

Manual de utilização do PIPA

Autor(es): Júnia Maísa Oliveira, Emanuella Ferraz, Vinícius Rodrigues Oliveira, Daniel Fernandes Macedo, Dorgival Olavo Guedes Neto, José Marcos Silva Nogueira.

Este manual pertence ao artigo “PIPA: Uma solução integradora de políticas de acesso a recursos” do Salão de Ferramentas do SBRC 2023. Nele é descrita a utilização da Plataforma Integrada de Políticas de Acesso (PIPA) e é acompanhado de um vídeo explicativo sobre a ferramenta disponível no link [\[1\]](#). O código fonte se encontra disponível no link [\[2\]](#).

Sumário

1. Descrição das tecnologias utilizadas	1
2. Interface e listagem de páginas	1
2.1. Página de entrada do administrador ou home	2
2.3. Páginas de solicitação de cadastramento de usuário	2
2.4. Página de Login	3
2.5. Página de lista de usuários	4
2.6. Página de dados cadastrais de usuário	5
2.7. Página de criação de grupos do PIPA	6
2.8. Página de lista de políticas	7
2.9. Página de dados de grupo específico de usuário	7
3. Implementação do backend do PIPA (endpoints)	9
3.1. Endpoint /user	9
3.2. Endpoint /user create	11
3.3. Endpoint /policy	11
3.4. Endpoint /user/policy	13
4. Configuração do Ambiente	14
4. Referências	15

1. Descrição das tecnologias utilizadas

- Python: Linguagem principal do *backend*;
- Flask: Framework utilizado para a construção do backend;
- Javascript: Linguagem principal do frontend;
- React.js: Framework para a construção do frontend;
- PostgreSQL: Banco de dados utilizado na plataforma.

2. Interface e listagem de páginas

Seguem abaixo as descrições das páginas presentes no protótipo do PIPA.

2.1. Página de entrada do administrador ou *home*

Descrição: Home - Página principal em forma de *dashboard* que dá acesso às principais funcionalidades da plataforma PIPA.



2.3. Páginas de solicitação de cadastramento de usuário

Descrição: FormCadastro - Página em forma de formulário que possibilita a criação de novos usuários;



Protótipo PIPA

PIPA

Home Fomulário Cadastro Usuários Grupos

Solicitação de acesso

Formulário para solicitar criação de usuário para acesso ao ambiente do GSI/MPMG

Primeiro nome:

Sobrenome:

Email:


Username:


Enviar solicitação


2.4. Página de Login

Descrição: Página de autenticação federada pelo WSO2 IS que permite acesso às funcionalidades do PIPA.

Sign In

 Username

 Password



Forgot [password](#) ?

☐ Remember me on this computer

We use browser cookies to track your session to give better experience. You can refer our [Cookie Policy](#) for more details.

By signing in, you agree to our [Privacy Policy](#)

Continue

2.5. Página de lista de usuários

Descrição: Usuários - Página que dispões todos os usuários criados no PIPA. Também redireciona para as informações específicas sobre cada usuário.

PIPA

Home

Formulário Cadastro

Usuários

Grupos

Usuários

Nome	Email	Username	Ações	
Dorgival Guedes	dguesdes@mail.com	dguedes	→ EDITAR	✓ VALIDAR
Daniel Macedo	dmacedo@mail.com	dmacedo	→ EDITAR	✓ VALIDAR
Emanuella Ferraz	eferraz@mail.com	eferraz	→ EDITAR	✓ VALIDAR
José Marcos	Jmarcos@mail.com	jmarcos	→ EDITAR	✓ VALIDAR
Júnia Maisa	jmaisa@mail.com	jmaisa	→ EDITAR	✓ VALIDAR
Vinicius Oliveira	voliveira@mail.com	vOliveira	→ EDITAR	✓ VALIDAR

Protótipo PIPA

2.6. Página de dados cadastrais de usuário

Descrição: Usuário - Página que contém as informações cadastrais de um usuário específico e que possibilita a exclusão, edição e cadastramento deste usuário do FreeIPA e no Gitlab.


