

Acesso, usuários e perfis

Índice de endpoints · Modelos

API-013 — Login

POST `/api/auth/login`

POST `/api/usuarios/login`

Finalidade: Login.

Fonte: `src/main/java/com/wester/storage/controller/AuthController.java:40` — `AuthController.login`.

Autenticação, autorização e perfis: Acesso público. Login não exige token. Consulte a correspondência entre autoridades e perfis em Autenticação.

Headers: nenhum de autenticação obrigatório. `Accept: application/json` recomendado. `Content-Type: application/json` para o corpo.

Parâmetros:

Sem query parameters, path variables ou headers de negócio declarados.

Corpo da requisição: `LoginDTO`; obrigatório. Validação Jakarta ativada por `@Valid`. Consulte os campos e restrições em Modelos.

Exemplo sintético (IDs devem existir e versões devem vir de consulta):

```
{
  "login": "operador@example.com",
  "senha": "SenhaExemplo123"
}
```

json

Resposta de sucesso: `UsuarioResponseDTO`. Campos em Modelos.

Exemplo estrutural sintético, sem afirmar valores de produção:

json

```
{
  "id": 1,
  "login": "operador@example.com",
  "nome": "Exemplo",
  "perfil": "CONSULTOR",
  "ativo": false,
  "token": "<TOKEN_OBTIDO_NO_LOGIN>",
  "perfilResumo": {
    "id": 1,
    "code": "CONSULTOR",
    "description": "Exemplo",
    "type": "READ_ONLY",
    "allowedScreens": [],
    "active": false,
    "createdAt": "2026-10-15T09:00:00",
    "updatedAt": "2026-10-15T09:00:00"
  }
}
```

Códigos HTTP: 200, 400, 401, 403, 429. Códigos de segurança dependem da cadeia de filtros; falhas não capturadas podem chegar ao tratamento global descrito em Erros.

Validações e erros identificados na cadeia local do serviço:

- Usuário inativo. Procure um administrador.
- Informe login e senha para continuar.

Regras de negócio e dependências: `AuthenticationService.authenticate`. Consulte documento interno de regras de negócio para efeitos persistentes e validações compartilhadas.

cURL (Bash):

bash

```
curl -X POST "${BASE_URL}/api/auth/login" -H 'Content-Type: application/json' --data '{"log
```

API-014 — Me

GET /api/auth/me

Finalidade: Me.

Fonte: `src/main/java/com/wester/storage/controller/AuthController.java:50` — `AuthController.me`.

Autenticação, autorização e perfis: Usuário autenticado; sem autoridade adicional declarada no método. Consulte a correspondência entre autoridades e perfis em Autenticação.

Headers: `Authorization: Bearer <TOKEN>`. `Accept: application/json` recomendado.

Parâmetros:

Sem query parameters, path variables ou headers de negócio declarados.

Corpo: não declarado; não enviar corpo.

Resposta de sucesso: `UsuarioResponseDTO`. Campos em Modelos.

Exemplo estrutural sintético, sem afirmar valores de produção:

```
{
  "id": 1,
  "login": "operador@example.com",
  "nome": "Exemplo",
  "perfil": "CONSULTOR",
  "ativo": false,
  "token": "<TOKEN_OBTIDO_NO_LOGIN>",
  "perfilResumo": {
    "id": 1,
    "code": "CONSULTOR",
    "description": "Exemplo",
    "type": "READ_ONLY",
    "allowedScreens": [],
    "active": false,
    "createdAt": "2026-10-15T09:00:00",
    "updatedAt": "2026-10-15T09:00:00"
  }
}
```

json

Códigos HTTP: `200`, `401`, `403`. Códigos de segurança dependem da cadeia de filtros; falhas não capturadas podem chegar ao tratamento global descrito em Erros.

Validações e erros identificados na cadeia local do serviço:

- Informe login e senha para continuar.

Regras de negócio e dependências: `AuthenticationService.getAuthenticatedUser`, `CurrentUserService.getCurrentLogin`. Consulte documento interno de regras de negócio para efeitos persistentes e validações compartilhadas.

cURL (Bash):

```
curl -X GET "${BASE_URL}/api/auth/me" -H "Authorization: Bearer ${TOKEN}"
```

bash

API-069 — Listar perfis

GET `/api/perfis`

GET `/api/profiles`

Finalidade: Listar perfis.

