, o autor de cada modificação, e a data em que foi realizada. Isso permite que o time de desenvolvimento reverta para versões anteriores se precisar, compare as diferenças entre as versões, e identifique rapidamente problemas ou erros.

Além disso, um controle de versão também oferece segurança e garantia de continuidade do projeto, pois permite fazer backup de todas as versões e garantir a integridade dos arquivos. Em caso de perda de dados, é possível restaurar uma versão anterior do projeto.

Em resumo, o controle de versão é uma ferramenta essencial para equipes de desenvolvimento de software, pois permite colaboração, gerenciamento, rastreabilidade e segurança no desenvolvimento de projetos.

4. FLUXO DE VERSIONAMENTO

Trabalhando em equipe ou sozinho, usar um padrão é, primeiramente, de fundamental importância, pois é algo definido pela gestão do projeto e que todos devem seguir, para que não haja conflito ou uma desordem. Além disso, usar um padrão reconhecido e principalmente documentado facilita, para cada pessoa que entrar no projeto, visualizar ou contribuir para o seu projeto de forma a reconhecer aquele padrão e assim, diminuir a curva de aprendizado e ainda aumentar a produtividade.

Para alcançar este padrão, o versionamento inclui aspectos facilitadores, como a criação de *branches.* O fluxo adotado inclui quatro níveis de branches, que são:

- Central (*master*) - Este branch é a principal e contém o código em produção do sistema. Apenas o Analista de Configuração terá permissão para escrever nesse branch;
- Intermediária (desenvolvimento) - se trata da cópia do *branch master* e será

disponibilizada aos desenvolvedores de acordo com o tipo de demanda. Será a versão mais atualizada até o momento de enviar para o master. Entretanto, esses branches não serão disponibilizados aos desenvolvedores;

- Desenvolvimento remoto (<numerodaissueredmine>) - *branch* criado a partir do *branch* desenvolvimento, onde será criado com o nome da demanda aberta no *redmine*;
- Desenvolvimento Local (<numerodaissueredmine>) - *branch* criado localmente no computador do desenvolvedor e deverá ser enviado diariamente para o repositório remoto para minimizar o risco de perda do código trabalhado.

4.1 BRANCHES PRINCIPAIS

O repositório central possui dois *branches* principais com vida infinita.

- *Master*;
- Desenvolvimento.

O *branch master* é o *branch* principal, a *HEAD* do projeto, nele há somente versões que estão em produção.

O *branch* de desenvolvimento possui todo código já entregue e as últimas versões em desenvolvimento para a próxima versão. Quando o código do *branch* desenvolvimento é considerado estável e pronto para ser implantado, todas as alterações devem ser mescladas de volta para o *branch master* e criada uma *tag*. No item a seguir são apresentados mais detalhes de como isso é feito.

4.2 BRANCHES DE SUPORTE

Junto aos *branches master* e desenvolvimento foram utilizados outros *branches* de suporte, para correção de erros, criação de melhorias e preparação para implantação. Diferente dos *branches* principais, esse tem uma vida limitada, uma vez que eles são removidos eventualmente.

Os diferentes tipos de *branches* a serem utilizados são:

- *Branches* de melhorias (*feature*);
- *Branches* de lançamento (*release*);

- *Branches* de correções (*hotfix*).

Cada tipo de *branch* tem um propósito específico e segue regras de quais *branches* devem ser originados e mesclados.

4.3 BRANCHES DE MELHORIAS OU EVOLUTIVAS

Devem ser criadas a partir da *Branch* de desenvolvimento e deve ser mesclado de volta para desenvolvimento, podendo ter *branches* intermediários.

Convenção de nome: <***Número da issue no redmine***>

Branches de melhorias são usados para desenvolver novas funcionalidades para o próximo lançamento. Em essência, um *branch* de melhoria existe apenas enquanto está em desenvolvimento, devendo ser mesclado ao *branch* desenvolvimento, assumindo que entrará no próximo lançamento, ou descartado caso não seja útil ou seja um experimento.

Para criar um branch de melhoria: ao iniciar o desenvolvimento de uma funcionalidade, criar um *branch* a partir do *branch* desenvolvimento:

```
$ git checkout -b feature/xpto desenvolvimento
```

4.3.1 Finalizando um *branch* de melhoria

Uma vez concluído o desenvolvimento no *branch*, ele deve ser incorporado de volta no *branch* desenvolvimento através de um *pull request.* Mas antes de incorporar deve ser criada uma *tag* para a melhoria ser testada.

