FatecWeek — Documentação do Projeto

Projeto: fatec_proj_dw3 v1.0.0
Descrição: Aplicação web para eventos/palestras com autenticação Google, inscrições, presenças e painel administrativo.
Data de geração: 01/11/2025, 19:50:39

Stack e Dependências

• Node.js + Express
• MongoDB + Mongoose
• Passport Google OAuth 2.0 (login com Google)
• PDFKit (geração de PDF)
• QRCode (qrcode)
• Nodemailer / SendGrid (adapter de email)
• Frontend: HTML/CSS/JS estático em public/

Dependências (package.json)

• @azure/msal-browser@^4.25.1
• @azure/msal-node@^3.8.0
• @azure/msal-react@^3.0.20
• @sendgrid/mail@^8.1.6
• bcrypt@^6.0.0
• cors@^2.8.5
• dotenv@^17.2.3
• express@^5.1.0
• express-session@^1.18.2
• mongoose@^8.18.1
• nodemailer@^7.0.10
• passport@^0.7.0
• passport-azure-ad@^4.3.5
• passport-google-oauth20@^2.0.0
• pdfkit@^0.17.2
• qrcode@^1.5.4

Arquitetura e Organização

• src/app.js — bootstrap do servidor, middlewares e rotas
• src/models — Mongoose Schemas (Usuario, Participante, Docente, Palestra, Inscricao, Presenca)
• src/routes — Rotas REST (auth, palestras, presenca, etc.)
• src/utils — Utilitários (EmailAdapter, presencaStrategies, location, UserFactory)
• public/ — Páginas estáticas (admin.html, dashboards, login, etc.)
• scripts — automações (criar admin, dropar índice RA, gerar PDF)

Padrões de Projeto Aplicados

Factory — UserFactory

Centraliza a construção/normalização de objetos de usuário (Usuario, Participante, Docente).

Strategy — Presença

Encapsula estratégias de verificação de presença (GPS e QR). Integrado em rotas de presença.

Adapter — EmailAdapter

Abstrai o provedor de email (SendGrid/SMTP) para envio padronizado de emails e certificados.

Autenticação e Autorização

• Login via Google OAuth — /api/auth/google e callback
• Sessão Express-Session; endpoint /api/auth/me para checagem

• Middleware ensureAdmin/ensureDocenteOuAdmin para rotas restritas
• Fluxo completar perfil — /api/auth/completar (aluno/docente)
• Admin Master — criado via script npm run create-admin

Administração

• Painel: public/admin.html
• Gerencia palestras (CRUD, QR Code, iniciar/finalizar)
• Lista presenças por palestra
• Gerencia inscrições
• Gerencia administradores (listar/criar/remover)

Rotas chave:
• GET/POST/PUT/DELETE /api/palestras
• POST /api/palestras/:id/iniciar | /finalizar
• GET /api/usuarios/admins | POST /api/usuarios/admins | DELETE /api/usuarios/admins/:id

Configuração (.env)

• MONGO_URI — conexão MongoDB
• SESSION_SECRET — segredo da sessão Express
• GOOGLE_CLIENT_ID / GOOGLE_CLIENT_SECRET / GOOGLE_REDIRECT_URI
• EMAIL_PROVIDER=sendgrid | smtp, SENDGRID_API_KEY, EMAIL_FROM
• ADMIN_BOOTSTRAP_KEY (opcional para bootstrap inicial)

Principais Endpoints

Auth

• GET /api/auth/google !’ OAuth Google
• GET /api/auth/google/callback !’ processamento e redirects
• GET /api/auth/me !’ sessão atual
• POST /api/auth/logout !’ encerra sessão
• POST /api/auth/completar !’ completa perfil (aluno/docente)

Palestras

• GET /api/palestras — lista
• POST /api/palestras — cria
• GET /api/palestras/:id — detalha
• PUT /api/palestras/:id — atualiza
• DELETE /api/palestras/:id — remove
• POST /api/palestras/:id/iniciar — inicia
• POST /api/palestras/:id/finalizar — finaliza

Presenças

• POST /api/presenca/registrar — registra presença por QR/GPS (exemplo)
• GET /api/presenca/palestra/:id — lista presenças por palestra

Como Rodar o Projeto

• 1) Configure o .env (MONGO_URI, Google OAuth, email provider)
• 2) Instale dependências: npm install
• 3) Inicie: npm run dev
• 4) Crie Admin Master: npm run create-admin -- --email="seu-email" --senha="suaSenha"
• 5) Acesse http://localhost:3000/admin.html e faça login com o Google

Notas e Próximos Passos

• Habilitar SendGrid e validar envio de certificado
• Adicionar testes E2E dos principais fluxos
• Higienizar lógica legacy de bootstrap admin se não mais usada
• Melhorias UX no admin (validações, atalhos, colunas responsivas)