



INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO PIAUÍ
CAMPUS ANGICAL
CURSO SUPERIOR DE TECNOLOGIA EM
ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

DIEGO DE SOUSA OLIVEIRA

**GAMIFY: desenvolvimento de uma plataforma gamificada para acompanhamento de
atividades no Google Classroom**

ANGICAL DO PIAUÍ

2026

DIEGO DE SOUSA OLIVEIRA

GAMIFY: desenvolvimento de uma plataforma gamificada para acompanhamento de atividades no Google Classroom

Trabalho de Conclusão de Curso (relatório técnico de software) apresentado como exigência parcial para obtenção do diploma do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia do Piauí, Campus Angical.

Orientador: Prof. Me. Mike Christian de Sousa Araujo.

ANGICAL DO PIAUÍ

2026

DIEGO DE SOUSA OLIVEIRA

GAMIFY: desenvolvimento de uma plataforma gamificada para acompanhamento de atividades no Google Classroom

Trabalho de Conclusão de Curso (relatório técnico de software) apresentado como exigência parcial para obtenção do diploma do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia do Piauí, Campus Angical.

Aprovada em: 28/07/2026.

BANCA EXAMINADORA

Prof. Me. Mike Christian de Sousa Araujo (Orientador)
Instituto Federal do Piauí (IFPI)

Prof. Me. José Soares da Silva Neto
Instituto Federal do Piauí (IFPI)

Prof. Dr. Thiago Reis da Silva
Instituto Federal do Maranhão (IFMA)

RESUMO

Este relatório técnico apresenta o desenvolvimento de uma plataforma web gamificada integrada ao Google Classroom, destinada ao acompanhamento de atividades, notas e indicadores de desempenho acadêmico. A solução foi concebida para complementar os recursos de gestão disponibilizados pelo Google Classroom, transformando dados acadêmicos em elementos visuais de acompanhamento, como pontos de experiência, níveis, barras de progresso, rankings e painéis de desempenho. O processo de desenvolvimento compreendeu a definição dos requisitos de gamificação, a modelagem do sistema, a implementação da arquitetura cliente-servidor, a integração com serviços Google, a persistência dos dados e a implantação da aplicação em ambiente web. O backend foi desenvolvido com Python, Django e Django REST Framework, enquanto a interface utilizou templates Django, HTML, CSS, JavaScript e Bootstrap. A autenticação foi implementada com conta Google e OAuth 2.0, e a sincronização de cursos, atividades, submissões e notas foi realizada por meio da Google Classroom API. Os dados processados são armazenados em PostgreSQL e utilizados na geração dos indicadores gamificados. A verificação técnica das funcionalidades foi realizada por meio de testes automatizados do Django, utilizando dados e respostas simuladas para evitar dependência direta da API externa durante a execução dos testes. Os cenários documentados confirmaram a integridade da persistência e dos relacionamentos de dados, a resolução de rotas, a proteção de acesso às funcionalidades, a precisão dos cálculos de gamificação e a estrutura adequada das respostas da API REST. Como resultado, foi obtido um protótipo capaz de integrar informações acadêmicas do Google Classroom e apresentá-las em uma interface gamificada.

Palavras-chave: gamificação; Google Classroom; Django; aplicação web; integração de sistemas.

ABSTRACT

This technical report presents the development of a gamified web platform integrated with Google Classroom for monitoring assignments, grades, and academic performance indicators. The solution was designed to complement the management resources provided by Google Classroom by transforming academic data into visual monitoring elements, including experience points, levels, progress bars, rankings, and performance dashboards. The development process comprised the definition of gamification requirements, system modeling, implementation of a client–server architecture, integration with Google services, data persistence, and deployment in a web environment. The backend was developed with Python, Django, and Django REST Framework, while the interface was built with Django templates, HTML, CSS, JavaScript, and Bootstrap. Authentication was implemented through Google accounts and OAuth 2.0, and courses, assignments, submissions, and grades were synchronized through the Google Classroom API. The processed data are stored in PostgreSQL and used to generate gamified indicators. The functionalities were technically verified through Django automated tests using simulated data and responses to avoid direct dependence on the external API during test execution. The documented test scenarios confirmed the integrity of data persistence and relationships, the correct resolution of routes, the protection of access to system functionalities, the accuracy of the gamification calculations, and the proper structure of the REST API responses. The resulting functional prototype can integrate academic information from Google Classroom and present it through a gamified interface.

Keywords: gamification; Google Classroom; Django; web application; systems integration.

LISTA DE FIGURAS

Figura 1 – Diagrama de casos de uso do sistema	22
Figura 2 – Diagrama de classes do sistema	23
Figura 3 – Arquitetura em camadas da solução	24
Figura 4 – Topologia de implantação da aplicação	30
Figura 5 – Diagrama entidade-relacionamento da plataforma	33
Figura 6 – Tela inicial antes da autenticação	34
Figura 7 – Tela de autenticação por conta Google	34
Figura 8 – Dashboard do usuário autenticado	35
Figura 9 – Indicadores de nível e progresso	36
Figura 10 – Cursos sincronizados do usuário	36
Figura 11 – Seleção de disciplina para o ranking	37
Figura 12 – Destaque das maiores pontuações	37
Figura 13 – Classificação geral do ranking	38
Figura 14 – Seleção de disciplina para análise	38
Figura 15 – Painel e gráfico de desempenho	39
Figura 16 – Tela inicial de busca de atividades	40
Figura 17 – Busca de novas atividades	40
Figura 18 – Atualização de nota por atividade	41

LISTA DE TABELAS

Tabela 1 – Matriz dos elementos de gamificação	18
Tabela 2 – Requisitos funcionais da plataforma	19
Tabela 3 – Requisitos de gamificação e formas de verificação	20
Tabela 4 – Requisitos não funcionais e formas de verificação	20
Tabela 5 – Restrições técnicas da plataforma	21
Tabela 6 – Resumo das tecnologias utilizadas	29
Tabela 7 – Cenários documentados da aplicação web	43
Tabela 8 – Cenários documentados da API REST	43

LISTA DE ABREVIATURAS E SIGLAS

API	<i>Application Programming Interface</i>
CSS	<i>Cascading Style Sheets</i>
DNS	<i>Domain Name System</i>
DRF	<i>Django REST Framework</i>
HTML	<i>HyperText Markup Language</i>
HTTP	<i>HyperText Transfer Protocol</i>
HTTPS	<i>HyperText Transfer Protocol Secure</i>
IP	<i>Internet Protocol</i>
JSON	<i>JavaScript Object Notation</i>
LTS	<i>Long Term Support</i>
MVT	<i>Model-View-Template</i>
OAuth	<i>Open Authorization</i>
ORM	<i>Object-Relational Mapping</i>
REST	<i>Representational State Transfer</i>
RF	Requisito Funcional
RNF	Requisito Não Funcional
SSH	<i>Secure Shell</i>
UML	<i>Unified Modeling Language</i>
VPS	<i>Virtual Private Server</i>
WSGI	<i>Web Server Gateway Interface</i>
XP	<i>Experience Points</i>

