



Curso Mestrado em Engenharia Informática e Tecnologias Web

Unidade Curricular: Programação Web Avançada

Ano Letivo: 2025/2026

Aluno: Weber Marcelo Guirra de Souza – UTAD: al77734

Tarefa 5.1

Projeto Final - Especificação de Aplicação Web

Sumário

Introdução.....	1
Arquitetura.....	2
Diagrama de Componentes (Arquitetura).....	2
Tecnologias.....	4
Front-end	4
Back-end.....	4
Design da Interface (Mockups e Fluxo de Navegação).....	5
Mockups	5
Padrões	10
Padrões arquiteturais e de organização.....	10
Segurança: autenticação e autorização	11
Banco de dados.....	13
Modelo de Dados (ER simplificado).....	13
Modelo de dados (PostgreSQL) — Dicionário de Dados	13
Casos de uso	16
Modelo de Casos de Uso	17
Referências	18

Índice:

Figura 1 - Diagrama de Componentes - Macro	3
Figura 2 - Mockup – Página Inicial - Não autenticada.....	5
Figura 3 - Mockup – Login Google OAuth 2.0.....	6
Figura 4 - Mockup – Listagem de Propostas.....	7
Figura 5 - Mockup – Detalhe de Proposta.....	7
Figura 6 - Mockup – Formulário de Edição de Proposta	8
Figura 7 - Mockup – Formulário de Inclusão de Nova de Proposta	9
Figura 8 - Modelo de dados (ER)	13
Figura 9 - Diagrama de Casos de Uso	17

Introdução

Este relatório visa apresentar a especificação técnica e as principais decisões arquiteturais do projeto final da unidade curricular Programação Web Avançada. Essa unidade curricular faz parte do Mestrado em Engenharia Informática e Tecnologias Web, que é oferecido no ano letivo de 2025/2026. O curso é uma parceria entre a Universidade Aberta e a Universidade de Trás-os-Montes e Alto Douro.

O projeto consiste na concepção de uma aplicação web destinada à gestão de propostas de temas para projeto final de curso, permitindo que docentes, após autenticação, possam criar, editar e remover propostas, bem como associar coorientadores, alunos e palavras-chave a cada proposta de tema.

A aplicação visa resolver um problema comum na área académica: a necessidade de ter uma aplicação para guardar e organizar as propostas de temas de projeto de forma atualizada. Para isso, a solução utiliza uma estrutura de aplicações Web modernas, que separa claramente as responsabilidades entre a parte que o front-end e o back-end, usando uma API REST e um banco de dados relacional, como estudado na Unidade Curricular.

Neste documento são descritas as opções tecnológicas e os padrões de projeto selecionados, bem como a modelagem dos dados e os principais casos de uso do sistema. As decisões apresentadas tiveram como critérios fundamentais a coerência com os conteúdos abordados ao longo da unidade curricular, a viabilidade técnica para uma prova de conceito funcional e a possibilidade de evolução futura da aplicação, mantendo princípios de qualidade, segurança e manutenibilidade do software.

Arquitetura

A aplicação será construída como uma aplicação web em camadas, composta por *front-end* SPA e *back-end* API REST, integrados por HTTP/JSON. A separação em camadas (apresentação, aplicação/serviços e persistência) é adotada para reduzir o acoplamento, facilitar testes e permitir a evolução independente do *front-end* e do back-end. Essa organização favorece uma evolução incremental do sistema, na qual o *front-end* pode avançar em ciclos curtos de usabilidade enquanto o back-end mantém contratos de *endpoints* estáveis e testáveis.

Na prática, isso também simplifica a prova de conceito, porque permite validar rapidamente os casos de uso centrais (criar/editar/listar propostas) sem “misturar” lógica de interface com persistência.

No back-end, a organização segue *Routes* → *Controllers* → *Services* → *Repositories*, isolando responsabilidades: os *controllers* tratam requisições e respostas; *services* concentram regras de negócio; *repositories* encapsulam a persistência em PostgreSQL. A autenticação será delegada ao Google OAuth2, e a aplicação consumirá o token/identidade para controlar acesso às operações de gestão de propostas.

Diagrama de Componentes (Arquitetura)

A Figura 1 apresenta uma visão macro dos principais componentes da solução e suas interações. O objetivo é evidenciar a separação entre a camada de apresentação (SPA), a camada de serviços exposta via API e a persistência, bem como os pontos de integração necessários para autenticação e acesso aos dados.