Fonte: `src/main/java/com/wester/storage/controller/PerfilController.java:57` — `PerfilController.listarPerfis`.

Autenticação, autorização e perfis: `hasAuthority('SCREEN_PROFILES')` or `hasAuthority('SCREEN_USERS')` Consulte a correspondência entre autoridades e perfis em Autenticação.

Headers: `Authorization: Bearer <TOKEN>`. `Accept: application/json` recomendado.

Parâmetros:

NOME	LOCAL	TIPO	OBRIGATÓRIO	PADRÃO
<code>page</code>	query	<code>int</code>	Não	0
<code>size</code>	query	<code>int</code>	Não	10
<code>search</code>	query	<code>String</code>	Não	—
<code>sort</code>	query	<code>String</code>	Não	—
<code>direction</code>	query	<code>String</code>	Não	—

Corpo: não declarado; não enviar corpo.

Resposta de sucesso: `Page<PerfilResponseDTO>`. Campos em Modelos.

Exemplo estrutural sintético, sem afirmar valores de produção:

```
{
  "content": [],
  "totalElements": 1,
  "totalPages": 0,
  "size": 1,
  "number": 1,
  "first": false,
  "last": false,
  "empty": false,
  "numberOfElements": 1,
  "pageable": {},
  "sort": {}
}
```

json

Códigos HTTP: `200`, `401`, `403`. Códigos de segurança dependem da cadeia de filtros; falhas não capturadas podem chegar ao tratamento global descrito em Erros.

Regras de negócio e dependências: `PerfilService.listarPerfis`. Consulte documento interno de regras de negócio para efeitos persistentes e validações compartilhadas.

cURL (Bash):

```
curl -X GET "${BASE_URL}/api/perfis" -H "Authorization: Bearer ${TOKEN}"
```

bash

API-070 — Listar perfis ativos para vínculo em usuários

GET `/api/perfis/ativos`

GET `/api/profiles/ativos`

Finalidade: Listar perfis ativos para vínculo em usuários.

Fonte: `src/main/java/com/wester/storage/controller/PerfilController.java:73` — `PerfilController.listarPerfisAtivos`.

Autenticação, autorização e perfis: `hasAuthority('SCREEN_PROFILES')` or `hasAuthority('SCREEN_USERS')` Consulte a correspondência entre autoridades e perfis em Autenticação.

Headers: `Authorization: Bearer <TOKEN>` . `Accept: application/json` recomendado.

Parâmetros:

Sem query parameters, path variables ou headers de negócio declarados.

Corpo: não declarado; não enviar corpo.

Resposta de sucesso: `List<PerfilDisponivelDTO>` . Campos em Modelos.

Exemplo estrutural sintético, sem afirmar valores de produção:

```
[ ]
```

json

Códigos HTTP: `200` , `401` , `403` . Códigos de segurança dependem da cadeia de filtros; falhas não capturadas podem chegar ao tratamento global descrito em Erros.

Regras de negócio e dependências: `PerfilService.listarPerfisAtivos` . Consulte documento interno de regras de negócio para efeitos persistentes e validações compartilhadas.

cURL (Bash):

```
curl -X GET "${BASE_URL}/api/perfis/ativos" -H "Authorization: Bearer ${TOKEN}"
```

bash

API-071 — Criar perfil

POST `/api/perfis`

POST `/api/profiles`

Finalidade: Criar perfil.

Fonte: `src/main/java/com/wester/storage/controller/PerfilController.java:84` — `PerfilController.criarPerfil` .

Autenticação, autorização e perfis: `hasAuthority('SCREEN_PROFILES')` and `hasRole('FULL_ACCESS')`
Consulte a correspondência entre autoridades e perfis em Autenticação.

Headers: `Authorization: Bearer <TOKEN>` . `Accept: application/json` recomendado. `Content-Type: application/json` para o corpo.

Parâmetros:

Sem query parameters, path variables ou headers de negócio declarados.