PIPA


HomeFomulário CadastroUsuáriosGrupos


Dorgival Guedes

Username	Email	Já cadastrado no FreeIPA?	Já cadastrado no GitLab?
dguedes	dguedes@mail.com	Não	Não

Ações

 CADASTRAR USUÁRIO NO FREEIPA E NO GITLAB

 EDITAR

 EXCLUIR

Protótipo PIPA


PIPA


HomeFomulário CadastroUsuáriosGrupos


Dorgival Guedes

Username	Email	Já cadastrado no GitLab?
dguedes	dguedes@mail.com	Não

Ações

 CADASTRAR USUARIO NO FREEIPA E NO GITLAB

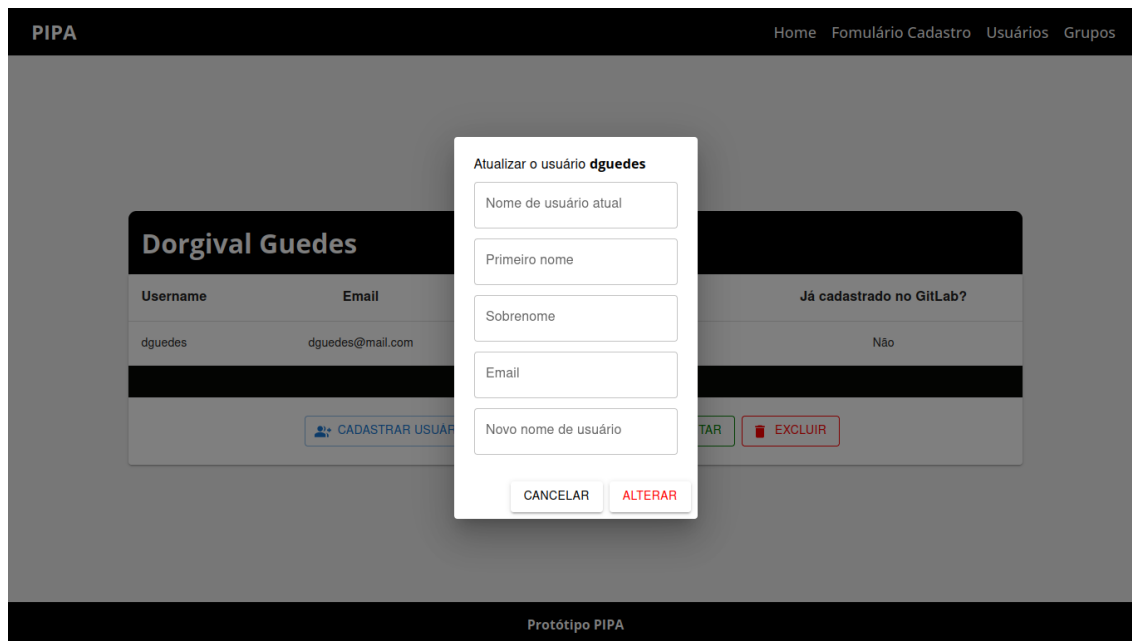
 EDITAR

 EXCLUIR

Você tem certeza que deseja excluir o usuário **dguedes**?

