# Projeto da API de Controle de Escola

- No pacote br.com.fuctura.escola.model, crie as classes de entidade:

Aluno
Professor
Curso
Turma
Matricula
TipoAluno (Enum)
TipoProfessor (Enum)

A classe Aluno tem os seguintes atributos:

```
private Long id;
private String cpf;
private String nome;
private String email;
private String fone;
private String tipo = TipoAluno.CONVENCIONAL.toString();
```

A classe Professor tem os seguintes atributos:

```
private Long id;
private String cpf;
private String nome;
private String email;
private Float valorHora;
private String certificados;
private String tipo = TipoProfessor.TITULAR.toString();
```

A classe Curso tem os seguintes atributos:

```
private Long id;
private String nome;
private String requisitos;
private Integer cargaHoraria;
private Float preco;
```

A classe Turma tem os seguintes atributos:

```
private Long id;
private String nome;
private Professor professor;
private Curso curso;
private Integer cargaHoraria;
```

A classe Matricula tem os seguintes atributos:

private Long id;
private Turma turma;
private Aluno aluno;
private LocalDateTime dataMatricula;

- Crie duas classes do tipo Enum para representar o TipoAluno e TipoProfessor

```
public enum TipoAluno {
   CONVENCIONAL,
   MONITOR;
}

public enum TipoProfessor {
   TITULAR,
   SUBSTITUTO;
}
```

- Para cada uma das classes de modelo, crie um construtor default e um construtor com todos os campos exceto o atributo Id. Crie também os métodos hashCode(), equals() e toString().

- Para cada uma das classes de entidade, faça o mapeamento das classes e atributos através do Spring Data JPA (baixar o arquivo model.zip com esse mapeamento pronto)

- No pacote br.com.fuctura.escola.repository, crie uma interface de Repositório para cada uma das classes de modelo:

AlunoRepository
ProfessorRepository
CursoRepository
TurmaRepository
MatriculaRepository

- No pacote br.com.fuctura.escola.controller, crie um controlador WelcomeController, para personalizar o endpoint com endereço "/", mostrando uma página estática com nome welcome.html que fique localizada na pasta src/main/resources/templates. Nessa página html, apresente uma mensagem de boas vindas ao usuário da API

Fuctura Tecnologia

# Curso de Spring Boot

## Olá Aluno, seja bem vindo ao curso de Spring Boot da Fuctura

Assuntos estudados no curso:

1. Rest
2. MVC
3. API
4. JSON
5. Java
6. Controller
7. JPA

- No pacote br.com.fuctura.escola.controller, crie um controlador para cada uma das classes de modelo

AlunoController
ProfessorController
CursoController
TurmaController
MatriculaController

- Para cada controlador, desenvolva os seguintes serviços REST: listarTodos(), cadastrar(), atualizar(), detalhar(), deletar()

Exemplo para Aluno:
GET: listarAlunos(), detalhar() e exportarRelatorioPDF()
POST: cadastrar()
PUT: atualizar()
DELETE: deletar()

Fazer todos os serviços REST para todos os demais Controladores:
ProfessorController
CursoController
TurmaController
MatriculaController

- Para cada Controlador, será necessário criar uma classe DTO para o método cadastrar, detalhar e atualizar. Exemplo: AlunoForm, AtualizacaoAlunoForm e DetalhesDoAlunoDto. Fazer igual para os outros Controladores

- No arquivo de configuração data.sql, crie comandos SQL para inserção de registros para cada uma das tabelas (baixa o arquivo data.sql com os inserts prontos)

- Após criar todos os métodos HTTP dos controladores, testá-los no Postman os seus resultados. Após testar todos os métodos HTTP no Postman, exportar o arquivo json contendo todos os serviços (escola-control-api.postman_collection.json).

- Para todos os serviços REST dos controladores, documentar os serviços com Swagger Open Api. Após documentar, acesse-os através da url http://localhost:8080/swagger-ui.html

Exemplo:

```
    @Operation(summary = "listarAlunos", description = "listar os alunos da escola")
    public Page<AlunoDto> listaAlunos() {

    @GetMapping("/{id}")
    @Operation(summary = "detalhar", description = "detalha um aluno de acordo com o Id")
    public ResponseEntity<DetalhesDoAlunoDto> detalhar(@PathVariable Long id) {
```

- Nos controladores, é recomendável que você desenvolva um método responsável por exportar todos os registros de cada entidade no formato PDF. Ver imagem de exemplo abaixo:

## Lista de Alunos

| ID | Nome | Cpf | E-mail | Fone | Tipo |
|---|---|---|---|---|---|
| 1 | Alberto | 11111111111 | aluno111@escola.com | 81 1234-5555 | CONVENCIONAL |
| 2 | Bruno | 22222222222 | aluno222@escola.com | 81 1234-5555 | CONVENCIONAL |
| 3 | Carlos | 33333333333 | aluno333@escola.com | 81 1234-5555 | MONITOR |
| 4 | Daniel | 44444444444 | aluno111@escola.com | 81 1234-5555 | CONVENCIONAL |
| 5 | Emerson | 55555555555 | aluno222@escola.com | 81 1234-5555 | CONVENCIONAL |
| 6 | Fernando | 66666666666 | aluno333@escola.com | 81 1234-5555 | MONITOR |
| 7 | Guilherme | 77777777777 | aluno111@escola.com | 81 1234-5555 | CONVENCIONAL |
| 8 | Heitor | 88888888888 | aluno222@escola.com | 81 1234-5555 | CONVENCIONAL |
| 9 | José | 99999999999 | aluno333@escola.com | 81 1234-5555 | MONITOR |
| 10 | Paulo | 44422233344 | aluno333@escola.com | 81 1234-5555 | MONITOR |
| 11 | Roberto | 55599944411 | aluno333@escola.com | 81 1234-5555 | MONITOR |

- Faça a mudança do banco de dados H2 da aplicação para outro tipo de banco de dados relacional, podendo ser MySQL, MariaDB ou PostgreSQL.

- Após tudo pronto, faça o deploy da sua API em algum servidor da sua preferência. Recomendável: Heroku (https://www.heroku.com/ )