Corpo da requisição: `PerfilUpsertRequestDTO`; obrigatório. Validação Jakarta ativada por `@Valid`. Consulte os campos e restrições em Modelos.

Exemplo sintético (IDs devem existir e versões devem vir de consulta):

```
{
  "description": "Consulta de estoque",
  "type": "READ_ONLY",
  "allowedScreens": [
    "DASHBOARD",
    "WAREHOUSE",
    "PRODUCTS"
  ],
  "active": true
}
```

json

Resposta de sucesso: `PerfilResponseDTO`. Campos em Modelos.

Exemplo estrutural sintético, sem afirmar valores de produção:

```
{
  "id": 1,
  "code": "CONSULTOR",
  "description": "Exemplo",
  "type": "READ_ONLY",
  "allowedScreens": [],
  "active": false,
  "createdAt": "2026-10-15T09:00:00",
  "updatedAt": "2026-10-15T09:00:00"
}
```

json

Códigos HTTP: `201`, `400`, `401`, `403`. Códigos de segurança dependem da cadeia de filtros; falhas não capturadas podem chegar ao tratamento global descrito em Erros.

Validações e erros identificados na cadeia local do serviço:

- Já existe um perfil com o código: [mensagem pode receber complemento dinâmico]
- Descrição do perfil é obrigatória.
- Não foi possível gerar o código do perfil.

- Selecione pelo menos uma tela permitida.
- Telas inválidas informadas: [mensagem pode receber complemento dinâmico]

Regras de negócio e dependências: `PerfilService.criarPerfil`. Consulte documento interno de regras de negócio para efeitos persistentes e validações compartilhadas.

cURL (Bash):

```
curl -X POST "${BASE_URL}/api/perfis" -H "Authorization: Bearer ${TOKEN}" -H 'Content-Type: application/json'
```

API-072 — Atualizar perfil

PUT `/api/perfis/{id}`

PUT `/api/profiles/{id}`

Finalidade: Atualizar perfil.

Fonte: `src/main/java/com/wester/storage/controller/PerfilController.java:100` — `PerfilController.atualizarPerfil`.

Autenticação, autorização e perfis: `hasAuthority('SCREEN_PROFILES')` and `hasRole('FULL_ACCESS')`
Consulte a correspondência entre autoridades e perfis em Autenticação.

Headers: `Authorization: Bearer <TOKEN>`. `Accept: application/json` recomendado. `Content-Type: application/json` para o corpo.

Parâmetros:

NOME	LOCAL	TIPO	OBRIGATÓRIO	PADRÃO
<code>id</code>	path	<code>Long</code>	Sim	—

Corpo da requisição: `PerfilUpsertRequestDTO`; obrigatório. Validação Jakarta ativada por `@Valid`.
Consulte os campos e restrições em Modelos.

Exemplo sintético (IDs devem existir e versões devem vir de consulta):

json

```
{
  "description": "Consulta de estoque",
  "type": "READ_ONLY",
  "allowedScreens": [
    "DASHBOARD",
    "WAREHOUSE",
    "PRODUCTS"
  ],
  "active": true
}
```

Resposta de sucesso: `PerfilResponseDTO`. Campos em Modelos.

Exemplo estrutural sintético, sem afirmar valores de produção:

json

```
{
  "id": 1,
  "code": "CONSULTOR",
  "description": "Exemplo",
  "type": "READ_ONLY",
  "allowedScreens": [],
  "active": false,
  "createdAt": "2026-10-15T09:00:00",
  "updatedAt": "2026-10-15T09:00:00"
}
```

Códigos HTTP: `200`, `400`, `401`, `403`, `404`, `500`. Códigos de segurança dependem da cadeia de filtros; falhas não capturadas podem chegar ao tratamento global descrito em Erros.

