Performance Sports

Planejamento Inicial (Backend, Frontend, Seguranca e Deploy)

Documento de referencia para a implementacao do sistema de consultorias esportivas, com foco em seguranca (JWT), escopo por professor, tokens de verificacao/recuperacao, eventos e pagamentos.

Data: 20/01/2026

Stack alvo: Java 21+ (Spring Boot), MySQL, Angular (com modulos), Bootstrap, FontAwesome, Docker.

1. Principios e Requisitos

1.1 Escopo do sistema

O sistema Performance Sports suportara cadastro e gestao de alunos e professores, modalidades, eventos e pagamentos. Todas as chamadas privadas ao backend devem ser autenticadas por JWT.

1.2 Requisitos de seguranca e privacidade

- JWT obrigatorio em 100% das rotas privadas; rotas publicas restritas a autenticacao e webhooks.
- Nenhuma exposicao de dados sensiveis em responses; requests limitados ao minimo necessario.
- Tokens de cadastro e recuperacao/alteracao de senha expiram em 1 minuto.
- 5 tokens invalidos geram cooldown de 10 minutos antes de novo envio/validacao.
- Escopo por professor: professor nivel USER acessa apenas seus alunos/pagamentos/eventos; ADMIN acessa todos; SUPER_ADMIN acessa tudo.

1.3 Ordem de execucao

- Backend com testes via Postman.
- Frontend Angular (modulos) integrando com a API Spring.
- Aprimoramento de UI/UX e responsividade.
- Teste completo das regras de negocio (E2E).

2. Definition of Done

Um item do backlog so pode ser considerado concluido quando atender simultaneamente aos criterios abaixo:

Categoria	Criterios
Seguranca	Rotas privadas exigem JWT; roles e escopo aplicados; logs sem token/CPF; erros 401/403/404 padronizados.
Privacidade	DTOs sem campos internos (hash, tokens, tentativas, cooldown); CPF mascarado; listagens sem data de nascimento completa.
Regras de negocio	Tokens expiram em 60s; 5 falhas causam cooldown de 10 min; aprovacao: aluno por professor; professor e evento por super admin.
Qualidade	Postman Collection atualizada e verde; migrations aplicam sem erro; testes unitarios minimos para token/cooldown/escopo.

3. Plano de acao (Backlog de alto nivel)

3.1 Epicos - Backend

Epico A - Fundacao (infra + dominio base): setup do projeto, migrations, entidades base e constraints.
Epico B - Autenticacao, tokens e cooldown: registro, confirmacao, login, esqueci senha, reset, alteracao confirmada por e-mail.
Epico C - Autorizacao por role e escopo: aplicacao consistente de permissao e escopo por professor/aluno.
Epico D - Cadastros: modalidades, enderecos (ate 2), professores (aprovacao), alunos (aprovacao).
Epico E - Eventos: criacao por professor, aprovacao SUPER_ADMIN, convites e confirmacoes.
Epico F - Pagamentos: geracao de link PagSeguro, webhook, conciliacao, lembretes e inadimplencia.
Epico G - Dashboards e relatorios: dashboards por role e relatorios consolidados para SUPER_ADMIN.

3.2 Epicos - Frontend

- Epico H - Estrutura Angular (modulos), Bootstrap/FontAwesome, interceptor JWT e guards por role.
- Epico I - Telas e fluxos: home publica, login/cadastro/reset, areas de aluno/professor/super admin.
- Epico J - Aprimoramentos: responsividade, acessibilidade, toasts, paginacao, estados de carregamento.

4. Matriz de permissoes

A matriz abaixo consolida o que cada perfil pode executar. 'Prof USER' = professor nivel USER (apenas seu escopo). 'Prof ADMIN' = professor ADMIN (global).

4.1 Autenticacao e conta (resumo)

Acao	Publico	Autenticado	Observacoes
Registrar aluno (envio codigo)	Sim	-	Resposta neutra; token 60s; tentativas/cooldown.
Confirmar cadastro (codigo)	Sim	-	Consome token; e-mail verificado; muda para PENDING_APPROVAL.
Login	Sim	-	Retorna JWT + perfil minimo nao sensivel.
Esqueci senha (envio codigo)	Sim	-	Resposta neutra; token 60s; tentativas/cooldown.
Reset senha (codigo + nova senha)	Sim	-	Consome token; nao retorna dados sensiveis.
Alterar senha logado (com codigo)	-	Aluno/Prof/Super	Fluxo: solicitar codigo -> confirmar -> alterar.

4.2 Cadastros, eventos e pagamentos (resumo)

Dominio	Acao	Aluno	Prof USER	Prof ADMIN	Super Admin	Escopo/Regra
Modalidades	Listar	Sim	Sim	Sim	Sim	Leitura global
Modalidades	Criar/Editar/Remover	Nao	Nao	Sim	Sim	Global
Alunos	Aprovar/Ativar	Nao	Sim	Sim	Sim	USER: apenas seus alunos
Alunos	Listar	Nao	Sim	Sim	Sim	USER: somente seus alunos
Professores	Aprovar/Ativar	-	Nao	Nao	Sim	Apenas SUPER_ADMIN
Enderecos	Manter ate 2	Sim (proprio)	Sim	Sim	Sim	Hard-limit 2 por usuario
Eventos	Criar (pendente)	Nao	Sim	Sim	Sim	Professor dentro do escopo
Eventos	Validar	Nao	Nao	Nao	Sim	Apenas SUPER_ADMIN
Pagamentos	Gerar link	Nao	Sim	Sim	Sim	USER: apenas seus alunos
Pagamentos	Webhook (gateway)	-	-	-	-	Endpoint publico assinado