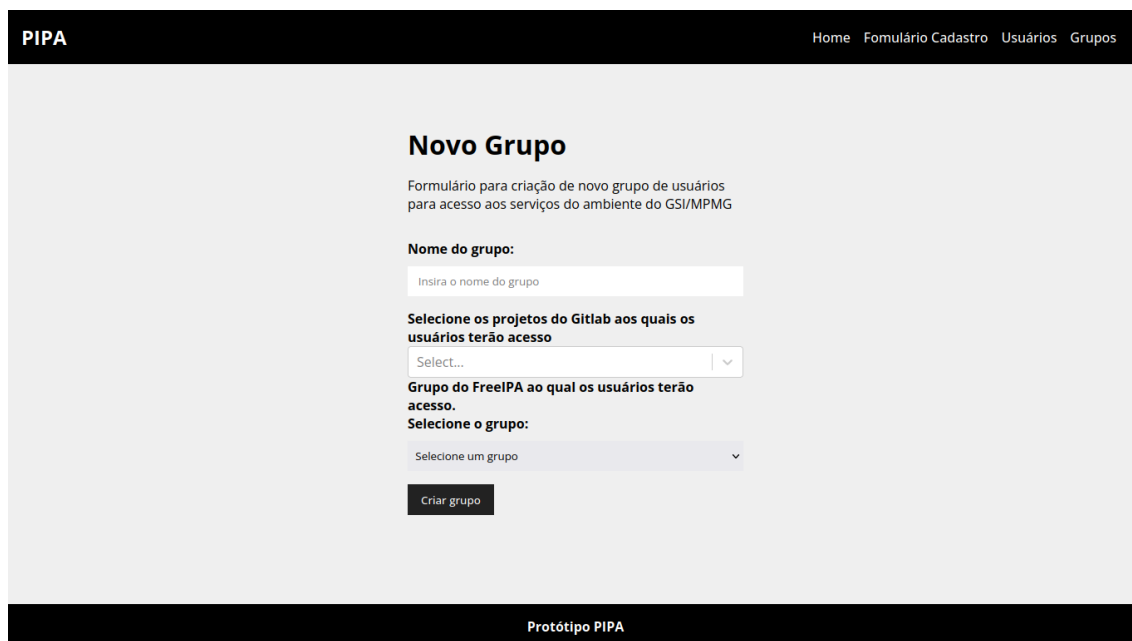
CANCELAREXCLUIR

Protótipo PIPA



2.7. Página de criação de grupos do PIPA

Descrição: CriarGrupo - Página em forma de formulário que possibilita a criação de um novo grupo do PIPA.