Validações e erros identificados na cadeia local do serviço:

- Descrição do perfil é obrigatória.
- Selecione pelo menos uma tela permitida.
- Telas inválidas informadas: [mensagem pode receber complemento dinâmico]

Regras de negócio e dependências: `PerfilService.atualizarPerfil`. Consulte documento interno de regras de negócio para efeitos persistentes e validações compartilhadas.

cURL (Bash):

```
curl -X PUT "${BASE_URL}/api/perfis/1" -H "Authorization: Bearer ${TOKEN}" -H 'Content-Type: application/json'
```

API-073 — Ativar perfil

PATCH `/api/perfis/{id}/ativar`

PATCH `/api/perfis/{id}/activate`

PATCH `/api/profiles/{id}/ativar`

PATCH `/api/profiles/{id}/activate`

Finalidade: Ativar perfil.

Fonte: `src/main/java/com/wester/storage/controller/PerfilController.java:123` —
`PerfilController.ativarPerfil`.

Autenticação, autorização e perfis: `hasAuthority('SCREEN_PROFILES')` and `hasRole('FULL_ACCESS')`
Consulte a correspondência entre autoridades e perfis em Autenticação.

Headers: `Authorization: Bearer <TOKEN>`. `Accept: application/json` recomendado.

Parâmetros:

NOME	LOCAL	TIPO	OBRIGATÓRIO	PADRÃO
<code>id</code>	path	<code>Long</code>	Sim	—

Corpo: não declarado; não enviar corpo.

Resposta de sucesso: `PerfilResponseDTO`. Campos em Modelos.

Exemplo estrutural sintético, sem afirmar valores de produção:

json

```
{
  "id": 1,
  "code": "CONSULTOR",
  "description": "Exemplo",
  "type": "READ_ONLY",
  "allowedScreens": [],
  "active": false,
  "createdAt": "2026-10-15T09:00:00",
  "updatedAt": "2026-10-15T09:00:00"
}
```

Códigos HTTP: 200, 401, 403, 404, 500. Códigos de segurança dependem da cadeia de filtros; falhas não capturadas podem chegar ao tratamento global descrito em Erros.

Regras de negócio e dependências: Implementação no próprio controller. Consulte documento interno de regras de negócio para efeitos persistentes e validações compartilhadas.

cURL (Bash):

bash

```
curl -X PATCH "${BASE_URL}/api/perfis/1/ativar" -H "Authorization: Bearer ${TOKEN}"
```

API-074 — Inativar perfil

PATCH /api/perfis/{id}/inativar

PATCH /api/perfis/{id}/inactivate

PATCH /api/profiles/{id}/inativar

PATCH /api/profiles/{id}/inactivate

Finalidade: Inativar perfil.

Fonte: src/main/java/com/wester/storage/controller/PerfilController.java:134 —
 PerfilController.inativarPerfil.

Autenticação, autorização e perfis: hasAuthority('SCREEN_PROFILES') and hasRole('FULL_ACCESS')
 Consulte a correspondência entre autoridades e perfis em Autenticação.

Headers: Authorization: Bearer <TOKEN>. Accept: application/json recomendado.

Parâmetros:

NOME	LOCAL	TIPO	OBRIGATÓRIO	PADRÃO
id	path	Long	Sim	—

Corpo: não declarado; não enviar corpo.

Resposta de sucesso: `PerfilResponseDTO`. Campos em Modelos.

Exemplo estrutural sintético, sem afirmar valores de produção:

```
{
  "id": 1,
  "code": "CONSULTOR",
  "description": "Exemplo",
  "type": "READ_ONLY",
  "allowedScreens": [],
  "active": false,
  "createdAt": "2026-10-15T09:00:00",
  "updatedAt": "2026-10-15T09:00:00"
}
```

json

Códigos HTTP: `200`, `401`, `403`, `404`, `500`. Códigos de segurança dependem da cadeia de filtros; falhas não capturadas podem chegar ao tratamento global descrito em Erros.

Regras de negócio e dependências: Implementação no próprio controller. Consulte documento interno de regras de negócio para efeitos persistentes e validações compartilhadas.

cURL (Bash):

```
curl -X PATCH "${BASE_URL}/api/perfis/1/inativar" -H "Authorization: Bearer ${TOKEN}"
```

bash

API-109 — Listar Todos Usuarios

GET `/api/usuarios`

Finalidade: Listar Todos Usuarios.

