**CONTEXTUALIZAÇÃO**

Este exercício tem como objetivo implementar melhorias na API do projeto XPTO, um sistema de ensino a distância. A API do projeto XPTO foi desenvolvida em Node.js (JavaScript), utilizando o conceito REST que visa o uso correto dos métodos disponibilizados pelo protocolo HTTP (e.g., GET, POST, PUT, PATCH, DELETE).

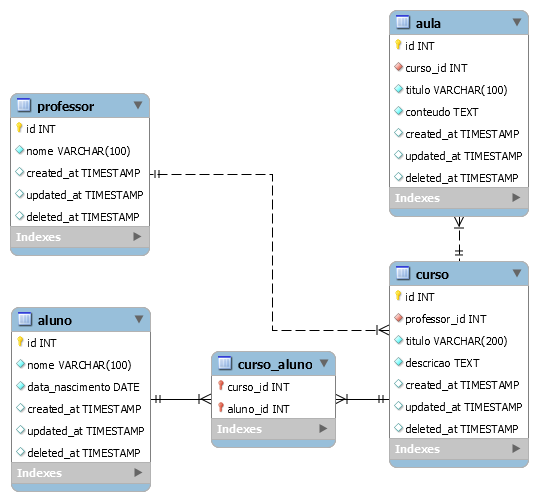
A API utiliza o framework Express (<http://expressjs.com/pt-br/>) para ofertar as funcionalidades e o projeto swagger (<https://swagger.io>) para realizar o design da API.   
Por fim, o ORM Sequelize (<http://docs.sequelizejs.com/>) é adotado para abstrair as interações com o banco de dados PostgreSQL.

As funcionalidades da API são definidas através da especificação do swagger na versão 2.0. <https://github.com/OAI/OpenAPI-Specification/blob/master/versions/2.0.md>

**EXERCÍCIO PRÁTICO**

Com base no modelo de entidade relacionamento do projeto XPTO (Figura 1), implemente as tarefas apresentadas neste documento. Use o código disponibilizado (API), bem como as informações do banco de dados fornecidas para execução das melhorias.

O banco de dados reflete o modelo apresentado na Figura 1. Dessa forma, concentre seus esforços no código disponibilizado (API).



**Figura 1 - Modelo de entidade relacionamento**

**CENÁRIO ATUAL DA APLICAÇÃO**

* O banco de dados fornecido contempla o modelo apresentado na Figura 1;
* O código disponibilizado oferta as funcionalidades relacionadas às entidades professor e curso.

**TAREFAS - ESPECIFICAÇÃO DE MODELOS E RELACIONAMENTOS (SEQUELIZE)**

* Especifique no código disponibilizado o modelo da entidade aluno;
* Especifique no código disponibilizado o modelo da entidade aula;
* Configure os relacionamentos dos modelos envolvendo os as entidades adicionadas;
* Especifique o relacionamento entre curso e aluno, através da entidade curso\_aluno.

**TAREFAS - CRUD DE ALUNO**

* Codifique o serviço /aluno (GET), listar alunos;
* Codifique o serviço /aluno (POST), registrar aluno;
* Codifique o serviço /aluno/{aluno\_id} (GET), resgatar aluno;
* Codifique o serviço /aluno/{aluno\_id} (PUT), atualizar aluno;
* Codifique o serviço /aluno/{aluno\_id} (DELETE), remover aluno.

**TAREFAS - CRUD DE AULA**

* Codifique o serviço /curso/{curso\_id}/aula (GET), listar aulas;
* Codifique o serviço /curso/{curso\_id}/aula (POST), registrar aula;
* Codifique o serviço /curso/{curso\_id}/aula/{aula\_id} (GET), resgatar aula;
* Codifique o serviço /curso/{curso\_id}/aula/{aula\_id} (PUT), atualizar aula;
* Codifique o serviço /curso/{curso\_id}/aula/{aula\_id} (DELETE), remover aula.

**TAREFAS - MATRICULAR ALUNO EM CURSO**

* Codifique o serviço /curso/{curso\_id}/matricular/{aluno\_id} (POST), matricular aluno no curso especificado;
* Codifique o serviço /curso/{curso\_id}/aluno (GET), listar alunos matriculados no curso especificado.

**TAREFAS - BÔNUS**

* Versionar alterações em repositório GIT (e.g., github, gitlab);
* Hospedar aplicação em algum servidor / plataforma (e.g., heroku, google, amazon);
* Apresentar documentação automática do swagger (modificada pela melhoria);
* Adicionar pacote de log para sinalizar quando um aluno for registrado (nome + data);