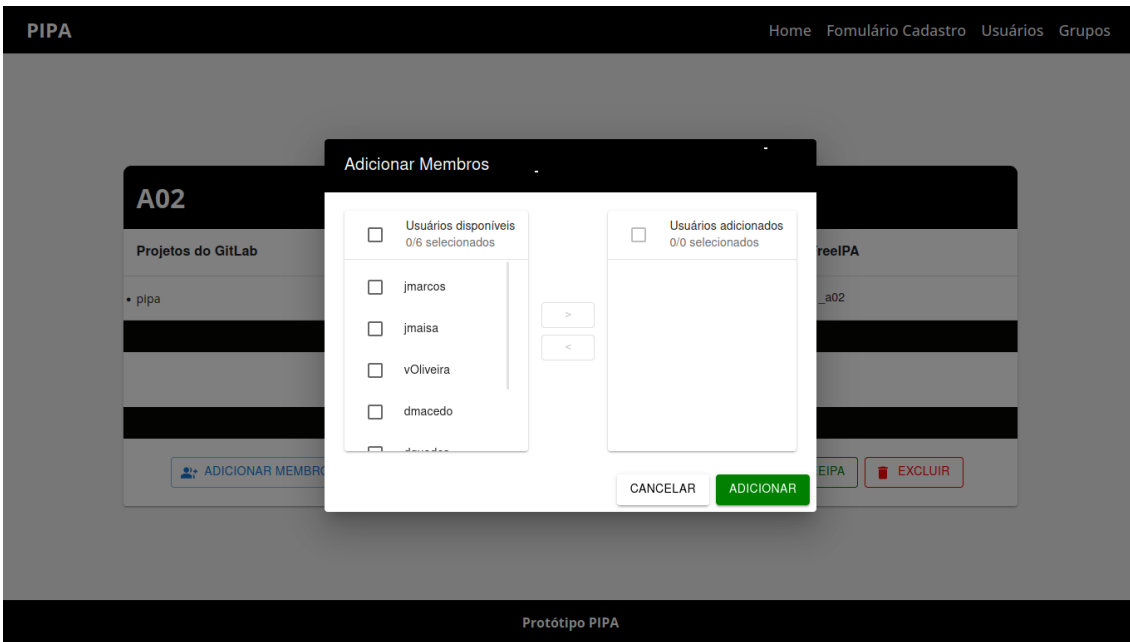
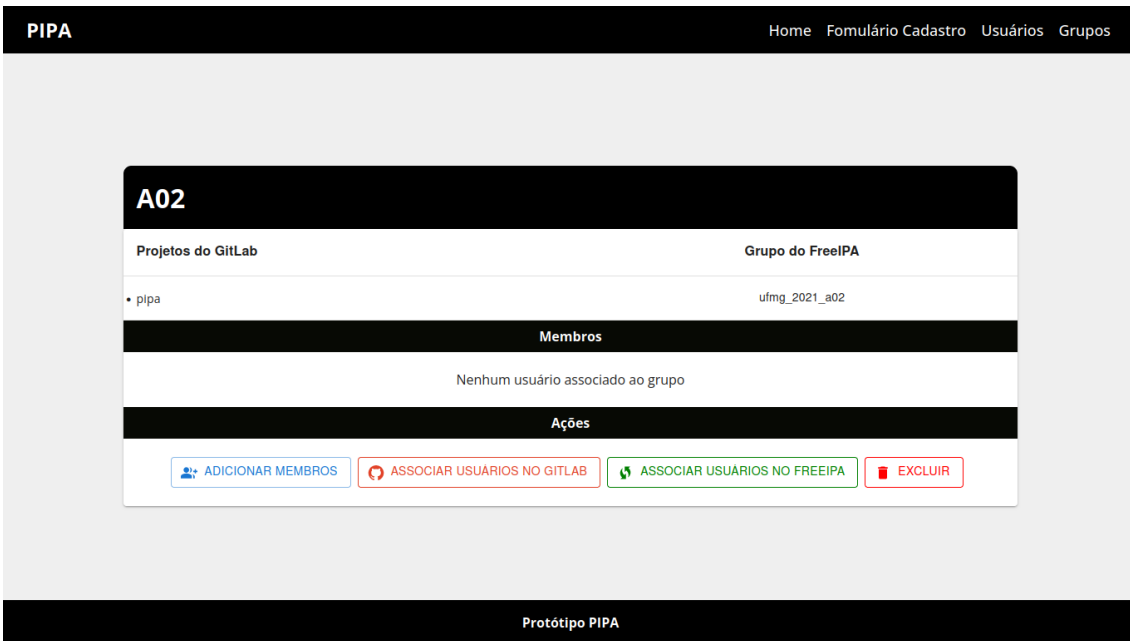
2.8. Página de lista de políticas

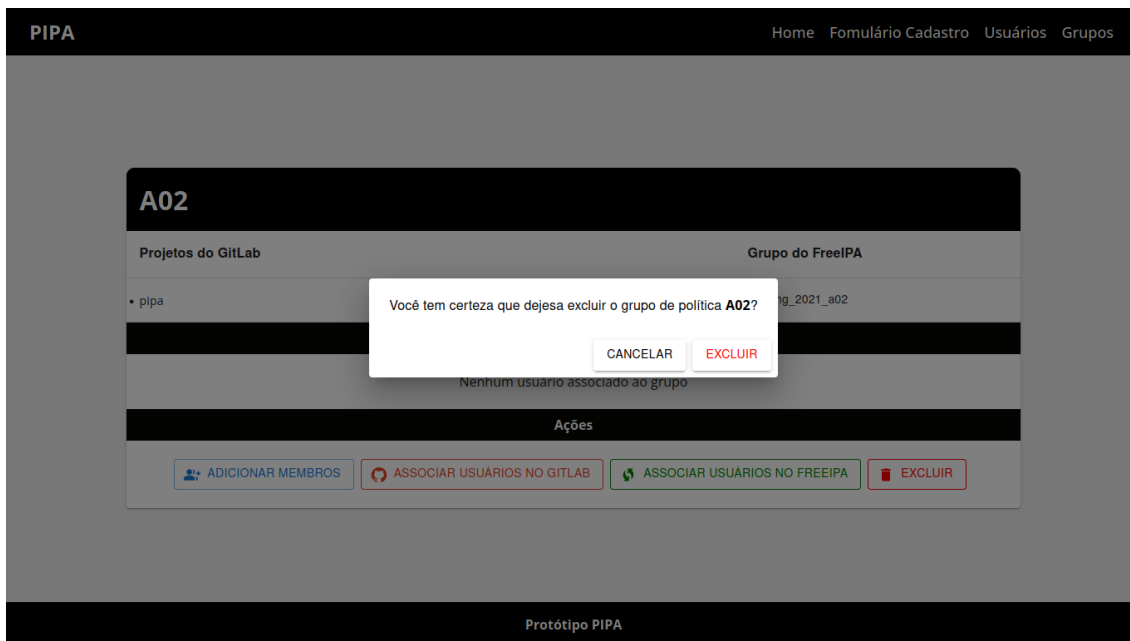
Descrição: GrupoPolíticas - Página que apresenta as políticas vigentes. Também redireciona para as informações específicas sobre cada grupo de políticas.