SUMÁRIO

1	INTRODUÇÃO	10
1.1	OBJETIVOS	10
1.1.1	Objetivo Geral	11
1.1.2	Objetivos Específicos	11
1.2	PROBLEMÁTICA	11
1.3	JUSTIFICATIVA	11
1.4	ORGANIZAÇÃO DO DOCUMENTO	12
2	REFERENCIAL TEÓRICO	13
2.1	ENGENHARIA DE REQUISITOS	13
2.1.1	Requisitos Funcionais	13
2.1.2	Requisitos Não Funcionais	13
2.2	GAMIFICAÇÃO	14
2.3	MODELAGEM DE SISTEMAS	14
2.4	DIAGRAMA DE CASOS DE USO	14
2.5	DIAGRAMA DE CLASSES	15
2.6	MODELAGEM DE BANCO DE DADOS	15
2.7	ARQUITETURA CLIENTE-SERVIDOR	15
2.8	TESTES DE SOFTWARE	15
2.9	DOMÍNIO E DNS	16
3	PROCESSO DE DESENVOLVIMENTO DO SOFTWARE	17
3.1	DEFINIÇÃO DOS ELEMENTOS DE GAMIFICAÇÃO	17
3.2	LEVANTAMENTO E ESPECIFICAÇÃO DOS REQUISITOS	18
3.2.1	Requisitos Funcionais	18
3.2.2	Requisitos Não Funcionais	19
3.2.3	Restrições Técnicas	21
3.3	MODELAGEM DO SISTEMA	21
3.3.1	Arquitetura Definida	23
3.4	IMPLEMENTAÇÃO DA SOLUÇÃO	24
3.4.1	Implementação da Integração com o Google Classroom	25
3.4.2	Implementação das Regras de Gamificação	25
3.4.3	Implementação das Interfaces e da API REST	25
3.5	PREPARAÇÃO DO AMBIENTE E GESTÃO DAS DEPENDÊNCIAS	25
3.6	IMPLANTAÇÃO	26
3.7	ESTRATÉGIA DE TESTES	26

4	PLATAFORMA GAMIFY	27
4.1	STACK TECNOLÓGICA	27
4.1.1	Python	27
4.1.2	Django	27
4.1.3	Django REST Framework	27
4.1.4	HTML, CSS, JavaScript, Bootstrap e Templates Django	27
4.2	BIBLIOTECAS E COMPONENTES DE INTEGRAÇÃO	28
4.3	PERSISTÊNCIA DE DADOS	28
4.4	SERVIÇOS EXTERNOS E APIS	28
4.5	FERRAMENTAS DE DESENVOLVIMENTO E MODELAGEM	28
4.6	GESTÃO DO AMBIENTE E DEPENDÊNCIAS	29
4.7	INFRAESTRUTURA DE IMPLANTAÇÃO	29
4.7.1	Infraestrutura	29
4.7.2	Preparação da VPS	30
4.7.3	Configuração do Servidor Web	31
4.7.4	Configuração do Django	31
4.7.5	Banco de Dados e Migrações	31
4.7.6	Domínio e DNS	32
4.8	ARQUITETURA DA SOLUÇÃO	32
4.9	MODELO DE DADOS	32
4.10	INTERFACES E FLUXOS DE UTILIZAÇÃO	33
4.10.1	Acesso e Autenticação	34
4.10.2	Dashboard e Cursos	35
4.10.3	Ranking por Disciplina	37
4.10.4	Análise de Desempenho	38
4.10.5	Sincronização de Disciplinas, Atividades e Notas	39
5	VERIFICAÇÃO TÉCNICA DO SOFTWARE	42
5.1	CONFIGURAÇÃO DO AMBIENTE DE TESTES	42
5.2	TESTES DA APLICAÇÃO WEB	42
5.3	TESTES DA API REST	43
5.4	RESULTADOS E LIMITAÇÕES DOS TESTES	43
6	CONSIDERAÇÕES FINAIS	45
6.1	RESULTADOS TÉCNICOS ALCANÇADOS	45
6.2	LIMITAÇÕES	45
	DECLARAÇÃO SOBRE USO DE INTELIGÊNCIA ARTIFICIAL	45
	REFERÊNCIAS	46

1 INTRODUÇÃO

Plataformas de gestão educacional são utilizadas para organizar turmas, disponibilizar conteúdos, atribuir atividades, registrar entregas e acompanhar notas. Entre essas plataformas, o Google Classroom destaca-se por integrar diferentes serviços do ecossistema Google e por disponibilizar recursos para gerenciamento de atividades acadêmicas. Entretanto, sua utilização está concentrada principalmente na organização e no registro das informações, não oferecendo, de forma nativa, mecanismos personalizados de visualização gamificada do progresso acadêmico (DETERDING et al., 2011; FARDO, 2013).

A gamificação consiste na aplicação de elementos associados ao design de jogos em contextos que não são propriamente jogos (DETERDING et al., 2011; WERBACH; HUNTER, 2012). Em ambientes educacionais, elementos como pontos, níveis, barras de progresso, rankings e feedbacks visuais podem ser empregados para representar metas, desempenho e evolução de maneira mais dinâmica. A utilização desses elementos, contudo, deve estar associada a comportamentos e necessidades claramente definidos, não devendo ser limitada à inserção de componentes meramente visuais (WERBACH; HUNTER, 2012; TODA et al., 2019).

O problema técnico abordado neste trabalho consiste em integrar uma aplicação web ao Google Classroom, persistir os dados recuperados, aplicar regras de gamificação e apresentar os resultados em interfaces distintas para acompanhamento de atividades e desempenho.

Como solução, foi desenvolvida uma plataforma web denominada Gamify, implementada com Python e Django e integrada ao Google Classroom por meio de OAuth 2.0 e da Google Classroom API. A aplicação permite autenticação por conta Google, sincronização de informações acadêmicas, cálculo de pontos de experiência e níveis, geração de barras de progresso, rankings e painéis de desempenho, além da disponibilização de dados por meio de uma API REST.

O escopo deste relatório concentra-se no processo de desenvolvimento, na arquitetura, nas tecnologias utilizadas, na implantação e na verificação técnica das funcionalidades implementadas. Não foram realizadas avaliações com professores ou estudantes, nem coletados dados de participantes para mensurar motivação, experiência de uso ou mudanças efetivas no engajamento. Dessa forma, as análises apresentadas referem-se ao funcionamento técnico da solução, e não aos seus efeitos comportamentais ou pedagógicos.

1.1 OBJETIVOS

Esta seção apresenta os objetivos que orientaram o desenvolvimento deste trabalho, organizados em objetivo geral e objetivos específicos.

1.1.1 Objetivo Geral

Desenvolver uma plataforma web gamificada integrada ao Google Classroom para sincronizar, processar e apresentar informações sobre atividades.

1.1.2 Objetivos Específicos

- Realizar o levantamento de elementos de gamificação com potencial para promover o engajamento na entrega de atividades
- Desenvolver a plataforma gamificada incorporando os elementos identificados na etapa de levantamento.
- Estabelecer a integração entre a plataforma gamificada e o Google Classroom.
- Avaliar o funcionamento da plataforma por meio de testes automatizados.

1.2 PROBLEMÁTICA

O processo de ensino e aprendizagem frequentemente enfrenta desafios relacionados à redução do engajamento dos estudantes diante de atividades repetitivas e pouco estimulantes (FARDO, 2013). Embora plataformas virtuais de aprendizagem tenham ampliado o acesso aos conteúdos e facilitado a organização das atividades acadêmicas, a simples digitalização das práticas educacionais não garante maior participação dos alunos. Em muitos casos, o uso dessas plataformas limita-se ao gerenciamento de turmas, disponibilização de materiais e registro de notas, sem oferecer mecanismos que incentivem a motivação contínua durante o processo de aprendizagem (FARDO, 2013; LEITÃO et al., 2022).

Nesse contexto, o Google Classroom destaca-se como uma ferramenta consolidada para a gestão de atividades, cursos e avaliações (GOOGLE, 2026c). Entretanto, sua proposta está voltada principalmente para aspectos administrativos, não disponibilizando, de forma nativa, recursos baseados em gamificação, como sistemas de pontuação, níveis, rankings e indicadores visuais de progresso. Essa limitação dificulta a utilização da plataforma como um instrumento capaz de estimular o acompanhamento do desempenho acadêmico de maneira mais atrativa para estudantes e professores.