Fonte: `src/main/java/com/wester/storage/controller/UsuarioController.java:53` — `UsuarioController.listarTodosUsuarios`.

Autenticação, autorização e perfis: `hasAuthority('SCREEN_USERS')` Consulte a correspondência entre autoridades e perfis em Autenticação.

Headers: `Authorization: Bearer <TOKEN>` . `Accept: application/json` recomendado.

Parâmetros:

Sem query parameters, path variables ou headers de negócio declarados.

Corpo: não declarado; não enviar corpo.

Resposta de sucesso: `List<UsuarioResponseDTO>` . Campos em Modelos.

Exemplo estrutural sintético, sem afirmar valores de produção:

```
[ ]
```

json

Códigos HTTP: `200` , `401` , `403` . Códigos de segurança dependem da cadeia de filtros; falhas não capturadas podem chegar ao tratamento global descrito em Erros.

Regras de negócio e dependências: `UsuarioService.listarTodosUsuarios` . Consulte documento interno de regras de negócio para efeitos persistentes e validações compartilhadas.

cURL (Bash):

```
curl -X GET "${BASE_URL}/api/usuarios" -H "Authorization: Bearer ${TOKEN}"
```

bash

API-110 — Buscar Usuario Por Id

`GET` `/api/usuarios/id/{id}`

Finalidade: Buscar Usuario Por Id.

Fonte: `src/main/java/com/wester/storage/controller/UsuarioController.java:65` — `UsuarioController.buscarUsuarioPorId` .

Autenticação, autorização e perfis: `hasAuthority('SCREEN_USERS')` Consulte a correspondência entre autoridades e perfis em Autenticação.

Headers: `Authorization: Bearer <TOKEN>` . `Accept: application/json` recomendado.

Parâmetros:

NOME	LOCAL	TIPO	OBRIGATÓRIO	PADRÃO
id	path	Long	Sim	—

Corpo: não declarado; não enviar corpo.

Resposta de sucesso: `UsuarioResponseDTO`. Campos em Modelos.

Exemplo estrutural sintético, sem afirmar valores de produção:

```

{
  "id": 1,
  "login": "operador@example.com",
  "nome": "Exemplo",
  "perfil": "CONSULTOR",
  "ativo": false,
  "token": "<TOKEN_OBTIDO_NO_LOGIN>",
  "perfilResumo": {
    "id": 1,
    "code": "CONSULTOR",
    "description": "Exemplo",
    "type": "READ_ONLY",
    "allowedScreens": [],
    "active": false,
    "createdAt": "2026-10-15T09:00:00",
    "updatedAt": "2026-10-15T09:00:00"
  }
}

```

Códigos HTTP: `200`, `401`, `403`, `404`. Códigos de segurança dependem da cadeia de filtros; falhas não capturadas podem chegar ao tratamento global descrito em Erros.

Regras de negócio e dependências: `UsuarioService.buscarUsuarioPorId`. Consulte documento interno de regras de negócio para efeitos persistentes e validações compartilhadas.

cURL (Bash):

```

curl -X GET "${BASE_URL}/api/usuarios/id/1" -H "Authorization: Bearer ${TOKEN}"

```

API-111 — Buscar Por Login

GET `/api/usuarios/by-login/{login}`

Finalidade: Buscar Por Login.

Fonte: `src/main/java/com/wester/storage/controller/UsuarioController.java:76` — `UsuarioController.buscarPorLogin`.

Autenticação, autorização e perfis: `hasAuthority('SCREEN_USERS')` Consulte a correspondência entre autoridades e perfis em Autenticação.

Headers: `Authorization: Bearer <TOKEN>`. `Accept: application/json` recomendado.

Parâmetros:

NOME	LOCAL	TIPO	OBRIGATÓRIO	PADRÃO
<code>login</code>	path	<code>String</code>	Sim	—

Corpo: não declarado; não enviar corpo.

Resposta de sucesso: `UsuarioResponseDTO`. Campos em Modelos.

