🎓 Sistema Acadêmico - Regras de Negócio

Este sistema gerencia **alunos, professores, secretários, turmas, endereços, atividades, entregas e frequência**.

O projeto segue o padrão **MVC modular**, com autenticação via **JWT**, controle de acesso por **papéis (RBAC)** e uso de banco relacional (PostgreSQL) via **Sequelize ORM**.

📓 Visão Geral

📌 Perfis do sistema

- **Secretário**: Administrador do sistema (cadastros, atribuições, turmas).
- **Professor**: Responsável por criar atividades e registrar frequência.
- **Aluno**: Participa de turmas, entrega atividades e visualiza sua evolução.

🧱 Entidades e Atributos Principais

Entidade	Atributos
`Secretario`	`id`, `nome`, `email`, `senha`
`Aluno`	`matricula`, `nome`, `email`, `senha`, `turma_id`
`Professor`	`matricula`, `nome`, `email`, `senha`
`Turma`	`id`, `nome`, `descricao`, `turno`, `professor_id`
`Endereco`	`id`, `cep`, `logradouro`, `numero`, `bairro`, `localidade`, `uf`, `ponto_referencia`, `usuario_id`, `tipo_usuario`
`Atividade`	`id`, `titulo`, `descricao`, `tipo`, `data_entrega`, `turma_id`, `professor_id`
`Entrega`	`id`, `atividade_id`, `aluno_id`, `data_entrega`, `arquivo_url`, `nota`, `status`
`Frequencia`	`id`, `aluno_id`, `turma_id`, `data`, `presenca` (boolean)

📐 Relacionamentos

- `Aluno` → pertence a uma `Turma`
- `Turma` → pertence a um `Professor`
- `Turma` → tem várias `Atividades`
- `Professor` → cria várias `Atividades`
- `Aluno` → realiza várias `Entregas` (de `Atividades`)
- `Frequencia` → é registrada por `Aluno` em `Turma`
- `Aluno`, `Professor`, `Secretário` → têm `1 Endereço`

🔐 Permissões por Papel (RBAC)

Ação	Secretário	Professor	Aluno
Criar/editar/excluir alunos	☑	❌	❌
Criar/editar/excluir professores	☑	❌	❌
Criar/editar/excluir turmas	☑	❌	❌
Atribuir professor à turma	☑	❌	❌
Atribuir aluno à turma	☑	❌	❌
Cadastrar endereço	☑	☑	☑
Criar atividades	❌	☑	❌
Lançar frequência	❌	☑	❌
Entregar atividade	❌	❌	☑
Visualizar suas turmas	☑	☑	☑
Visualizar todos os alunos	☑	☑	❌
Visualizar todos os professores	☑	❌	❌
Visualizar frequência	☑	☑	☑
Visualizar entregas e notas	☑	☑	☑

🧠 Regras de Negócio

📌 Secretário

- Acesso total ao sistema.
- Pode cadastrar, editar e excluir alunos, professores e turmas.
- Atribui professores e alunos às turmas.
- Pode gerenciar endereços de qualquer usuário.

📌 Professor

- Só pode visualizar alunos das suas turmas.
- Pode criar atividades vinculadas às suas turmas.
- Pode lançar notas e presença por data e aluno.

📌 Aluno

- Só pode visualizar suas próprias informações e sua turma.
- Pode entregar atividades dentro do prazo estipulado.
- Pode visualizar suas notas e sua frequência.
- Pode cadastrar/editar seu próprio endereço.

📌 Atividades

- São criadas pelos professores e associadas a uma turma.
- Podem ser do tipo: `Prova`, `Trabalho`, `Atividade Extra`, etc.
- Possuem título, descrição, data de entrega e tipo.

📌 Entregas

- São feitas pelos alunos para atividades da sua turma.
- Podem conter arquivo e são entregues dentro do prazo.
- Professores podem atribuir notas e marcar status: `Entregue`, `Corrigido`, etc.

📌 Frequência

- Cada aula tem presença registrada por aluno e por data.
- Professores podem lançar a presença dos alunos das suas turmas.
- Alunos podem visualizar seu histórico de frequência.

🚦 Fluxos Operacionais

📝 Cadastro de Aluno

1. Secretário autentica no sistema.
2. Preenche dados do aluno.
3. Atribui a uma turma existente.
4. Cria ou permite que o aluno cadastre seu endereço.

📝 Criação de Atividade

1. Professor acessa suas turmas.
2. Cria nova atividade com título, descrição e data de entrega.
3. Atividade é vinculada à turma.

📝 Entrega de Atividade

1. Aluno visualiza lista de atividades pendentes.
2. Anexa o arquivo e envia dentro do prazo.
3. Professor pode corrigir e dar nota.

📝 Registro de Frequência

1. Professor seleciona a turma e a data da aula.
2. Marca presença dos alunos (`presente` ou `ausente`).
3. Aluno pode visualizar frequência por mês ou disciplina.

🌓 Segurança e Autenticação

- Autenticação por **JWT**.
- Controle de acesso via **RBAC** (roles: `secretario`, `professor`, `aluno`).

- Middleware para verificação de token e permissões.
- Todas as rotas protegidas com base no papel do usuário.

☑ Próximos Passos

- ☐ Criar models Sequelize com relacionamentos e validações.
- ☐ Criar migrations para todas as tabelas.
- ☐ Implementar middleware de autenticação e RBAC.
- ☐ Criar seed inicial com um `secretário` padrão.
- ☐ Construir controllers e rotas para todas as entidades.

🗁 Sugestão de Estrutura de Pastas

```
src/
├── modules/
│   ├── aluno/
│   │   ├── controllers/
│   │   ├── models/
│   │   ├── routes/
│   │   └── services/
│   ├── professor/
│   │   ├── controllers/
│   │   ├── models/
│   │   ├── routes/
│   │   └── services/
│   ├── secretario/
│   │   ├── controllers/
│   │   ├── models/
│   │   ├── routes/
│   │   └── services/
│   ├── turma/
│   │   ├── controllers/
│   │   ├── models/
│   │   ├── routes/
│   │   └── services/
│   ├── endereco/
│   │   ├── controllers/
│   │   ├── models/
│   │   ├── routes/
│   │   └── services/
│   ├── atividade/
│   │   ├── controllers/
│   │   ├── models/
│   │   ├── routes/
│   │   └── services/
│   ├── entrega/
│   │   ├── controllers/
│   │   ├── models/
│   │   ├── routes/
```

```
│   │   └── services/
│   └── frequencia/
│       ├── controllers/
│       ├── models/
│       ├── routes/
│       └── services/
├── config/
│   └── database.js
├── database/
│   ├── migrations/
│   └── seeders/
├── middleware/
│   ├── autenticacao.js
│   └── autorizacao.js
└── index.js
```

> Essa estrutura segue uma arquitetura **MVC modularizada**, separando responsabilidades por entidade. Ideal para escalar o projeto, manter a legibilidade e aplicar princípios de **Clean Architecture**.

✨ Tecnologias sugeridas

- **Node.js** + **Express**
- **Sequelize ORM**
- **PostgreSQL**
- **JWT** para autenticação
- **bcrypt** para hash de senha
- **express-validator** para validações
- **dotenv** para configuração por ambiente

📢 Contribuições futuras

- 📦 Implementar upload de arquivos para entregas
- 📊 Relatórios por turma (notas, presença)
- 🗓️ Calendário acadêmico e avisos por turma
- 📬 Notificações de prazo via e-mail

👨‍💻 Observação

Desenvolvido para fins acadêmicos e profissionais
Projeto de sistema escolar completo com foco em organização e boas práticas

---