1.3 JUSTIFICATIVA

A utilização de estratégias de gamificação na educação tem sido apontada como uma alternativa para tornar o processo de aprendizagem mais envolvente por meio da incorporação de elementos característicos dos jogos em contextos não lúdicos (FARDO, 2013; TODA et al., 2019). Recursos como pontos, níveis, rankings, barras de progresso e feedback imediato podem incentivar a participação dos estudantes, favorecer o acompanhamento do próprio desempenho e

estimular uma competição saudável entre os participantes (ZICHERMANN; CUNNINGHAM, 2011; LEITÃO et al., 2022).

Considerando que o Google Classroom já é amplamente utilizado para o gerenciamento de atividades acadêmicas, o desenvolvimento de uma plataforma complementar que utilize os dados disponibilizados por essa ferramenta permite agregar funcionalidades voltadas ao acompanhamento gamificado sem alterar o ambiente originalmente utilizado por professores e estudantes. Dessa forma, a aplicação proposta aproveita as informações sincronizadas da plataforma Google Classroom para gerar indicadores visuais de desempenho, aproveitando a infraestrutura já existente e ampliando suas possibilidades de uso.

1.4 ORGANIZAÇÃO DO DOCUMENTO

Este documento está organizado em seis capítulos. Este primeiro capítulo apresentou a introdução, os objetivos, a problemática e a justificativa do trabalho. O segundo reúne o referencial teórico que fundamenta a especificação, a modelagem, a arquitetura, os testes, os dados e a publicação de sistemas. O terceiro descreve o processo de desenvolvimento do software, incluindo a definição dos elementos de gamificação e dos requisitos. O quarto apresenta a plataforma Gamify, sua arquitetura, suas tecnologias, seu modelo de dados e suas interfaces. O quinto reúne os procedimentos e os resultados dos testes automatizados. Por fim, o sexto capítulo discute os resultados técnicos, as limitações e as possibilidades de evolução do sistema.

2 REFERENCIAL TEÓRICO

Este capítulo apresenta os fundamentos conceituais relacionados aos principais conceitos utilizados no desenvolvimento do Gamify. A compreensão desses elementos fornece uma base conceitual para as atividades de especificação, modelagem, arquitetura, persistência, implantação e verificação técnica descritas nos capítulos posteriores.

2.1 ENGENHARIA DE REQUISITOS

A Engenharia de Requisitos compreende o conjunto de atividades relacionadas à identificação, análise, especificação, verificação e gerenciamento das necessidades e restrições de um sistema. Sua finalidade é estabelecer uma compreensão documentada do problema e dos resultados esperados, fornecendo uma base para a modelagem, o projeto, a implementação e os testes do software (ISO/IEC/IEEE, 2018; SOMMERVILLE, 2016).

2.1.1 Requisitos Funcionais

Requisitos funcionais descrevem os serviços, comportamentos e transformações que um sistema deve realizar para atender às necessidades de seus interessados. Eles especificam as respostas esperadas diante de entradas, eventos ou condições determinadas e delimitam as funções que devem estar disponíveis por meio do software. A engenharia de requisitos organiza a identificação, a análise, a documentação e o gerenciamento dessas necessidades durante o ciclo de vida do sistema (ISO/IEC/IEEE, 2018; SOMMERVILLE, 2016).

A definição precisa dos requisitos funcionais estabelece uma referência para a modelagem, a implementação e os testes. Requisitos claros e verificáveis reduzem ambiguidades entre os interessados e a equipe técnica, possibilitam a rastreabilidade entre necessidades e componentes e fornecem critérios objetivos para verificar se os serviços previstos foram implementados corretamente (PRESSMAN; MAXIM, 2020; IEEE COMPUTER SOCIETY, 2024).

2.1.2 Requisitos Não Funcionais

Requisitos não funcionais expressam propriedades de qualidade, restrições e condições sob as quais as funcionalidades devem ser oferecidas. Eles abrangem aspectos como desempenho, segurança, confiabilidade, manutenibilidade, compatibilidade, portabilidade e usabilidade. Diferentemente dos requisitos que descrevem serviços específicos, esses requisitos condicionam o comportamento global do produto e influenciam decisões de arquitetura e projeto (ISO/IEC/IEEE, 2018; ISO/IEC, 2023).

O modelo de qualidade da ISO/IEC 25010 organiza características que podem ser utilizadas na especificação e na avaliação de produtos de software. A formalização desses

atributos por meio de critérios mensuráveis favorece a definição de objetivos arquiteturais, estratégias de teste e condições de aceitação, evitando que qualidades relevantes sejam tratadas apenas de forma subjetiva (ISO/IEC, 2023; BASS; CLEMENTS; KAZMAN, 2021).

2.2 GAMIFICAÇÃO

Gamificação corresponde ao uso de elementos de design de jogos em contextos que não são jogos. Em sistemas computacionais, essa abordagem utiliza elementos característicos dos jogos para representar metas, progressão, desempenho e retorno das ações por meio de mecanismos como pontos, níveis, rankings, desafios e indicadores visuais. Sua finalidade consiste em estruturar a interação com base em regras e sinais de progresso, sem transformar necessariamente a atividade principal em um jogo (DETERDING et al., 2011; WERBACH; HUNTER, 2012).

A seleção dos elementos deve considerar os objetivos do sistema, os comportamentos que se pretende apoiar e a informação disponível para alimentar as regras. Taxonomias de gamificação educacional contribuem para distinguir elementos de desempenho, progressão, interação e apresentação, enquanto abordagens de projeto orientam a associação entre mecânicas, feedback e experiência de uso (TODA et al., 2019; ZICHERMANN; CUNNINGHAM, 2011).

2.3 MODELAGEM DE SISTEMAS

A modelagem de sistemas consiste na elaboração de representações abstratas utilizadas para descrever diferentes perspectivas de um software durante seu desenvolvimento. Esses modelos permitem representar estrutura, comportamento, interações e dados do sistema, facilitando a análise, a comunicação entre os envolvidos e a documentação das decisões de projeto. A UML constitui uma linguagem padronizada amplamente utilizada para especificar e documentar esses modelos (OBJECT MANAGEMENT GROUP, 2017; PRESSMAN; MAXIM, 2020).

2.4 DIAGRAMA DE CASOS DE USO

O diagrama de casos de uso representa as funcionalidades oferecidas pelo sistema sob a perspectiva dos atores que interagem com ele. Os atores correspondem a papéis desempenhados por pessoas, organizações ou sistemas externos, enquanto os casos de uso delimitam objetivos alcançados por meio da interação com o software. Essa representação concentra-se no comportamento externamente observável, sem detalhar os algoritmos ou a organização interna da solução (OBJECT MANAGEMENT GROUP, 2017; SOMMERVILLE, 2016).

Na Engenharia de Software, esse diagrama apoia a delimitação do escopo funcional, a identificação de interfaces externas e a comunicação dos serviços esperados. As relações entre atores e casos de uso também auxiliam na organização de cenários e na derivação de critérios

de verificação, desde que sejam complementadas por descrições textuais quando o fluxo exigir maior precisão (PRESSMAN; MAXIM, 2020; OBJECT MANAGEMENT GROUP, 2017).

2.5 DIAGRAMA DE CLASSES

O diagrama de classes representa a estrutura estática de um sistema orientado a objetos. Classes são descritas por responsabilidades, atributos e operações, e podem ser relacionadas por associações, agregações, composições, generalizações e dependências. A notação permite representar tipos, multiplicidades e vínculos estruturais que permanecem relevantes independentemente de uma execução específica do sistema (OBJECT MANAGEMENT GROUP, 2017; PRESSMAN; MAXIM, 2020).