Exemplo estrutural sintético, sem afirmar valores de produção:

json

```
{
  "id": 1,
  "login": "operador@example.com",
  "nome": "Exemplo",
  "perfil": "CONSULTOR",
  "ativo": false,
  "token": "<TOKEN_OBTIDO_NO_LOGIN>",
  "perfilResumo": {
    "id": 1,
    "code": "CONSULTOR",
    "description": "Exemplo",
    "type": "READ_ONLY",
    "allowedScreens": [],
    "active": false,
    "createdAt": "2026-10-15T09:00:00",
    "updatedAt": "2026-10-15T09:00:00"
  }
}
```

Códigos HTTP: 200, 401, 403, 404. Códigos de segurança dependem da cadeia de filtros; falhas não capturadas podem chegar ao tratamento global descrito em Erros.

Regras de negócio e dependências: `UsuarioService.buscarPorLogin`. Consulte documento interno de regras de negócio para efeitos persistentes e validações compartilhadas.

cURL (Bash):

bash

```
curl -X GET "${BASE_URL}/api/usuarios/by-login/1" -H "Authorization: Bearer ${TOKEN}"
```

API-112 — Criar Usuario

POST `/api/usuarios`

Finalidade: Criar Usuario. Público. Somente SCREEN_USERS + FULL_ACCESS podem escolher perfil; nos demais casos é forçado CONSULTOR.

Fonte: `src/main/java/com/wester/storage/controller/UsuarioController.java:86` — `UsuarioController.criarUsuario`.

Autenticação, autorização e perfis: Acesso público. Público. Somente SCREEN_USERS + FULL_ACCESS podem escolher perfil; nos demais casos é forçado CONSULTOR. Consulte a

correspondência entre autoridades e perfis em Autenticação.

Headers: nenhum de autenticação obrigatório. `Accept: application/json` recomendado. `Content-Type: application/json` para o corpo.

Parâmetros:

Sem query parameters, path variables ou headers de negócio declarados.

Corpo da requisição: `UsuarioRequestDTO` ; obrigatório. Validação Jakarta ativada por `@Valid` . Consulte os campos e restrições em Modelos.

Exemplo sintético (IDs devem existir e versões devem vir de consulta):

```
{
  "login": "novo.usuario@example.com",
  "nome": "Usuário de exemplo",
  "senha": "SenhaExemplo123",
  "perfil": "CONSULTOR"
}
```

json

Resposta de sucesso: `UsuarioResponseDTO` . Campos em Modelos.

Exemplo estrutural sintético, sem afirmar valores de produção:

```
{
  "id": 1,
  "login": "operador@example.com",
  "nome": "Exemplo",
  "perfil": "CONSULTOR",
  "ativo": false,
  "token": "<TOKEN_OBTIDO_NO_LOGIN>",
  "perfilResumo": {
    "id": 1,
    "code": "CONSULTOR",
    "description": "Exemplo",
    "type": "READ_ONLY",
    "allowedScreens": [],
    "active": false,
    "createdAt": "2026-10-15T09:00:00",
    "updatedAt": "2026-10-15T09:00:00"
  }
}
```

json

Códigos HTTP: 201, 400, 401, 403, 429, 500. Códigos de segurança dependem da cadeia de filtros; falhas não capturadas podem chegar ao tratamento global descrito em Erros.

Validações e erros identificados na cadeia local do serviço:

- Login já existe: [mensagem pode receber complemento dinâmico]
- Nenhum perfil ativo cadastrado para validar usuários.

Regras de negócio e dependências: `UsuarioService.salvarUsuario`. Consulte documento interno de regras de negócio para efeitos persistentes e validações compartilhadas.

cURL (Bash):

```
curl -X POST "${BASE_URL}/api/usuarios" -H 'Content-Type: application/json' --data '{"login
```

bash

API-113 — Atualizar usuário (nome/login/perfil)

PUT `/api/usuarios/id/{id}`

Finalidade: Atualizar usuário (nome/login/perfil).

Fonte: `src/main/java/com/wester/storage/controller/UsuarioController.java:117` — `UsuarioController.atualizarUsuario`.

Autenticação, autorização e perfis: `hasAuthority('SCREEN_USERS')` and `hasRole('FULL_ACCESS')`
Consulte a correspondência entre autoridades e perfis em Autenticação.