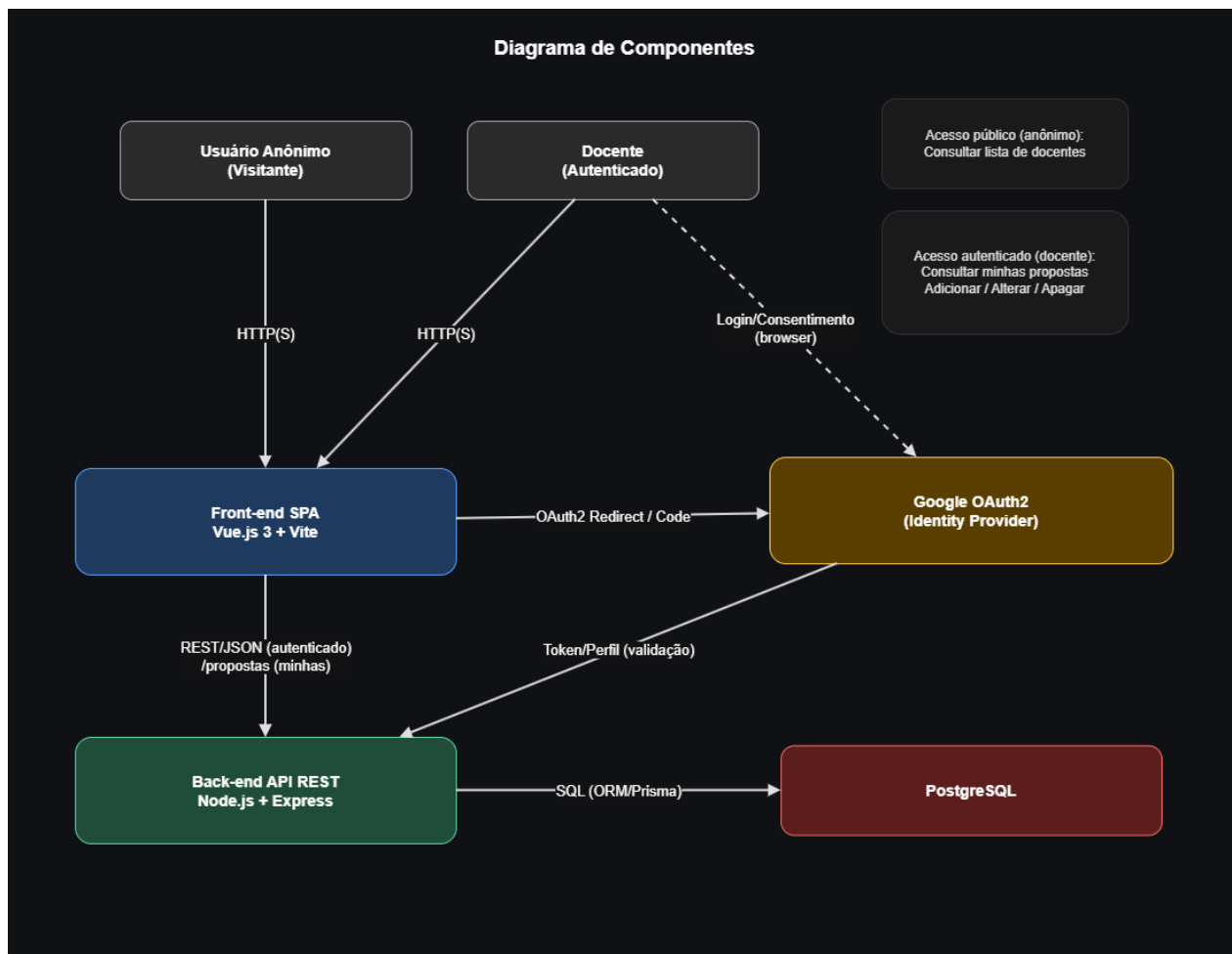


Figura 1 - Diagrama de Componentes - Macro

Tecnologias

Foram selecionadas tecnologias compatíveis com os requisitos da atividade e com as decisões arquiteturais adotadas, mantendo coerência com os conteúdos trabalhados ao longo da unidade curricular.

Front-end

Vue.js 3 + Vite

Vue Router (rotas de navegação)

Pinia (estado global simples)

Bootstrap 5 (layout responsivo e componentes UI)

Validação de formulários: VeeValidate + Yup

Back-end

Node.js + Express (API REST)

PostgreSQL (persistência relacional)

Prisma ORM (migrações, schema versionado, queries tipadas)

OAuth2 Google com Passport.js

OpenAPI/Swagger para documentação dos endpoints

A configuração sensível (por exemplo, credenciais e URLs) será externalizada por variáveis de ambiente, seguindo a prática de separar configuração do código-fonte, o que facilita a execução em diferentes ambientes (desenvolvimento e produção) sem alterar o repositório.

Decidi pela adoção do OAuth2 com Google com o objetivo de equilibrar segurança e pragmatismo. Para o escopo do projeto, delegar autenticação reduz a exposição de credenciais e elimina a necessidade de gerir processos sensíveis como recuperação de senha e armazenamento de *hashes*. Além disso, o login federado é compatível com o contexto acadêmico, pois permite controlar o acesso às operações de gestão de propostas com base em uma identidade confiável e amplamente utilizada.