Esse artefato é utilizado para organizar conceitos do domínio, analisar responsabilidades e orientar a estrutura do código e da persistência. Sua importância está na capacidade de tornar explícitas as dependências entre elementos, facilitar a análise de coesão e acoplamento e manter uma representação comum entre as etapas de análise, projeto e implementação (SOMMERVILLE, 2016; OBJECT MANAGEMENT GROUP, 2017).

2.6 MODELAGEM DE BANCO DE DADOS

A modelagem de banco de dados consiste na representação estruturada das entidades de um domínio, seus atributos, relacionamentos e restrições antes da implementação física da base de dados. Essa atividade organiza as informações de forma consistente, reduz redundâncias e contribui para preservar a integridade dos dados ao longo do desenvolvimento do sistema. No modelo relacional, essas informações são organizadas em tabelas relacionadas por chaves primárias e estrangeiras, permitindo consultas e operações de persistência de maneira eficiente (ELMASRI; NAVATHE, 2016; POSTGRESQL GLOBAL DEVELOPMENT GROUP, 2026b).

2.7 ARQUITETURA CLIENTE-SERVIDOR

A arquitetura cliente-servidor organiza uma aplicação em componentes responsáveis por solicitar e fornecer serviços. Enquanto o cliente realiza as requisições e apresenta a interface ao usuário, o servidor concentra o processamento das regras de negócio, o acesso aos dados e a geração das respostas. Essa separação favorece a manutenção do sistema, a reutilização de componentes e o atendimento simultâneo de múltiplos clientes (BASS; CLEMENTS; KAZMAN, 2021; SOMMERVILLE, 2016).

2.8 TESTES DE SOFTWARE

Testes de software consistem na execução ou avaliação sistemática de componentes com a finalidade de revelar defeitos e verificar o atendimento aos requisitos especificados. A atividade

envolve o planejamento de condições, entradas, resultados esperados e critérios de conclusão, podendo abranger diferentes níveis, como testes unitários, de integração, de sistema e de aceitação (IEEE COMPUTER SOCIETY, 2024; PRESSMAN; MAXIM, 2020).

Na Engenharia de Software, os testes fornecem evidências sobre os comportamentos examinados, mas não demonstram a ausência completa de defeitos. Sua eficácia depende da seleção de casos representativos, da rastreabilidade com os requisitos, do controle do ambiente e do registro dos resultados. A automatização contribui para repetir verificações, detectar regressões e apoiar alterações incrementais no software (SOMMERVILLE, 2016; IEEE COMPUTER SOCIETY, 2024).

2.9 DOMÍNIO E DNS

Um nome de domínio corresponde à identificação textual utilizada para localizar serviços na Internet. O Sistema de Nomes de Domínio (DNS) é responsável por traduzir esse nome em um endereço IP, permitindo que usuários e aplicações acessem servidores sem a necessidade de utilizar endereços numéricos. Essa estrutura distribuída constitui um dos principais mecanismos de funcionamento da Internet, viabilizando a localização dos serviços publicados (INTERNET CORPORATION FOR ASSIGNED NAMES AND NUMBERS, 2022; MOCKAPETRIS, 1987a,b).

3 PROCESSO DE DESENVOLVIMENTO DO SOFTWARE

Este capítulo descreve o processo de desenvolvimento do Gamify, abrangendo a definição do escopo, a seleção dos elementos de gamificação, o levantamento dos requisitos, a modelagem, a implementação, a preparação do ambiente, a implantação e a estratégia de testes. O desenvolvimento ocorreu de maneira incremental, permitindo que as decisões de modelagem e arquitetura fossem ajustadas conforme as necessidades de integração com o Google Classroom e as funcionalidades previstas para a plataforma.

3.1 DEFINIÇÃO DOS ELEMENTOS DE GAMIFICAÇÃO

A definição dos elementos de gamificação partiu da identificação das informações disponíveis por meio da Google Classroom API e dos comportamentos que a aplicação pretende apoiar. Foram considerados, principalmente, o acompanhamento do progresso acadêmico, a visualização do desempenho e a comparação de resultados dentro de cada disciplina.

Os elementos selecionados foram pontos de experiência (XP), níveis, barras de progresso, rankings e feedback visual por meio de painéis e gráficos. A escolha desses componentes fundamentou-se na necessidade de representar o desempenho acadêmico utilizando dados efetivamente disponibilizados pela Google Classroom API e em sua compatibilidade com o objetivo da plataforma de acompanhar a evolução dos usuários. Além disso, esses elementos correspondem às categorias relacionadas ao desempenho, à progressão e à apresentação de resultados descritas na taxonomia de gamificação educacional proposta por Toda et al. (2019), sendo amplamente empregados em sistemas gamificados para fornecer retorno contínuo sobre o progresso do usuário (WERBACH; HUNTER, 2012; LEITÃO et al., 2022).

Na aplicação desenvolvida, os pontos de experiência foram associados ao desempenho registrado nas atividades, permitindo representar numericamente a evolução do usuário. Os níveis organizam a progressão acumulada em etapas. As barras de progresso tornam visível a situação do usuário em relação às atividades e aos pontos necessários para alcançar um novo nível. Os rankings apresentam uma comparação de pontuação por disciplina, enquanto os painéis e gráficos fornecem feedback visual sobre os dados processados (ZICHERMANN; CUNNINGHAM, 2011).

A seleção também considerou a viabilidade técnica de implementação. Como o Google Classroom disponibiliza informações sobre cursos, atividades, submissões e notas, esses dados podem ser sincronizados, armazenados e transformados em indicadores gamificados. Elementos dependentes de informações temporais não processadas pelo sistema não foram incluídos nesta versão. O termo “missões” é utilizado apenas como denominação visual das atividades e não representa um elemento de gamificação independente. A Tabela 1 sintetiza a relação entre as necessidades identificadas, os elementos selecionados, sua implementação e o resultado técnico

esperado.

Tabela 1 – Matriz dos elementos de gamificação

Necessidade	Elemento	Implementação no sistema	Resultado técnico esperado
Acompanhar a evolução	Barra de progresso	Percentual de progresso e XP necessário para o próximo nível	Tornar o progresso visível
Representar o desempenho acumulado	Pontos de experiência	Conversão das notas em XP	Quantificar a progressão
Organizar a progressão	Níveis	Cálculo do nível com base no XP acumulado	Estruturar etapas de evolução
Comparar pontuações	Ranking	Ordenação dos usuários por curso ou disciplina	Apresentar desempenho comparativo
Fornecer retorno visual	Painéis e gráficos	Indicadores de notas, atividades e progresso	Facilitar a interpretação dos dados

Fonte: Elaborada pelo autor (2026).

Conforme apresentado na Tabela 1, cada elemento possui correspondência com uma necessidade de acompanhamento e com uma operação implementada no sistema. A matriz evidencia que pontos, níveis, barras de progresso, rankings e painéis não foram tratados apenas como componentes visuais, mas como formas de representar progressão, desempenho comparativo e retorno sobre os dados acadêmicos processados.

3.2 LEVANTAMENTO E ESPECIFICAÇÃO DOS REQUISITOS

Os requisitos foram definidos a partir do problema técnico, do escopo da integração com o Google Classroom, dos elementos de gamificação selecionados e dos fluxos esperados para a plataforma. Inicialmente, foram identificadas as informações disponibilizadas pela API externa. Em seguida, foram definidos os comportamentos que a aplicação deveria executar sobre esses dados, incluindo sincronização, persistência, cálculo de indicadores e apresentação das informações. O processo também considerou requisitos não funcionais de segurança, integridade, interoperabilidade, manutenibilidade e configuração do ambiente.