Headers: `Authorization: Bearer <TOKEN>`. `Accept: application/json` recomendado. `Content-Type: application/json` para o corpo.

Parâmetros:

NOME	LOCAL	TIPO	OBRIGATÓRIO	PADRÃO
id	path	Long	Sim	—

Corpo da requisição: `UsuarioUpdateRequestDTO`; obrigatório. Validação Jakarta ativada por `@Valid`.
Consulte os campos e restrições em Modelos.

Exemplo sintético (IDs devem existir e versões devem vir de consulta):

json

```
{
  "login": "operador@example.com",
  "nome": "Usuário de exemplo",
  "perfil": "CONSULTOR"
}
```

Resposta de sucesso: `UsuarioResponseDTO`. Campos em Modelos.

Exemplo estrutural sintético, sem afirmar valores de produção:

json

```
{
  "id": 1,
  "login": "operador@example.com",
  "nome": "Exemplo",
  "perfil": "CONSULTOR",
  "ativo": false,
  "token": "<TOKEN_OBTIDO_NO_LOGIN>",
  "perfilResumo": {
    "id": 1,
    "code": "CONSULTOR",
    "description": "Exemplo",
    "type": "READ_ONLY",
    "allowedScreens": [],
    "active": false,
    "createdAt": "2026-10-15T09:00:00",
    "updatedAt": "2026-10-15T09:00:00"
  }
}
```

Códigos HTTP: `200`, `400`, `401`, `403`, `404`, `500`. Códigos de segurança dependem da cadeia de filtros; falhas não capturadas podem chegar ao tratamento global descrito em Erros.

Validações e erros identificados na cadeia local do serviço:

- Login já existe: [mensagem pode receber complemento dinâmico]
- Nenhum perfil ativo cadastrado para validar usuários.

Regras de negócio e dependências: `UsuarioService.atualizarUsuario`. Consulte documento interno de regras de negócio para efeitos persistentes e validações compartilhadas.

cURL (Bash):

```
curl -X PUT "${BASE_URL}/api/usuarios/id/1" -H "Authorization: Bearer ${TOKEN}" -H 'Content-Type: application/json' -d '{ "senhaAtual": "SenhaExemplo123", "novaSenha": "NovaSenhaExemplo123", "confirmarNovaSenha": "NovaSenhaExemplo123" }'
```

API-114 — Alterar senha do usuário

POST `/api/usuarios/id/{id}/alterar-senha`

Finalidade: Alterar senha do usuário. Próprio usuário OU SCREEN_USERS + FULL_ACCESS. A senha atual é validada nos dois casos; não é um fluxo de recuperação administrativa sem senha.

Fonte: `src/main/java/com/wester/storage/controller/UsuarioController.java:160` — `UsuarioController.alterarSenha`.

Autenticação, autorização e perfis: Usuário autenticado; sem autoridade adicional declarada no método. Consulte a correspondência entre autoridades e perfis em Autenticação.

Headers: `Authorization: Bearer <TOKEN>`. `Accept: application/json` recomendado. `Content-Type: application/json` para o corpo.

Parâmetros:

NOME	LOCAL	TIPO	OBRIGATÓRIO	PADRÃO
<code>id</code>	path	<code>Long</code>	Sim	—

Corpo da requisição: `UsuarioChangePasswordRequestDTO`; obrigatório. Validação Jakarta ativada por `@Valid`. Consulte os campos e restrições em Modelos.

Exemplo sintético (IDs devem existir e versões devem vir de consulta):

```
{
  "senhaAtual": "SenhaExemplo123",
  "novaSenha": "NovaSenhaExemplo123",
  "confirmarNovaSenha": "NovaSenhaExemplo123"
}
```

Resposta de sucesso: sem corpo quando 204.

Códigos HTTP: `204`, `400`, `401`, `403`, `404`, `500`. Códigos de segurança dependem da cadeia de filtros; falhas não capturadas podem chegar ao tratamento global descrito em Erros.

Validações e erros identificados na cadeia local do serviço:

- Confirmação de senha não confere.
- Nova senha deve ter no mínimo 6 caracteres.
- Senha atual inválida.