2.9. Página de dados de grupo específico de usuário

Descrição: GrupoPolitica - Página que contém as informações de um grupo de usuários específico, neste momento apresentando os projetos do GitLab e grupo do FreeIPA ao qual dá acesso. A página possibilita que o administrador adicione usuários ao grupo, associe usuários ao Gitlab, associe os usuários ao Freeipa e também possibilita a exclusão do grupo.





3. Implementação do *backend* do PIPA (endpoints)

A implementação do backend do PIPA será aqui mostrada por meio de um conjunto de *endpoints* de APIs. Cada *endpoint* tem um nome, uma descrição e métodos suportados. Para cada método há uma descrição, seus argumentos e um exemplo de código, quando pertinente.

3.1. Endpoint /user

Endpoint: /user

Descrição: rota para gerenciar dados de usuários na aplicação.

Métodos suportados: [GET, POST, PUT, DELETE]

GET /user

Descrição: retorna todos os usuários.

GET /user?username= X

Descrição: retorna informações do usuário X.

POST /user

Descrição: adiciona o usuário no repositório de dados da aplicação.

Argumentos esperados:

- firstName: primeiro nome do usuário
- lastName: último nome do usuário
- email: email do usuário
- username: nome de usuário

Exemplo:

```
'POST'
{
  "firstName": "User",
  "lastName": "01",
  "email": "user01@teste.com",
  "username": "user01"
}
```

PUT /user

Descrição: atualiza dados do usuário no repositório de dados da aplicação.

Argumentos esperados:

- userupdate: nome de usuário que deve ser atualizado
- firstName: primeiro nome do usuário
- lastName: último nome do usuário
- email: email do usuário
- username: nome de usuário

Exemplo:

```
'PUT'
{
  "userupdate": "user04",
  "firstName": "User",
  "lastName": "05",
  "email": "user05@teste.com",
  "username": "user05"
}
```

DELETE /user

Descrição: apaga dados do usuário no repositório de dados da aplicação.

Argumentos esperados:

- usertodelete: nome de usuário que deve ser apagado

Exemplo:

```
'DELETE'
{
  "usertodelete": "user05"
}
```

3.2. Endpoint /user create

Endpoint: /user/create

Descrição: rota para criar usuário nos serviços FreeIPA e GitLab.

Método suportado: [POST]

POST /user/create

Descrição: cria usuários nos serviços FreeIPA e GitLab.

Argumentos esperados:

- username: nome de usuário que deve ser cadastrado nos serviços FreeIPA e GitLab

Exemplo:

```
'POST'
{
  "username": username
}
```

3.3. Endpoint /policy

Endpoint: /policy

Descrição: rota para gerenciar dados de grupos de políticas na aplicação.

Métodos suportados: [GET, POST, PUT, DELETE]

GET /policy

Descrição: retorna todos os grupos de políticas.

GET /policy?policyid= X

Descrição: retorna informações do grupo de política com identificador numérico X.

POST /policy

Descrição: adiciona grupo de política no repositório de dados da aplicação.