Os requisitos foram organizados em quatro grupos. Os requisitos funcionais descrevem comportamentos disponibilizados pelo software; os requisitos de gamificação especificam a transformação dos dados acadêmicos em indicadores; os requisitos não funcionais representam propriedades de qualidade esperadas; e as restrições técnicas registram tecnologias, padrões e condições de implementação adotadas no projeto. Por complementarem as funcionalidades, os requisitos de gamificação e os requisitos não funcionais são apresentados na mesma subseção, em tabelas distintas.

3.2.1 Requisitos Funcionais

Os comportamentos necessários à autenticação, à sincronização, à persistência, à apresentação dos indicadores e à exposição dos dados foram registrados de maneira identificável. A

Tabela 2 apresenta os requisitos funcionais definidos para delimitar os serviços disponibilizados pela plataforma.

Tabela 2 – Requisitos funcionais da plataforma

Código	Requisito
RF01	Permitir autenticação por conta Google e obter autorização OAuth 2.0 para acesso aos dados do Google Classroom.
RF02	Criar ou atualizar o estudante local e o rastreador de acesso a partir do perfil Google autenticado.
RF03	Consultar e sincronizar cursos vinculados ao usuário autenticado.
RF04	Consultar e importar atividades associadas aos cursos sincronizados.
RF05	Consultar submissões e notas associadas às atividades.
RF06	Persistir estudantes, cursos, atividades, notas e seus relacionamentos.
RF07	Registrar a participação do usuário em cursos e seu papel como estudante ou professor.
RF08	Atualizar notas persistidas mediante nova consulta à Google Classroom API.
RF09	Exibir dashboard com cursos, indicadores de progresso, nível e XP acumulado.
RF10	Disponibilizar ranking de usuários por disciplina.
RF11	Disponibilizar painéis e gráficos para acompanhamento dos dados acadêmicos.
RF12	Permitir a busca de novas disciplinas e atividades a partir de ação do usuário.
RF13	Disponibilizar links para abertura de cursos e atividades no Google Classroom quando fornecidos pela API.
RF14	Expor dados de participações, cursos, notas e desempenho por endpoints REST.

Fonte: Elaborada pelo autor (2026).

A Tabela 2 evidencia que o fluxo funcional abrange desde a autorização de acesso aos dados externos até sua transformação e disponibilização nas interfaces web e nos endpoints REST. A identificação individual dos requisitos também permite relacionar autenticação, sincronização, armazenamento, gamificação e consulta às etapas posteriores de implementação e verificação técnica.

3.2.2 Requisitos Não Funcionais

Os requisitos de gamificação descrevem a transformação dos dados acadêmicos em indicadores, enquanto os requisitos não funcionais registram propriedades de qualidade e condições técnicas da solução. A Tabela 3 apresenta os requisitos de gamificação e a Tabela 4, os requisitos não funcionais, relacionando cada requisito à evidência prevista para sua verificação.

Tabela 3 – Requisitos de gamificação e formas de verificação

Código	Categoria	Requisito	Evidência ou verificação
RG01	Gamificação	O sistema deve converter o desempenho acadêmico em pontos de experiência.	Testes das regras de cálculo de XP.
RG02	Gamificação	O sistema deve calcular o nível do usuário com base no XP acumulado.	Testes das regras de cálculo de nível.
RG03	Gamificação	O sistema deve apresentar visualmente a progressão até o próximo nível.	Testes de progresso e inspeção da interface.
RG04	Gamificação	O sistema deve calcular o progresso do usuário por disciplina.	Testes das respostas de progresso por disciplina.
RG05	Gamificação	O sistema deve ordenar os usuários em ranking por disciplina.	Testes de cálculo e resposta do ranking.
RG06	Gamificação	O sistema deve apresentar indicadores gráficos de desempenho.	Inspeção dos painéis e gráficos.
RG07	Gamificação	O sistema deve informar ao usuário os critérios utilizados no cálculo de XP e ranking.	Inspeção da interface e das regras apresentadas.

Fonte: Elaborada pelo autor (2026).

Como demonstra a Tabela 3, os requisitos de gamificação estabelecem a sequência entre conversão das notas em XP, definição de níveis, cálculo de progresso, ordenação do ranking e apresentação dos indicadores.

Tabela 4 – Requisitos não funcionais e formas de verificação

Código	Categoria	Requisito	Evidência ou verificação
RNF01	Segurança	O acesso aos dados do Google Classroom deve ocorrer mediante autenticação e autorização OAuth 2.0.	Testes de rotas protegidas e fluxo de autenticação.
RNF02	Segurança	A aplicação deve solicitar apenas os escopos necessários à execução de suas funcionalidades.	Revisão da configuração OAuth.
RNF03	Segurança	Credenciais, chaves e segredos não devem ser armazenados diretamente no código-fonte.	Inspeção da configuração e das variáveis de ambiente.
RNF04	Segurança	O ambiente público deve utilizar HTTPS, DEBUG=False e restrição de hosts permitidos.	Inspeção do ambiente implantado.
RNF05	Confiabilidade	O sistema deve preservar a integridade dos relacionamentos entre cursos, estudantes, atividades e notas.	Testes de modelos e restrições do banco.
RNF06	Desempenho	Os dados sincronizados devem ser persistidos localmente para evitar chamadas externas em todas as visualizações.	Inspeção da arquitetura e testes de persistência.
RNF07	Manutenibilidade	O código deve separar apresentação, regras de negócio, persistência e integração externa.	Inspeção da estrutura do projeto.
RNF08	Interoperabilidade	Os dados da aplicação devem ser disponibilizados em formato JSON por endpoints REST.	Testes dos endpoints.
RNF09	Usabilidade	A interface deve apresentar navegação consistente e adaptação a diferentes dimensões de tela.	Inspeção da interface.
RNF10	Portabilidade	As dependências e configurações necessárias à execução devem ser declaradas e reproduzíveis.	requirements.txt, ambiente virtual e documentação.

Fonte: Elaborada pelo autor (2026).

Entre os requisitos não funcionais apresentados na Tabela 4, segurança concentra parte relevante das condições definidas, incluindo autorização OAuth, proteção de credenciais, HTTPS e configuração do ambiente público. Os demais requisitos abrangem integridade, persistência

local, separação de responsabilidades, interoperabilidade, adaptação da interface e reprodução do ambiente.

3.2.3 Restrições Técnicas

As restrições técnicas delimitam as tecnologias e os padrões efetivamente adotados na construção e na implantação da plataforma. A Tabela 5 apresenta as escolhas que condicionam o backend, a persistência, a interface, a API REST, a integração externa e o ambiente de execução.

Tabela 5 – Restrições técnicas da plataforma

Código	Restrição técnica
RT01	O backend implementado com Python e Django.
RT02	Persistência com PostgreSQL e ORM do Django.
RT03	Interface com templates Django.
RT04	A camada REST desenvolvida com Django REST Framework.
RT05	A integração com Google Classroom API e OAuth 2.0.
RT06	A aplicação implantada em ambiente Linux com servidor compatível com WSGI.