Design da Interface (Mockups e Fluxo de Navegação)

Embora a presente tarefa tenha como foco principal a especificação arquitetural e funcional da aplicação, considerando a ênfase da unidade curricular nas etapas de concepção e prototipação, optei por incluir um conjunto de mockups de interface como artefato complementar ao projeto.

Esses mockups têm como finalidade tornar mais explícita a forma como os casos de uso definidos ao longo do documento se materializam na interação com o utilizador, facilitando a compreensão da navegação, da organização da informação e dos principais fluxos funcionais do sistema.

As telas apresentadas representam uma proposta de interface alinhada às funcionalidades especificadas, em especial aos casos de uso relacionados à criação, edição e consulta de propostas, servindo como apoio à validação conceitual do sistema e como referência para as etapas posteriores do desenvolvimento.

Mockups

A Figura 2 apresenta a página inicial da aplicação para utilizadores não autenticados. Esta tela funciona como ponto de entrada do sistema, permitindo o acesso às funcionalidades públicas e à autenticação necessária para docentes que pretendam gerir propostas.

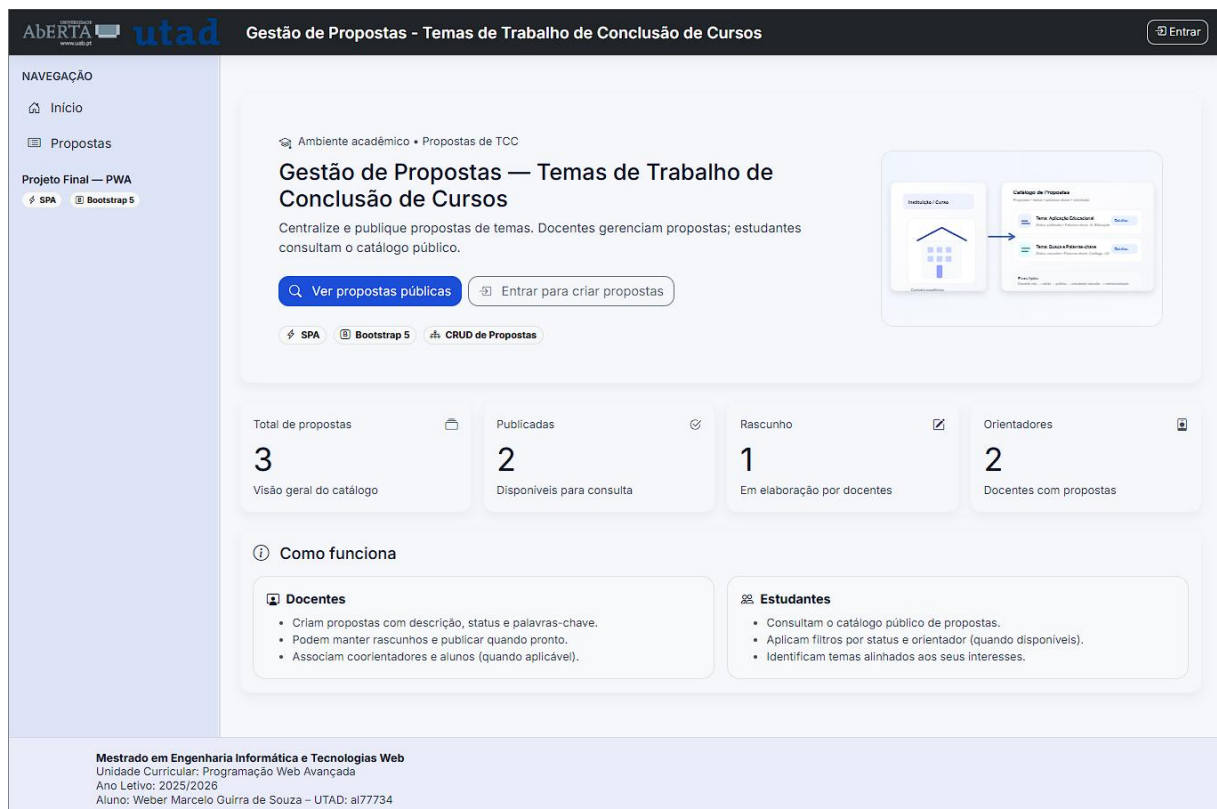


Figura 2 - Mockup – Página Inicial - Não autenticada

A Figura 3 apresenta o *mockup* da interface de autenticação da aplicação, concebida para refletir simultaneamente o caráter de prova de conceito e uma abordagem próxima de um ambiente de produção. A interface adota o mecanismo de autenticação via Google OAuth 2.0, delegando o processo de verificação de identidade ao provedor externo, enquanto a aplicação mantém o controle integral da autorização.



Figura 3 - Mockup – Login Google OAuth 2.0

A Figura 4 ilustra a tela de listagem de propostas, que constitui o principal ponto de navegação para docentes autenticados. A partir desta interface, é possível consultar propostas existentes e aceder aos respetivos detalhes.