Para enviar o *branch* para o servidor:

```
$ git push origin feature/xpto
```

Deve-se navegar até a interface do projeto no *GitLab* e clicar em *Repository* >*Tags* > *New tag,* criar a *tag* de teste utilizando a convenção de nome: <numerodaissueredmine_T_1>, para o primeiro teste, <numerodaissueredmine_T_2> caso a melhoria não passe pelo primeiro teste e assim por diante, até que a melhoria passe no teste. Com a melhoria aprovada pelo teste, o desenvolvedor deve verificar se o *branch* desenvolvimento já foi atualizado. Caso tenha sido atualizado o mesmo deve fazer um *merge* do desenvolvimento em sua *branch*, para assim realizar um *merge request* no desenvolvimento.

Deve-se navegar até a interface do projeto no *GitLab* e clicar em *new merge request*, selecionando como base sua branch e o branch desenvolvimento.

Uma vez aceito o *merge request* pelo administrador do projeto, será criado uma *tag* para essa versão ser posta em homologação, após homologado a melhoria entrará na próxima versão.

4.4 BRANCHES DE LANÇAMENTO

Deve ser criado a partir da *branch* desenvolvimento e deve ser mesclado na *branch* desenvolvimento e na *branch master.*

Convenção de nome: numerodaissueredmine1_numerodaissueredmine1

Exemplo: feature75_historicousuarios_75

Branches de lançamento são usados para preparação do lançamento da próxima versão de produção. Neles são permitidas pequenas correções e atualização de versão nos arquivos. Fazendo isso no *branch* de lançamento, o *branch* desenvolvimento fica livre para receber novas melhorias para a próxima versão.

Na criação do *branch* de lançamento é decidido qual versão o projeto terá, até este momento o *branch* desenvolvimento reflete as alterações da próxima versão, independente de qual for. Esta decisão é feita na criação do *branch* de lançamento e segue as convenções de versionamento do projeto.

4.4.1 Criando Um Branch De Lançamento

Branches de lançamento são criados a partir do *branch* desenvolvimento. Digamos que o projeto está na versão 1.5.3 e há uma implantação em breve. O estado do *branch* desenvolvimento está pronto para a próxima implantação e decidimos que vai se tornar 1.2.0 (ao invés de 1.1.5 ou 2.0.0). Então criamos um *branch* com o nome da versão que escolhemos.

```
$ git checkout -b release/1.2.0 desenvolvimentos
```

Depois de criar o *branch*, a primeira coisa a fazer é aumentar a versão no arquivo composer.json e comitar.

```
$ git commit -a -m "Atualização da aplicação para a versão 1.2.0"
```

Este *branch* deve existir por enquanto, até que esteja pronto para ser implantado definitivamente em produção. Pequenas correções são permitidas nele (ao invés do *branch* desenvolvimento). Adicionar funcionalidades novas nele são proibidas, apenas correções pontuais desta versão de lançamento.

4.4.2 Finalizando um branch de lançamento

Quando o *branch* de lançamento estiver pronto para ser implantado em produção, algumas ações precisam ser tomadas.

Primeiro o *branch* de lançamento é mesclado com o *branch master* (uma vez que cada commit no master é uma nova versão, por definição):

```
$ git checkout master
$ git merge --no-ff release/1.2.0
```

Em seguida este *commit* deve ser *tagueado* para referência futura para esta versão: $ git tag -a 1.2.0

Finalmente, as mudanças feitas no *branch* de lançamento precisam mescladas de volta no *branch* desenvolvimento, para que as versões futuras também possuam as correções feitas neste *branch*.

```
$ git checkout desenvolvimento
```

Neste momento podem ocorrer alguns conflitos, já que as correções podem mudar a versão dos arquivos, se acontecer, deve-se corrigir e efetuar *commit.*

Feitos os merges podemos excluir o *branch* de lançamento, já que não precisamos mais dele: $ git branch -d release/1.2.0