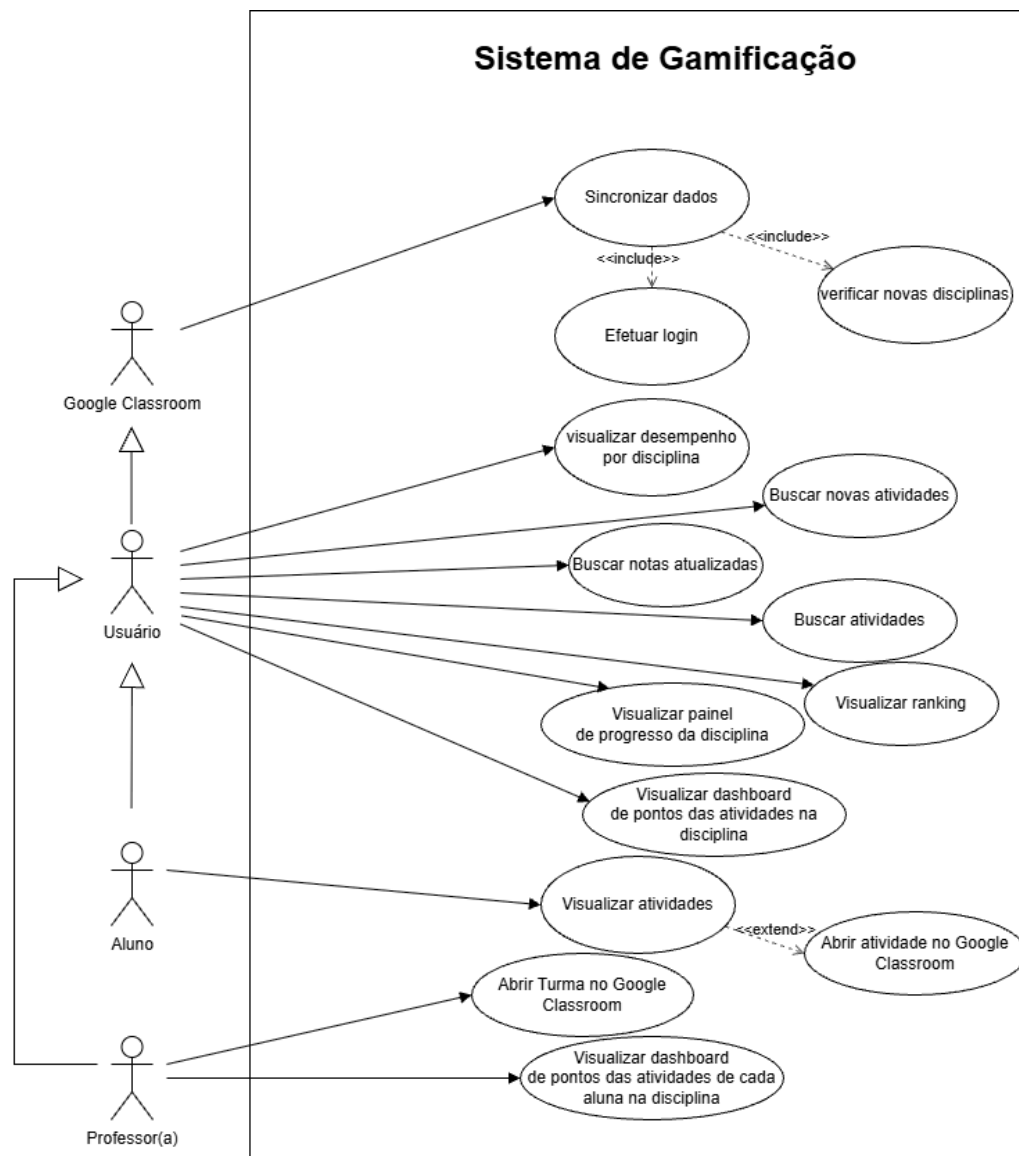
Fonte: Elaborada pelo autor (2026).

A Tabela 5 mostra que as tecnologias foram definidas de forma integrada: Django estrutura o backend e a interface renderizada, PostgreSQL e o ORM sustentam a persistência, Django REST Framework fornece a camada de serviços e OAuth 2.0 viabiliza a comunicação autorizada com o Google Classroom. A implantação em Linux com suporte a WSGI completa o conjunto de condições técnicas da solução.

3.3 MODELAGEM DO SISTEMA

A modelagem foi utilizada para representar atores, funcionalidades, entidades, relacionamentos e componentes antes e durante a implementação. A Figura 1 apresenta o diagrama de casos de uso, empregado para registrar as interações externas e as funcionalidades acessadas pelos atores da plataforma.

Figura 1 – Diagrama de casos de uso do sistema

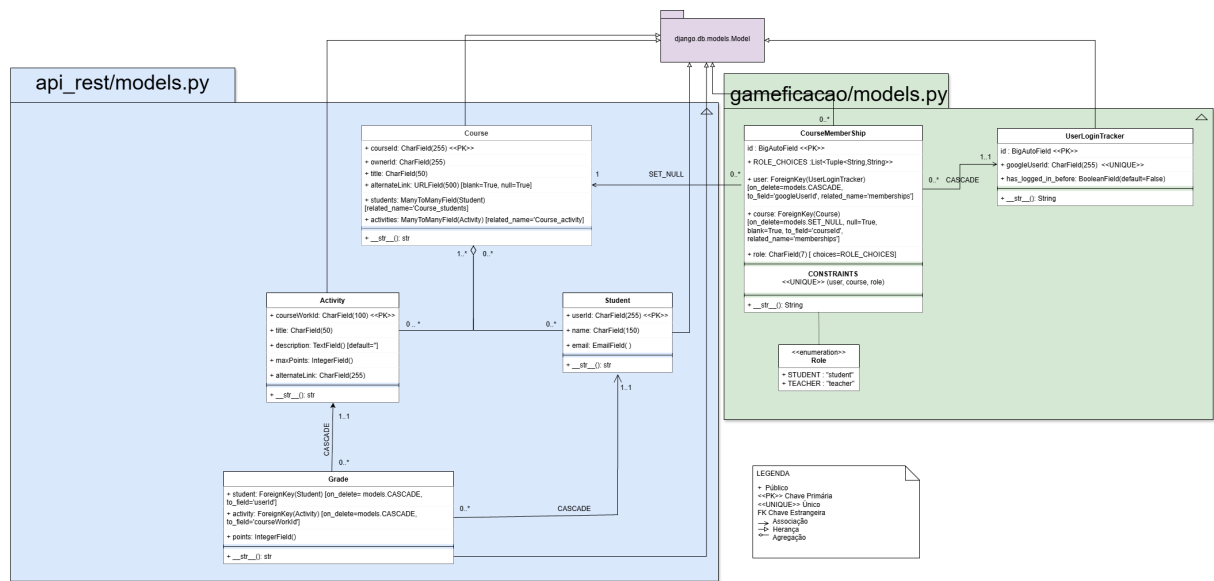


Fonte: Elaborada pelo autor (2026).

A Figura 1 organiza as funcionalidades em torno da sincronização de informações acadêmicas e de sua apresentação em interfaces de acompanhamento. Alunos e professores utilizam a autenticação Google e acessam os dados de acordo com os vínculos de curso, enquanto o Google Classroom atua como sistema externo provedor das informações.

Para complementar a perspectiva comportamental, a Figura 2 apresenta a estrutura estática do domínio, identificando as classes e os relacionamentos responsáveis pela representação dos dados acadêmicos.

Figura 2 – Diagrama de classes do sistema



Fonte: Elaborada pelo autor (2026).

A Figura 2 representa estudantes, cursos, atividades, notas, usuários rastreados e vínculos de participação. A classe `Grade` associa estudante e atividade; cursos agregam estudantes e atividades; e `CourseMembership` relaciona o usuário autenticado ao curso e ao papel desempenhado.