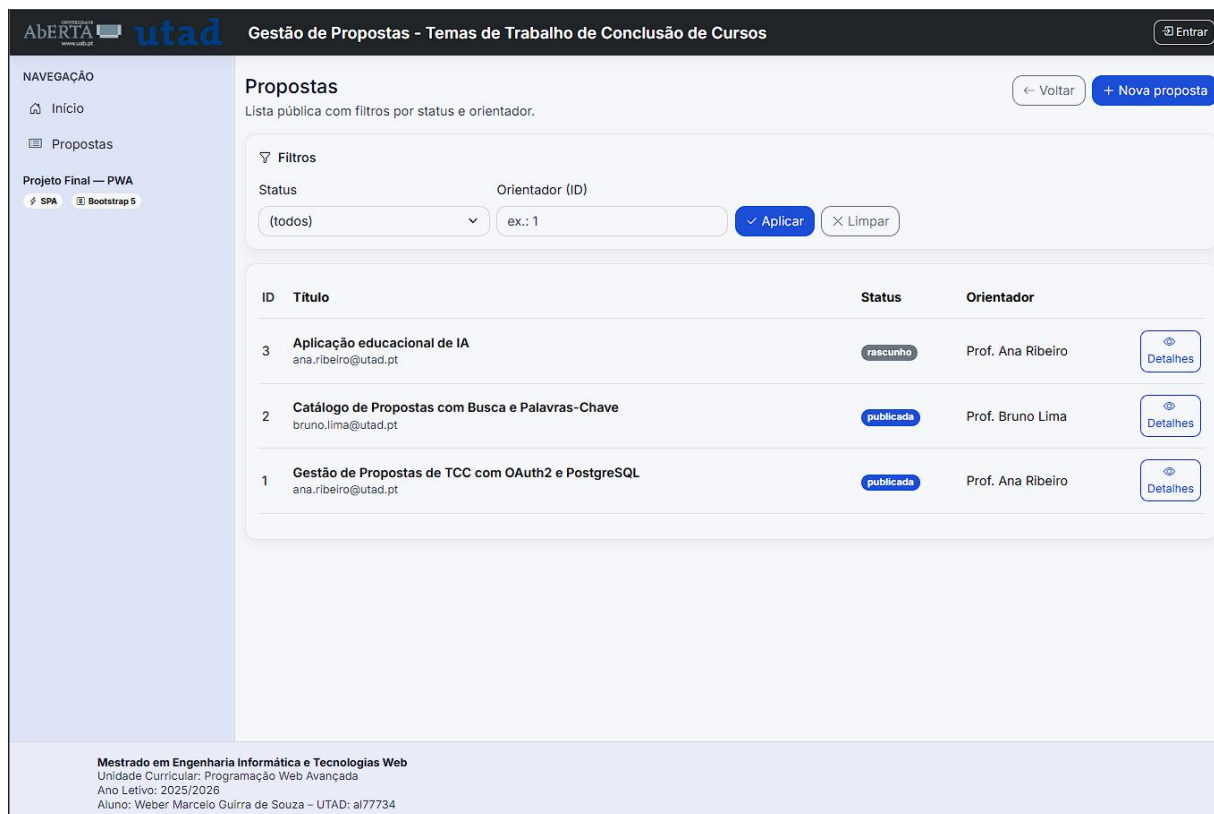


Figura 4 - Mockup – Listagem de Propostas

A Figura 5 apresenta a visualização detalhada de uma proposta de tema, evidenciando as informações essenciais associadas ao projeto e servindo como ponto de partida para ações de edição ou gestão de vínculos, esta tela funciona como transição entre consulta e manutenção.



Figura 5 - Mockup – Detalhe de Proposta

A Figura 6 representa o formulário de edição de proposta, utilizado para a atualização das informações previamente registadas. Esta interface está diretamente associada aos casos de uso de alteração de propostas e gestão dos respetivos vínculos.

AbERTA

utad

Gestão de Propostas - Temas de Trabalho de Conclusão de Cursos

Entrar

NAVEGAÇÃO

Início

Propostas

Projeto Final — PWA

SPA

Bootstrap 5

Editar proposta

Voltar

Preencha os campos principais e associe coorientadores, alunos e palavras-chave.

Dados principais

Título *

Aplicação educacional de IA

Evite títulos genéricos. Seja específico no tema e recorte. 28/200

Status

rascunho

Use **rascunho** enquanto estiver ajustando a proposta.

Descrição

Porposta de trabalho MEIW

Use parágrafos curtos. Isso melhora a legibilidade. 25/2000

Associações

Coorientadores

Limpar

Prof. Ana Ribeiro (ana.ribeiro@utad.pt)

Prof. Bruno Lima (bruno.lima@utad.pt)

Selecione um ou mais docentes por nome (Ctrl/Cmd para seleção múltipla).