Regras de negócio e dependências: `UsuarioService.alterarSenha`. Consulte documento interno de regras de negócio para efeitos persistentes e validações compartilhadas.

cURL (Bash):

```
curl -X POST "${BASE_URL}/api/usuarios/id/1/alterar-senha" -H "Authorization: Bearer ${TOKEN}"
```

API-115 — Atualizar status do usuário (ativo/inativo)

PATCH `/api/usuarios/id/{id}/status`

Finalidade: Atualizar status do usuário (ativo/inativo).

Fonte: `src/main/java/com/wester/storage/controller/UsuarioController.java:215` — `UsuarioController.atualizarStatusUsuario`.

Autenticação, autorização e perfis: `hasAuthority('SCREEN_USERS')` and `hasRole('FULL_ACCESS')`
Consulte a correspondência entre autoridades e perfis em Autenticação.

Headers: `Authorization: Bearer <TOKEN>`. `Accept: application/json` recomendado. `Content-Type: application/json` para o corpo.

Parâmetros:

NOME	LOCAL	TIPO	OBRIGATÓRIO	PADRÃO
<code>id</code>	path	<code>Long</code>	Sim	—

Corpo da requisição: `UsuarioStatusUpdateRequestDTO`; obrigatório. Validação Jakarta ativada por `@Valid`. Consulte os campos e restrições em Modelos.

Exemplo sintético (IDs devem existir e versões devem vir de consulta):

json

```
{
  "ativo": false
}
```

Resposta de sucesso: `UsuarioResponseDTO`. Campos em Modelos.

Exemplo estrutural sintético, sem afirmar valores de produção:

json

```
{
  "id": 1,
  "login": "operador@example.com",
  "nome": "Exemplo",
  "perfil": "CONSULTOR",
  "ativo": false,
  "token": "<TOKEN_OBTIDO_NO_LOGIN>",
  "perfilResumo": {
    "id": 1,
    "code": "CONSULTOR",
    "description": "Exemplo",
    "type": "READ_ONLY",
    "allowedScreens": [],
    "active": false,
    "createdAt": "2026-10-15T09:00:00",
    "updatedAt": "2026-10-15T09:00:00"
  }
}
```

Códigos HTTP: `200`, `400`, `401`, `403`, `404`, `500`. Códigos de segurança dependem da cadeia de filtros; falhas não capturadas podem chegar ao tratamento global descrito em Erros.

Regras de negócio e dependências: `UsuarioService.atualizarStatus`. Consulte documento interno de regras de negócio para efeitos persistentes e validações compartilhadas.

cURL (Bash):

bash

```
curl -X PATCH "${BASE_URL}/api/usuarios/id/1/status" -H "Authorization: Bearer ${TOKEN}" -F
```

API-116 — Excluir usuário por ID

DELETE `/api/usuarios/id/{id}`

Finalidade: Excluir usuário por ID.

Fonte: `src/main/java/com/wester/storage/controller/UsuarioController.java:238` — `UsuarioController.deletarUsuario`.

Autenticação, autorização e perfis: `hasAuthority('SCREEN_USERS')` and `hasRole('FULL_ACCESS')`
Consulte a correspondência entre autoridades e perfis em Autenticação.

Headers: `Authorization: Bearer <TOKEN>`. `Accept: application/json` recomendado.

Parâmetros:

NOME	LOCAL	TIPO	OBRIGATÓRIO	PADRÃO
<code>id</code>	path	<code>Long</code>	Sim	—

Corpo: não declarado; não enviar corpo.

Resposta de sucesso: sem corpo quando 204.

Códigos HTTP: `204`, `401`, `403`, `404`, `500`. Códigos de segurança dependem da cadeia de filtros; falhas não capturadas podem chegar ao tratamento global descrito em Erros.

Regras de negócio e dependências: `UsuarioService.deletarUsuario`. Consulte documento interno de regras de negócio para efeitos persistentes e validações compartilhadas.

cURL (Bash):

```
curl -X DELETE "${BASE_URL}/api/usuarios/id/1" -H "Authorization: Bearer ${TOKEN}"
```

bash