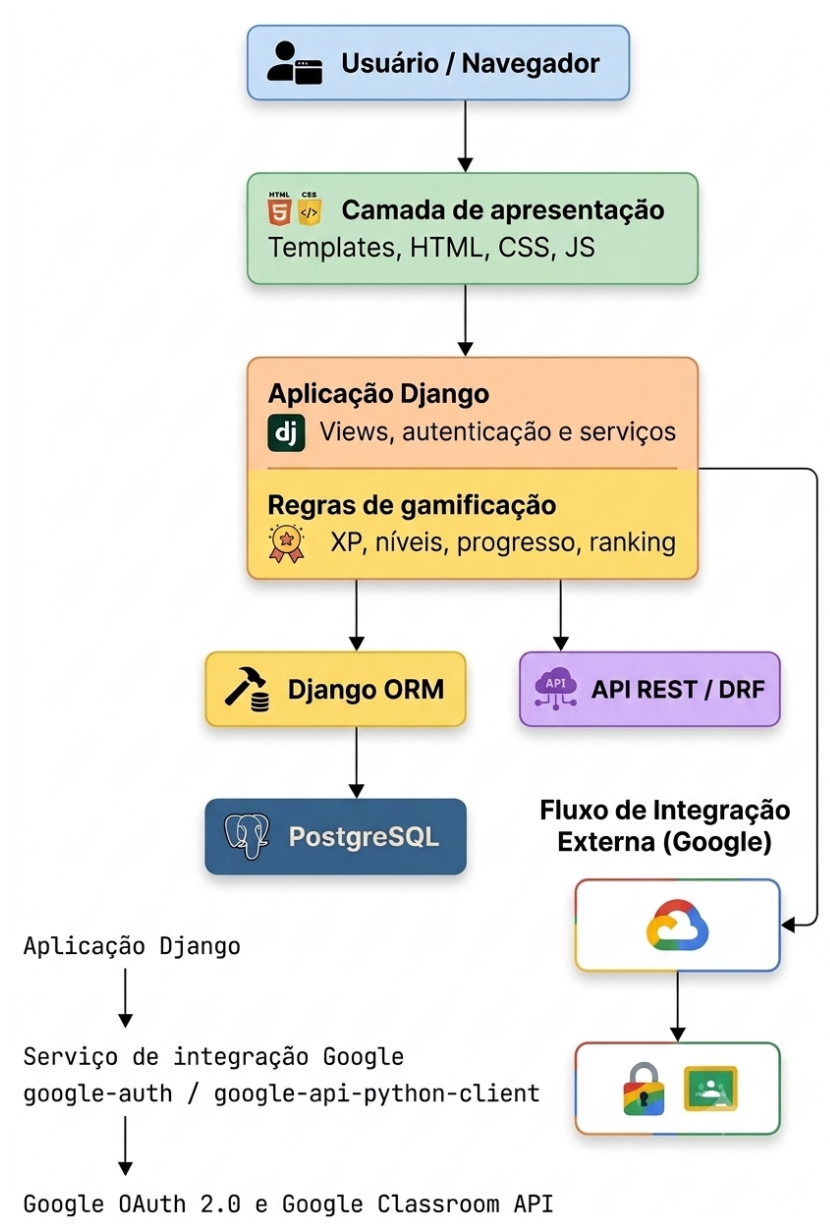
Os dados de gamificação, como XP, nível, progresso e posição no ranking, não aparecem como classes ou atributos no diagrama porque não são armazenados no banco de dados. Esses valores são calculados no momento da consulta pelos utilitários de gamificação, a partir do atributo `points` da classe `Grade` e do atributo `maxPoints` da classe `Activity`, conforme descrito na Seção 3.4.2.

3.3.1 Arquitetura Definida

A solução adota uma arquitetura cliente-servidor organizada em camadas. O cliente corresponde ao navegador utilizado para acessar as páginas renderizadas pela aplicação. A camada de apresentação é formada por templates Django, HTML, CSS, JavaScript e Bootstrap. A camada de aplicação, implementada com Django, concentra autenticação, controle de sessão, processamento das requisições e regras de negócio. A lógica de gamificação calcula pontos de experiência, níveis, progresso e rankings a partir dos dados acadêmicos persistidos.

A integração externa é realizada por componentes responsáveis pela autenticação OAuth 2.0 e pelo consumo da Google Classroom API. Os dados recuperados são convertidos em entidades locais e armazenados por meio do ORM do Django em uma base PostgreSQL. Paralelamente, a aplicação disponibiliza endpoints desenvolvidos com Django REST Framework, permitindo acesso estruturado aos dados em formato JSON. A Figura 3 apresenta o fluxo entre essas camadas e os serviços externos envolvidos.

Figura 3 – Arquitetura em camadas da solução



Fonte: Imagem gerada com Gemini (2026).

Conforme representado na Figura 3, a organização separa interface, regras de negócio, integração externa, persistência e exposição dos dados. Essa divisão reduz o acoplamento entre os componentes e facilita a manutenção e a evolução da aplicação.

3.4 IMPLEMENTAÇÃO DA SOLUÇÃO

A implementação foi dividida entre a aplicação `gameficacao`, responsável pelas páginas, autenticação, sincronização e regras de gamificação, e a aplicação `api_rest`, responsável pela exposição estruturada dos dados. As views coordenam as requisições e preparam os dados para os templates; os modelos representam as entidades persistidas; e os utilitários concentram os cálculos de XP, níveis e ranking e a comunicação com o Google Classroom.

3.4.1 Implementação da Integração com o Google Classroom

O arquivo `resquest_classroom_api.py` concentra as funções utilizadas na obtenção dos dados externos. A função `courses()` lista os cursos vinculados ao usuário autenticado e retorna seus identificadores, nomes, links e proprietários. A função `coursesWork()` percorre os cursos e consulta as atividades associadas, estruturando título, pontuação máxima, descrição e link de acesso.

A função `studentSubmissions()` consulta as submissões de cada atividade e relaciona as pontuações retornadas aos estudantes do curso. A função `getWorks()` compara os identificadores recebidos da API com as atividades já persistidas para importar somente novos registros. Por fim, `getGrade()` localiza a submissão de um estudante em uma atividade específica e retorna a nota atribuída quando o valor está disponível e é numérico.

O fluxo de integração inicia pela obtenção dos cursos, prossegue pela consulta de atividades, submissões, estudantes e notas e termina na criação ou atualização dos modelos locais. A separação dessas funções em um componente específico evita que os detalhes de comunicação com a API sejam incorporados diretamente aos templates ou à camada de persistência.

3.4.2 Implementação das Regras de Gamificação

As notas persistidas são normalizadas e convertidas em pontos de experiência. A regra documentada utiliza XP base, multiplicadores associados à nota, bonificações para desempenhos elevados e redução para notas abaixo do limite definido. O total acumulado determina o nível do usuário e a quantidade de XP necessária para a etapa seguinte. Os rankings utilizam os pontos acumulados e as atividades avaliadas para ordenar os usuários no contexto de cada disciplina.

3.4.3 Implementação das Interfaces e da API REST

As páginas são renderizadas por templates Django e apresentam dashboard, cursos, progresso, ranking, análise gráfica e busca de novas atividades. A camada REST disponibiliza consultas de vínculos de curso, notas e percentuais por atividade em formato JSON, permitindo o consumo estruturado dos dados sem depender da renderização HTML.

3.5 PREPARAÇÃO DO AMBIENTE E GESTÃO DAS DEPENDÊNCIAS

O desenvolvimento utilizou Python 3.12 em ambiente virtual. As dependências foram declaradas em `requirements.txt`, e as configurações variáveis, como credenciais Google e acesso ao banco, foram carregadas por variáveis de ambiente com *python-dotenv*. Essa organização separa o código-fonte das configurações específicas de cada máquina e permite reproduzir o ambiente de execução.

3.6 IMPLANTAÇÃO

A aplicação foi disponibilizada em uma VPS da Hostinger com Ubuntu Server 24.04 LTS e OpenLiteSpeed. O processo compreendeu acesso remoto por SSH, obtenção do código-fonte, criação do ambiente virtual, instalação das dependências, execução das migrações, coleta dos arquivos estáticos e configuração do ponto de entrada WSGI. O domínio `eduquestify.com.br` é utilizado somente como endereço de hospedagem da plataforma Gamify. A infraestrutura e as configurações de publicação são detalhadas na Seção 4.7.