Alunos

Limpar

Carla Souza (carla.souza@alunos.utad.pt)

Diego Martins (diego.martins@alunos.utad.pt)

Elisa Ferreira (elisa.ferreira@alunos.utad.pt)

Selecione um ou mais alunos por nome (Ctrl/Cmd para seleção múltipla).

Palavras-chave

api-rest

express

oauth2

postgresql

prisma

vue.js

Lista carregada do BD. Se estiver vazia, verifique o seed ou crie uma proposta contendo palavras-chave.

api-rest

oauth2

Salvar alterações

Cancelar

Mestrado em Engenharia Informática e Tecnologias Web

Unidade Curricular: Programação Web Avançada

Ano Letivo: 2025/2026

Aluno: Weber Marcelo Guirra de Souza – UTAD: al77734

Figura 6 - Mockup – Formulário de Edição de Proposta

Página 8 de 20

A Figura 7 apresenta o formulário de inclusão de nova proposta, concebido para apoiar o processo de criação de temas de projeto. Nesta interface concentram-se as operações de definição do tema e de associação de coorientadores, alunos e palavras-chave.

AbERTA

utad

Gestão de Propostas - Temas de Trabalho de Conclusão de Cursos

Entrar

NAVEGAÇÃO

Início

Propostas

Projeto Final — PWA

SPA

Bootstrap 5

Nova proposta

Voltar

Preencha os campos principais e associe coorientadores, alunos e palavras-chave.

Dados principais

Título *

Ex.: Análise de Dados Educacionais

Evite títulos genéricos. Seja específico no tema e recorte.

0/200

Status

rascunho

Use **rascunho** enquanto estiver ajustando a proposta.

Descrição

Descreva objetivos, escopo, tecnologias e resultados esperados...

Use parágrafos curtos. Isso melhora a legibilidade.

0/2000

Associações

Coorientadores

Limpar

Prof. Ana Ribeiro (ana.ribeiro@utad.pt)

Prof. Bruno Lima (bruno.lima@utad.pt)

Selecione um ou mais docentes por nome (Ctrl/Cmd para seleção múltipla).

Alunos

Limpar

Carla Souza (carla.souza@alunos.utad.pt)

Diego Martins (diego.martins@alunos.utad.pt)

Elisa Ferreira (elisa.ferreira@alunos.utad.pt)

Selecione um ou mais alunos por nome (Ctrl/Cmd para seleção múltipla).

Palavras-chave

api-rest

express

oauth2

postgresql

prisma

vue.js

Lista carregada do BD. Se estiver vazia, verifique o seed ou crie uma proposta contendo palavras-chave.

Criar proposta

Cancelar

Mestrado em Engenharia Informática e Tecnologias Web

Unidade Curricular: Programação Web Avançada

Ano Letivo: 2025/2026

Aluno: Weber Marcelo Guirra de Souza – UTAD: al77734

Figura 7 - Mockup – Formulário de Inclusão de Nova de Proposta

Página 9 de 20

Padrões

A adoção de padrões no projeto foi orientada pela necessidade de manter o código organizado e com responsabilidades bem delimitadas, especialmente em uma aplicação com front-end desacoplado de uma API e persistência relacional. Neste contexto, os padrões não são tratados como uma formalidade, mas como um meio prático de sustentar manutenibilidade, clareza e evolução incremental, reduzindo dependências indevidas entre camadas e componentes. No back-end, essa organização dialoga com a lógica do MVC adaptado ao contexto de APIs, conforme discutido por Queirós e Portela (2020), priorizando a separação de responsabilidades em vez de uma adoção rígida do modelo clássico.

Padrões arquiteturais e de organização

- Camadas (Layered Architecture): UI (SPA) → API (Controllers/Services) → Persistência (Repositories/ORM).
- MVC (adaptado ao Express)
- Padrões de projeto (no back-end)
- Service Layer: regras de negócio centralizadas
- Repository Pattern: operações no PostgreSQL encapsuladas
- DTO / ViewModel: padroniza payloads de entrada/saída e evita “vazar” modelo do banco.
- Middleware Chain (Express): autenticação OAuth2, autorização, validação e tratamento de erros.
- Facade: simplifica fluxos mais longos (ex.: AuthFacade para login + obtenção de perfil).

Segurança: autenticação e autorização

A autenticação foi delegada a um provedor externo confiável, recorrendo ao Google OAuth 2.0, enquanto a autorização é inteiramente controlada pela aplicação, com base no seu modelo de dados e nas regras de negócio do domínio. Esta abordagem reduz a complexidade associada à gestão de credenciais, reforça a segurança da solução e assegura aderência às restrições funcionais definidas no enunciado.

O Google OAuth 2.0 é utilizado exclusivamente como mecanismo de autenticação, isto é, para comprovar a identidade do utilizador. O Google não atribui papéis nem permissões no contexto da aplicação; limita-se a fornecer atributos básicos do perfil autenticado, como o endereço de correio eletrónico e o nome.