5. UTILIZANDO O GIT E GITLAB

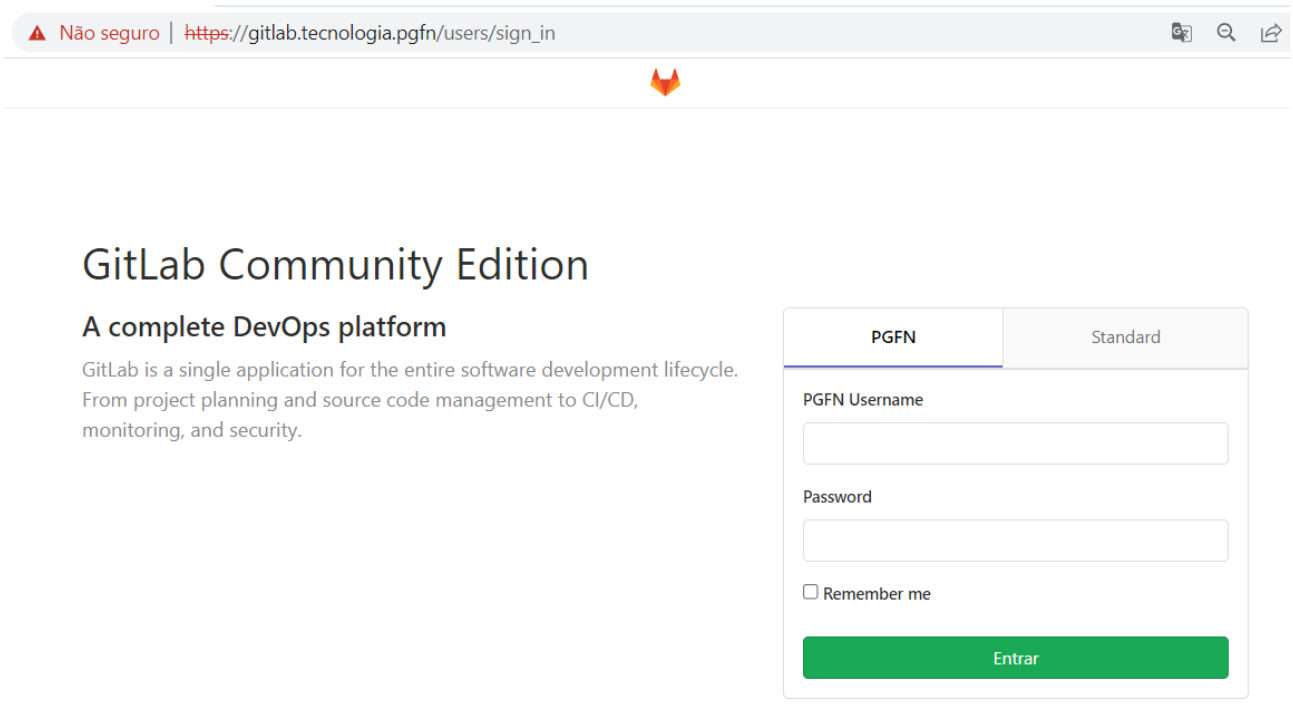
Para iniciar a utilização, o desenvolvedor precisa ter o conhecimento que o Gerenciador de Repositórios adotado pelo MDS é o GitLab, o qual pode ser acessado através do link: https://gitlab.com/pgfn

O acesso ao Gitlab deverá ser solicitado para o Analista de Configuração responsável pela manutenibilidade da ferramenta da CGTI.

Uma vez que o acesso foi liberado, o desenvolvedor deverá acessar o link do GitLab

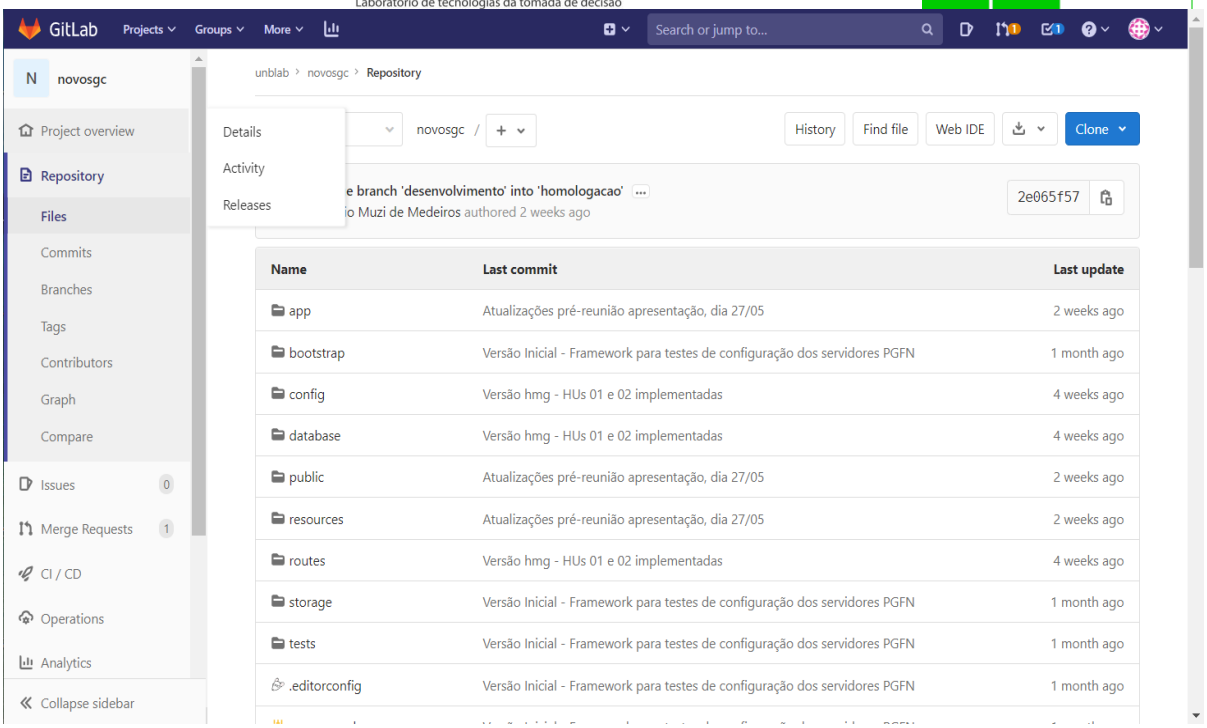
e se autenticar utilizando o login de rede (AD), conforme pode-se verificar na tela a seguir:

Figura 1 – Autenticação



Já autenticado, o desenvolvedor terá a visibilidade de todos os projetos ao qual foi vinculado previamente. Selecionando o projeto desejado, o desenvolvedor será direcionado para uma tela semelhante à tela a seguir:

Figura 2 – Projeto



Nesta tela é apresentado o link direto do projeto, o qual será utilizado para realizar o clone do mesmo. Deve-se clicar no ícone “Clone”, selecionar a opção “Link https” para copiar o link para a área de transferência.

Tendo o link copiado, pode-se iniciar os procedimentos de clonagem no computador local, utilizando o *Git Bash*. Antes de realizar a clonagem, é necessário definir as configurações locais do computador.

5.1 CONFIGURAÇÕES GLOBAIS

No *prompt* de comandos do *Git Bash*, deve-se configurar o nome e e-mail do usuário conforme mostrado a seguir:

$git config --global user.name "Nome Sobrenome"

$git config --global user.email “nome.sobrenome@email.com.br”

Estas definições servirão para identificar o usuário responsável por qualquer alteração realizada no repositório.

5.2 OBTENDO UM REPOSITÓRIO

Para se obter o projeto, deve-se utilizar o link copiado para a área de

transferência anteriormente, e cloná-lo no computador local com o comando "git clone". Conforme pode-se verificar a seguir:

$git clone https://gitlab.tecnologia.pgfn/unblab/novosgc.git

Uma vez que o repositório foi clonado, este já se encontra pronto para utilização.

Ao baixar a aplicação NSGC do Git (link informado no último passo da da instalação do Laravel), na raiz do projeto o arquivo “.env.exemple” deverá ser renomeado para “.env”.

Em seguida, abra o arquivo e altere os parâmetros abaixo, referente aos bancos de dados, com as informações do servidor da PGFN:

DB_HOST
DB_PORT
DB_DATABASE
DB_USERNAME
DB_PASSWORD
DB_PASSWORD_SALT

Diretórios

A estrutura de diretório do Laravel e do Angular são descritas, em detalhes, suas respectivas pághinas oficiais.

Laravel:
https://laravel.com/docs/8.x/structure

6. CONCLUSÃO

Através de um trabalho coordenado e interdependente entre as equipes da Procuradoria Geral da Fazenda Nacional (PGFN) e da Universidade de Brasília (UNB), as atividades de elaboração deste RT foram planejadas, discutidas, executadas e documentadas.

A conclusão do RT é importante pois cumpre a etapa básica de inicialização, prevista na metodologia de gestão do projeto. O plano define a estratégia inicial adotada, servindo de referencial para adaptações dessa estratégia caso a monitoração do projeto assim o determine.

Além de conter as decisões iniciais da gestão do projeto, o presente relatório servirá então, durante as demais etapas, para registrar as alternativas e embasar a tomada de decisão para prosseguimento do projeto.

As atividades envolvidas nesta etapa observaram formalmente à execução dos passos da metodologia elencada para gestão do projeto, conforme definido pelo PMI em seu guia PMBoK.

A equipe da UnB considera que teve acesso a todas as informações necessárias à boa condução dos trabalhos e que a disponibilização dessas informações pela equipe do MDSA, assim como as atividades conjuntas de análise e discussão levaram à etapa do projeto a bom termo.

7. REFERÊNCIAS BIBLIOGRÁFICAS

Nayan B. Ruparelia. 2010. **The history of version control. SIGSOFT Softw. Eng**. Notes 35, 1 (January 2010), 5-9. DOI=http://dx.doi.org/10.1145/1668862.1668876;

D. Spinellis, "Version control systems," in IEEE Software, vol. 22, no. 5, pp. 108-109, Sept.-Oct. 2005. doi: 10.1109/MS.2005.140, disponível em: : http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1504674&isnumber=32260

Universidade de Brasília – UnB

FUNAPE - Fundação de Apoio à Pesquisa - UFG

Laboratório de Tecnologias da Tomada de Decisão – LATITUDE

www.unb.br – https://funape.org.br – www.latitude.unb.br