Argumentos esperados:

- policynome: nome do grupo de política
- projectsGitLab: nested array com os projetos do GitLab
- groupipa: array com o grupo do FreelPA

Exemplo:

```
'POST'
{
  "policynome": "Policy Teste",
  "projectsGitLab": [{"value": 1, "label": "Monitoring"}],
  "groupipa": {
    "id": "4",
    "name": "D01"
  }
}
```

PUT /policy

Descrição: atualiza dados de grupo de política no repositório de dados da aplicação.

Argumentos esperados:

- policyid: id do grupo de política que deve ser atualizado
- policynome: nome do grupo de política
- projectsGitLab: nested array com os projetos do GitLab
- groupipa: array com o grupo do FreelPA

Exemplo:

```
'PUT'
{
  "policyid" : "4",
  "policynome": "GrupoA04",
  "projectsGitLab": {
    "lins": {
      "id": 5,
      "name": "lemonade"
    }
  }
}
```

```
}  
  
},  
"groupipa": "ufmg-2021a04"  
}
```

DELETE /policy

Descrição: apaga dados do usuário no repositório de dados da aplicação.

Argumentos esperados:

- polycytodelete: grupo de política que deve ser deletado

Exemplo:

```
{  
  "polycytodelete" : "4"  
}
```

3.4. Endpoint /user/policy

Endpoint: /user/policy

Descrição: rota para atualizar a qual grupo de política o usuário pertence na aplicação.

Método suportado: [POST]

POST /user/policy

Descrição: atualizara qual grupo de política o usuário pertence na aplicação.

Argumentos esperados:

- policyid: id do grupo de política
- usernames: array com os nomes dos usuários que pertencem ao grupo de política

Exemplo:

```
'POST'  
{  
  "policyid": 1,  
  "usernames": ["user01", "user02"]  
}
```

```
}
```

4. Configuração do Ambiente

Atualmente o PIPA faz o gerenciamento de identidades de dois serviços, o FreeIPA e o Gitlab. Para tanto, é necessário que sejam criados os ambientes desses dois serviços. Para o *build* do FreeIPA, ler mais informações em [\[3\]](#). Para o *build* do Gitlab, ler mais informações em [\[4\]](#).

É obrigatório ter as seguintes dependências instaladas para o build do projeto:

- Python3;
- PostgreSQL;
- Node.js.

Para instalar dependências, em /backend, digitar o comando:

```
python3 -m pip install -r requirements.txt
```

Para instalar dependências, em /frontend, digitar o comando:

```
npm install
```

Para criar seu banco de dados, em /db, digitar o comando:

```
python3 init_db.py
```

Para executar o programa, digitar os comandos:

```
/backend: python3 server.py  
/frontend: npm start
```

Para fazer a conexão entre os serviços e o PIPA, é necessário criar um arquivo `.env` e inserir as credenciais de cada serviço como no arquivo `/backend/.env.example`:

```
FREEIPA_DOMAIN =  
FREEIPA_ROOT_USERNAME =  
FREEIPA_ROOT_PASSWORD =  
  
GITLAB_DOMAIN =  
GITLAB_ROOT_USERNAME =  
GITLAB_ROOT_PASSWORD =
```

```
DATABASE_URL=  
DATABASE_PORT =  
DB =  
DB_USERNAME =  
DB_PASSWORD =  
  
OAUTH_CLIENT_KEY =  
OAUTH_CLIENT_SECRET =
```

4. Referências

[1] PIPA - Plataforma Integrada de Políticas de Acesso - Disponível em <https://www.youtube.com/watch?v=Vx9CEWh_mxM>. Acesso em 03/04/2023.

[2] PIPA repository - Disponível em <<https://github.com/WinetLabUFMG/PIPA>>. Acesso em 03/04/2023.

[3] *Quick Start Guide - FreeIPA*. Disponível em <https://www.freeipa.org/page/Quick_Start_Guide>. Acesso em 03/04/2023.

[4] *Use Gitlab*. Disponível em <<https://docs.gitlab.com/ee/user/>>. Acesso em 03/04/2023.