O fluxo de autenticação implementado é o seguinte:

1. O utilizador inicia o processo de autenticação a partir da interface Web, selecionando a opção “Entrar com Google”.
2. O front-end redireciona o utilizador para o endpoint `/auth/google` do back-end.
3. O back-end inicia o fluxo OAuth 2.0 e redireciona o utilizador para o serviço de autenticação do Google.
4. Após autenticação e consentimento, o Google redireciona o utilizador para o endpoint de callback `/auth/google/callback`.
5. O back-end valida a resposta do Google e obtém os dados de perfil do utilizador.
6. Com base nesses dados, a aplicação executa o processo de autorização interna.
7. Sendo a autenticação autorizada, o back-end emite um JSON Web Token (JWT) próprio da aplicação.
8. O utilizador é redirecionado para o front-end com o token, que passa a ser utilizado nas chamadas subsequentes aos endpoints protegidos.

A utilização de JWT permite uma arquitetura stateless, adequada a aplicações Web modernas, facilitando a escalabilidade e a separação clara entre front-end e back-end.

A autorização não é delegada ao Google, sendo implementada internamente através de um modelo de Role-Based Access Control (RBAC), alinhado com o modelo de dados da aplicação.

A regra central de autorização é definida da seguinte forma:

1. Um utilizador autenticado só é considerado **docente** se o endereço de e-mail devolvido pelo Google existir previamente registado na entidade `docentes` da base de dados.
2. Caso o e-mail autenticado não corresponda a nenhum docente registado, o acesso às funcionalidades protegidas é recusado.
3. O papel (`role`) atribuído ao utilizador é incluído como *claim* no JWT emitido pelo back-end.

Desta forma, a aplicação garante que nem todas as contas Google têm acesso às funcionalidades de docente, cumprindo o requisito de que apenas docentes podem criar, alterar ou remover propostas.

Além da verificação do papel, a aplicação aplica também regras de autorização contextual, nomeadamente: um docente só pode alterar ou remover propostas das quais seja orientador responsável.

Banco de dados

No que diz respeito à persistência, a escolha de um banco de dados relacional, aliada a um ORM com suporte a migrações e versionamento de schema, contribui para a rastreabilidade das alterações no modelo de dados e para a consistência entre ambientes.

Modelo de Dados (ER simplificado)

Para sustentar os casos de uso previstos, foi definido um modelo relacional centrado na entidade “proposta”, associada obrigatoriamente a um docente orientador. As relações com coorientadores, alunos e palavras-chave foram modeladas de forma a refletir cardinalidades do enunciado e permitir evolução do catálogo de propostas sem perda de integridade.