5. Checklist de dados sensiveis e padrao de DTO

5.1 Nunca retornar em responses

- Senha, hash, salt, policy interna.
- Tokens de verificacao/reset/alteracao; contadores de tentativas e cooldownUntil.
- JWT/refresh token fora do contexto de login/renovacao (se existir).
- CPF completo (retornar apenas mascarado quando estritamente necessario).
- Payloads internos do gateway de pagamento; stack traces e mensagens tecnicas.

5.2 Requests: limites e excecoes

A senha e inevitavel em endpoints de autenticacao (login/registro/reset/alteracao). Fora disso, requests devem evitar trafego de CPF e quaisquer tokens/codigos.

5.3 Padroes por tipo de resposta

Listagens: somente campos de exibicao (nome, cidade, status, indicadores agregados). Sem CPF e sem data de nascimento completa.

Detalhe: enderecos (ate 2) e CPF mascarado apenas se necessario. Preferir idade ao inves de data completa.

Erros: 400/422 com mensagem curta e fieldErrors; 401/403/404 sem vazamento de ownership/escopo.

6. Modelo de estados

6.1 Usuario

Estado	Descricao	Permite login
PENDING_EMAIL	Criado; aguarda confirmacao de e-mail (token 60s).	Nao
PENDING_APPROVAL	E-mail verificado; aguarda aprovacao (aluno por professor / professor por super admin).	Nao
ACTIVE	Conta ativa e habilitada.	Sim
SUSPENDED	Conta desabilitada por acao administrativa.	Nao

6.2 Token (registro/reset/alteracao)

Atributo	Regra
expiresAt	60 segundos apos criacao
attempts	5 falhas invalidas
cooldownUntil	10 minutos apos a 5a falha
consumed	Token nao pode ser reutilizado

6.3 Evento e inscricoes

Entidade	Estados
Evento	PENDING_VALIDATION -> APPROVED / REJECTED (opcional: CANCELLED)
Inscricao	INVITED -> CONFIRMED_YES / CONFIRMED_NO

6.4 Pagamento

Estado	Descricao
PENDING	Link gerado; aguardando confirmacao do gateway
PAID	Confirmado por webhook
OVERDUE	Vencido (scheduler)
CANCELLED	Cancelado
REFUNDED	Estornado (opcional)

7. Fluxos operacionais e checklist Postman

7.1 Fluxo - Cadastro do aluno

- Registro (publico): cria conta PENDING_EMAIL e envia codigo (60s).
- Confirmacao (publico): valida codigo; muda para PENDING_APPROVAL; aplica tentativas/cooldown.
- Aprovacao (professor): ativa aluno (ACTIVE) e habilita acesso.

Checklist Postman:
- Confirmar dentro de 60s: sucesso.
- Confirmar apos 60s: expirado.
- 5 codigos errados: cooldown 10 min.

7.2 Fluxo - Esqueci senha e reset

- Solicitar reset (publico): resposta neutra; envia codigo (60s).
- Reset (publico): valida codigo; troca senha; consome token.

Checklist Postman:
- Reset dentro de 60s: sucesso.
- Token expirado: falha.
- 5 tentativas invalidas: cooldown 10 min.

7.3 Fluxo - Eventos

- Professor cria evento (JWT): status PENDING_VALIDATION.
- SUPER_ADMIN valida (JWT): status APPROVED.
- Sistema envia convites por e-mail e registra inscricoes como INVITED.
- Aluno confirma SIM/NAO: atualiza inscricao.

Checklist Postman:
- Professor USER cria apenas com seus alunos (escopo).
- Aprovacao apenas SUPER_ADMIN.
- Aluno confirma apenas propria inscricao.

7.4 Fluxo - Pagamentos

- Professor gera link (JWT): cria pagamento PENDING e armazena referencia do gateway.
- Webhook (publico, assinado): atualiza PAID/CANCELLED.
- Scheduler marca OVERDUE e dispara lembretes.

Checklist Postman:
- Sem JWT: proibido gerar link.
- Professor USER: apenas seus alunos.
- Webhook idempotente: nao duplica eventos/atualizacoes.

8. Deploy - roteiro didatico (visao geral)

8.1 Pre-requisitos

- Projetos separados: backend (Spring) e frontend (Angular).
- Docker funcionando localmente (MySQL + API).
- Variaveis de ambiente: DB, JWT, SMTP, URLs publicas.
- Colecao Postman verde antes do deploy.

8.2 Sequencia recomendada

- Deploy do banco e backend (Railway/Render ou equivalente).
- Validar endpoints publicos e privados via Postman no ambiente cloud.
- Deploy do frontend (Vercel ou equivalente) apontando para a URL do backend.
- Ajustar CORS/HTTPS e validar fluxos end-to-end.

8.3 Observabilidade minima

- Health check (ex.: actuator) e logs estruturados sem dados sensiveis.
- Auditoria de acoes criticas (aprovacoes, geracao de links, validacao de eventos).