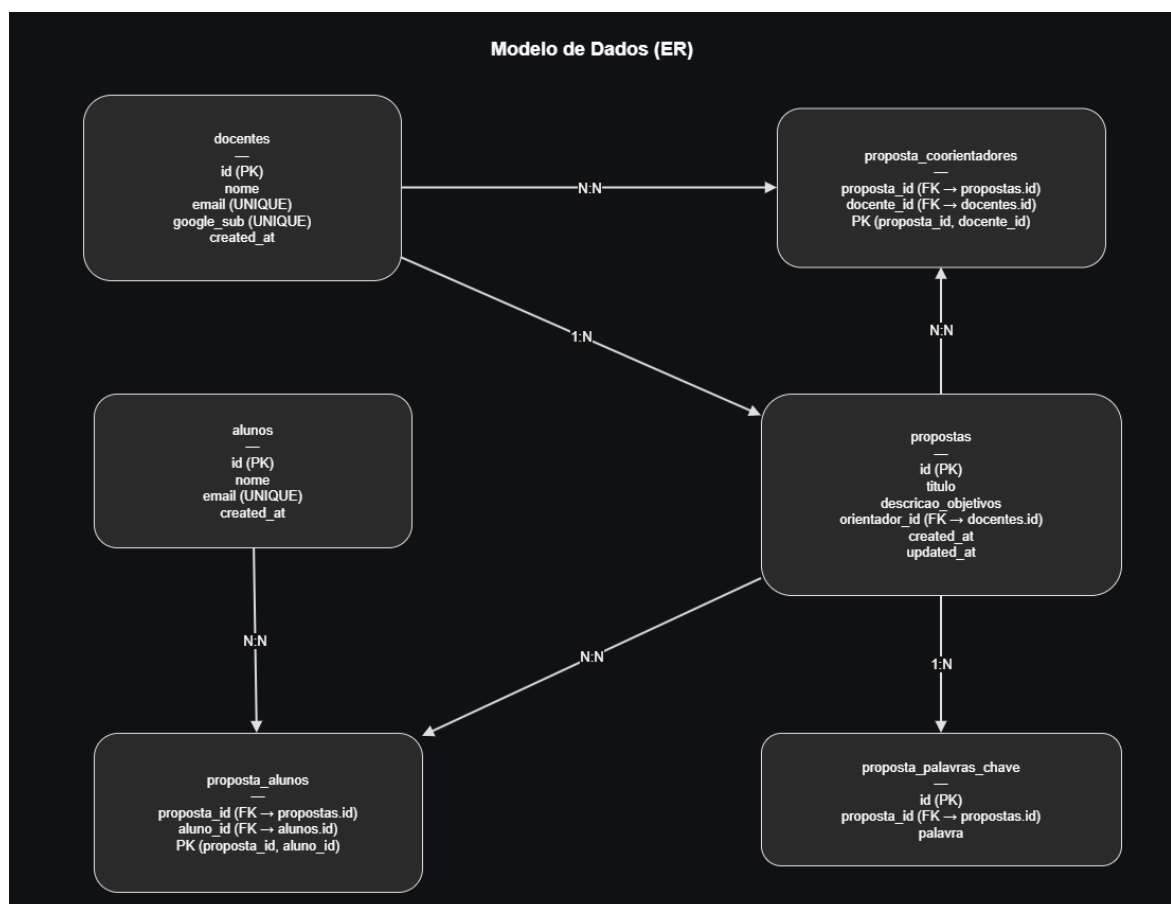


Figura 8 - Modelo de dados (ER)

Modelo de dados (PostgreSQL) — Dicionário de Dados

A seguir apresenta-se o dicionário de dados simplificado, descrevendo as tabelas principais e suas restrições. A intenção é tornar explícitas as chaves primárias, relacionamentos e campos essenciais para a implementação com PostgreSQL e para a camada de persistência via ORM.

Tabela: docentes			
Campo	Tipo	Chave / Restrição	Descrição
id	UUID / INT	PK	Identificador do docente
nome	VARCHAR	NOT NULL	Nome do docente
email	VARCHAR	UNIQUE, NOT NULL	Email institucional
google_sub	VARCHAR	UNIQUE	Identificador Google (OAuth2)
created_at	TIMESTAMP	NOT NULL	Data de criação

Tabela: alunos			
Campo	Tipo	Chave / Restrição	Descrição
id	UUID / INT	PK	Identificador do aluno
nome	VARCHAR	NOT NULL	Nome do aluno
email	VARCHAR	UNIQUE, NOT NULL	Email do aluno
created_at	TIMESTAMP	NOT NULL	Data de criação

Tabela: propostas			
Campo	Tipo	Chave / Restrição	Descrição
id	UUID / INT	PK	Identificador da proposta
titulo	VARCHAR	NOT NULL	Título do tema
descricao_objetivos	TEXT	NOT NULL	Descrição e objetivos
orientador_id	UUID / INT	FK → docentes.id, NOT NULL	Docente orientador
created_at	TIMESTAMP	NOT NULL	Criação
updated_at	TIMESTAMP	NOT NULL	Atualização

Tabela: proposta_coorientadores (0..N)			
Campo	Tipo	Chave / Restrição	Descrição
proposta_id	UUID / INT	FK → propostas.id	Proposta
docente_id	UUID / INT	FK → docentes.id	Docente coorientador
(proposta_id, docente_id)	-	PK composta	Evita duplicidade

Tabela: proposta_alunos (0..N)			
Campo	Tipo	Chave / Restrição	Descrição
proposta_id	UUID / INT	FK → propostas.id	Proposta
aluno_id	UUID / INT	FK → alunos.id	Aluno associado
(proposta_id, aluno_id)	-	PK composta	Evita duplicidade

Tabela: proposta_palavras_chave			
Campo	Tipo	Chave / Restrição	Descrição
id	UUID / INT	PK	Identificador do termo
proposta_id	UUID / INT	FK → propostas.id, NOT NULL	Proposta
palavra	VARCHAR	NOT NULL	Palavra-chave

Casos de uso

A identificação dos casos de uso do sistema teve como ponto de partida a análise do enunciado do problema proposto na atividade, no qual são explicitadas as necessidades funcionais associadas ao contexto acadêmico de gestão de propostas de temas para projeto final de curso. A partir desse enunciado, foram identificados os principais atores envolvidos — docentes e usuários não autenticados — bem como as ações que cada um pode realizar no sistema.

O processo de definição dos casos de uso consistiu em decompor os requisitos descritos no problema em funcionalidades observáveis, relacionadas diretamente às operações de consulta, criação, edição e remoção de propostas. As restrições apresentadas no enunciado, como a obrigatoriedade de autenticação para docentes e o controle de acesso baseado na autoria das propostas, foram consideradas como parte integrante do comportamento do sistema e refletidas nos casos de uso definidos.

Dessa forma, os casos de uso apresentados procuram representar, de forma clara e estruturada, o comportamento funcional esperado da aplicação, servindo como ponte entre os requisitos descritos no problema e as decisões arquiteturais e de implementação adotadas no projeto.

Ator: Docente (autenticado via Google OAuth2)

- Autenticar com Google
- Alterar proposta existente (*somente propostas em que é orientador*)
- Adicionar nova proposta
- Apagar proposta existente (*somente propostas em que é orientador*)
- Consultar minhas propostas
- Consultar detalhes de uma proposta
- Gerir vínculos: coorientadores, alunos e palavras-chave (na criação/edição)

Ator: Visitante (não autenticado)

- Consultar Lista de docentes (consulta pública)
- Consultar docente (detalhe)

Modelo de Casos de Uso

A Figura 9 sintetiza os casos de uso identificados, evidenciando as principais funcionalidades disponíveis para docentes autenticados e para usuários não autenticados. O objetivo é apresentar, de forma visual, o escopo funcional do sistema e as interações esperadas entre atores e aplicação.

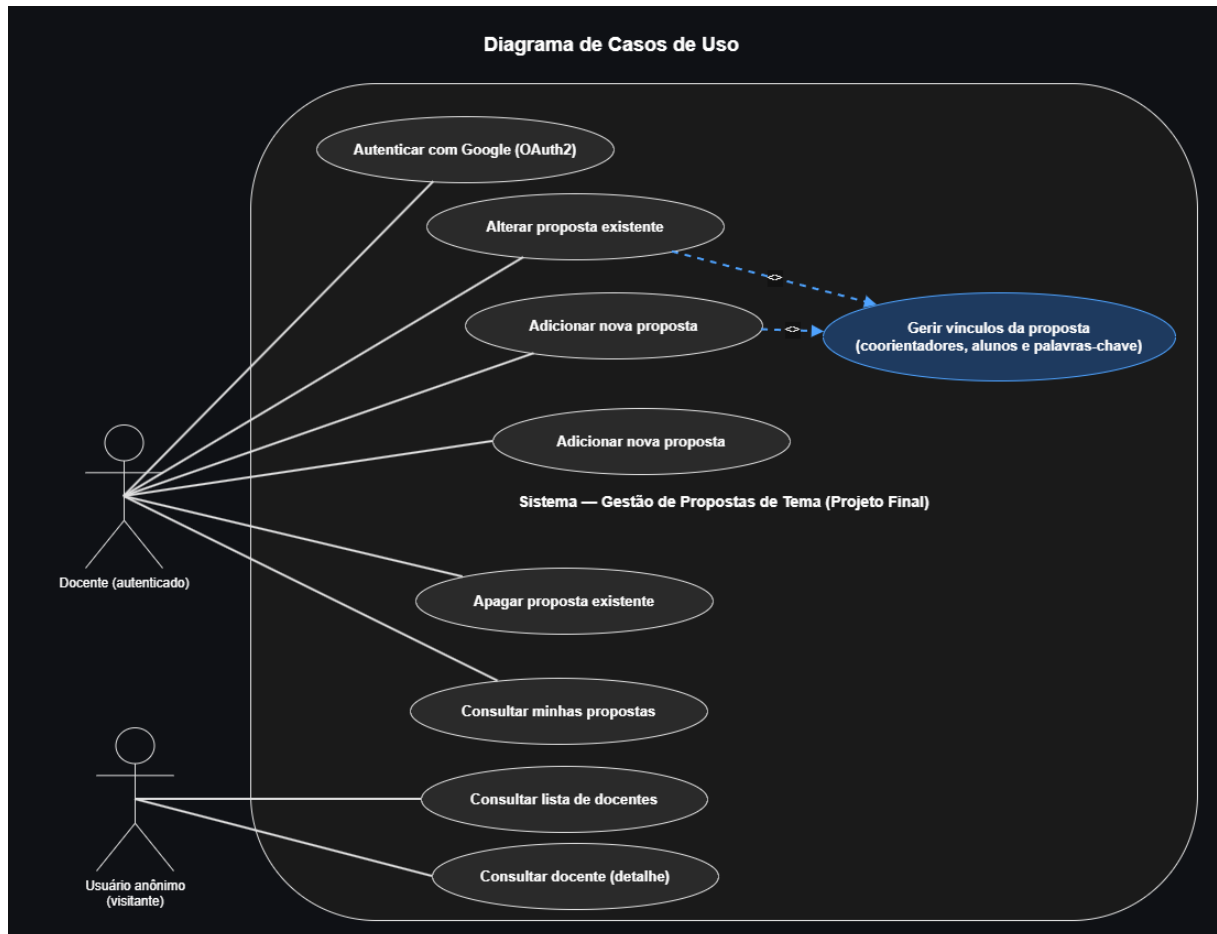


Figura 9 - Diagrama de Casos de Uso

Referências

Queirós, R., & Portela, F. (2020). *Desenvolvimento Avançado para Web* (1º ed.). FCA - Editora de Informática LTDA.