3.7 ESTRATÉGIA DE TESTES

A estratégia de verificação técnica utilizou testes automatizados do Django distribuídos entre as aplicações `gameficacao` e `api_rest`. Os testes foram planejados para validar modelos, views, URLs, sessões e controle de acesso, regras de gamificação, funções de integração, serializers e endpoints REST. Os cenários verificam persistência e relacionamentos, renderização e proteção das páginas, resolução das rotas, respostas JSON, sincronização, atualização de notas e tratamento de consultas inválidas.

O framework de testes do Django foi empregado por meio de `TestCase`, `SimpleTestCase`, cliente de testes e `RequestFactory`. Nos cenários que utilizam persistência, o executor cria uma base específica e isolada da aplicação, permitindo preparar dados conhecidos, repetir as execuções e comparar automaticamente os resultados com os valores esperados. Essa infraestrutura evita alterações nos dados dos ambientes de desenvolvimento e implantação e reduz interferências entre os testes.

As regras de negócio verificadas incluem cálculo de XP, níveis, progresso, estatísticas e posição no ranking. Na camada REST, foram examinados modelos acadêmicos, serialização das participações, rotas, retornos de cursos e notas, percentuais por atividade e respostas para usuários ou cursos inexistentes. Essa cobertura acompanha a separação arquitetural entre persistência, aplicação, integração e exposição dos dados.

A biblioteca `unittest.mock`, por meio de `patch` e `MagicMock`, substituiu credenciais, clientes e chamadas à Google Classroom API por objetos e respostas simuladas. Com isso, os testes de sincronização e atualização puderam ser executados sem conexão com a Internet, credenciais OAuth válidas ou disponibilidade do serviço externo, mantendo isolamento e repetibilidade. A estratégia fornece evidências sobre requisitos relacionados à autenticação, persistência, integração, gamificação e interoperabilidade, favorecendo a identificação de regressões e aumentando a confiabilidade técnica da solução antes da implantação, sem substituir avaliações de desempenho, segurança dinâmica, acessibilidade ou uso em contexto real.

4 PLATAFORMA GAMIFY

Este capítulo descreve a estrutura técnica da plataforma Gamify. São apresentadas a stack tecnológica, as bibliotecas de integração, a persistência, os serviços externos, as ferramentas, a gestão do ambiente, a infraestrutura de implantação, a arquitetura, o modelo de dados e as interfaces implementadas.

4.1 STACK TECNOLÓGICA

A stack tecnológica reúne a linguagem de programação, os frameworks e os recursos de interface utilizados na construção da plataforma. As subseções a seguir descrevem a função de cada uma dessas tecnologias no projeto.

4.1.1 Python

Python 3.12 foi utilizado como linguagem principal do backend. A linguagem implementa as regras de negócio, a comunicação com os serviços Google, o processamento dos dados acadêmicos e os componentes executados pelo framework Django (PYTHON SOFTWARE FOUNDATION, 2026).

4.1.2 Django

Django foi utilizado como framework web principal. O framework organiza o sistema segundo o padrão Model-View-Template, gerencia autenticação, rotas, sessões, modelos de dados, migrações e renderização das páginas (DJANGO SOFTWARE FOUNDATION, 2026).

4.1.3 Django REST Framework

Django REST Framework foi empregado na implementação da API REST. A camada disponibiliza dados de cursos, notas, participações e rankings em formato JSON por meio dos endpoints da aplicação (ENCODE OSS LTD., 2026).

4.1.4 HTML, CSS, JavaScript, Bootstrap e Templates Django

HTML e templates Django estruturam as páginas renderizadas no servidor; CSS e Bootstrap definem o layout e os componentes responsivos; e JavaScript complementa as interações e visualizações dinâmicas da interface (MDN WEB DOCS, 2026b,a,c; BOOTSTRAP TEAM, 2026).

4.2 BIBLIOTECAS E COMPONENTES DE INTEGRAÇÃO

O sistema utiliza bibliotecas complementares para autenticação, autorização, comunicação com serviços externos e configuração do ambiente. O *django-allauth* integra o mecanismo de autenticação do Django ao provedor Google. As bibliotecas *google-auth* e *google-api-python-client* são utilizadas, respectivamente, na manipulação das credenciais OAuth e no consumo da Google Classroom API. O *python-dotenv* permite carregar configurações e credenciais a partir de variáveis de ambiente, reduzindo a exposição de dados sensíveis no código-fonte (DJANGO-ALLAUTH CONTRIBUTORS, 2026; GOOGLE, 2026b,a).

4.3 PERSISTÊNCIA DE DADOS

PostgreSQL foi utilizado como sistema gerenciador de banco de dados relacional. A base armazena usuários vinculados, cursos, estudantes, atividades, notas e relações de participação sincronizadas a partir do Google Classroom. O acesso aos dados é realizado pelo ORM do Django, que representa as entidades como modelos, define relacionamentos e executa operações de persistência sem acoplamento direto a comandos SQL. A persistência local também possibilita calcular pontos de experiência, níveis, progresso e rankings sem realizar nova chamada à API externa em cada visualização (POSTGRES GLOBAL DEVELOPMENT GROUP, 2026a; DJANGO SOFTWARE FOUNDATION, 2026).

4.4 SERVIÇOS EXTERNOS E APIS

A integração com o ecossistema Google utiliza a Google Classroom API, o protocolo OAuth 2.0 e recursos de configuração disponibilizados pelo Google Cloud Console. A Google Classroom API fornece cursos, atividades, submissões, estudantes e notas. O OAuth 2.0 controla a autorização de acesso, enquanto o Google Cloud Console é utilizado para habilitar a API, cadastrar o cliente OAuth e configurar as URIs de redirecionamento. O OAuth Playground foi empregado durante o desenvolvimento para testar escopos e respostas da API, não integrando a execução final da aplicação (GOOGLE, 2026c,e; GOOGLE CLOUD, 2026; GOOGLE, 2026d).

4.5 FERRAMENTAS DE DESENVOLVIMENTO E MODELAGEM

O Visual Studio Code foi utilizado para edição e navegação no projeto. Git e GitHub apoiaram o controle de versão e a hospedagem do repositório, enquanto o Draw.io foi empregado na produção dos diagramas de casos de uso, classes, dados e arquitetura (MICROSOFT, 2026; GIT, 2026; GITHUB, 2026).

4.6 GESTÃO DO AMBIENTE E DEPENDÊNCIAS

O ambiente de desenvolvimento foi isolado por meio de um ambiente virtual Python. As dependências necessárias à execução foram registradas em `requirements.txt`. Configurações variáveis, como credenciais, endereço do banco e segredos da aplicação, foram carregadas a partir de variáveis de ambiente. Para consolidar as tecnologias descritas e suas responsabilidades, a Tabela 6 apresenta a stack organizada por categoria e função no projeto.

Tabela 6 – Resumo das tecnologias utilizadas

Categoria	Tecnologias	Função no projeto
Backend	Python 3.12 e Django	Regras de negócio, autenticação, rotas, modelos e renderização.
API	Django REST Framework	Disponibilização de dados em endpoints REST.
Frontend	HTML, CSS, JavaScript, Bootstrap e templates Django	Construção das interfaces e dos componentes visuais.
Autenticação e integração	<i>django-allauth</i> , OAuth 2.0, <i>google-auth</i> e <i>google-api-python-client</i>	Login Google, autorização e consumo da Classroom API.
Persistência	PostgreSQL e ORM do Django	Armazenamento e consulta dos dados sincronizados.
Serviços externos	Google Classroom API e Google Cloud Console	Fornecimento dos dados acadêmicos e configuração da integração.
Desenvolvimento	VS Code, Git, GitHub e Draw.io	Codificação, versionamento, documentação e modelagem.
Configuração	<i>venv</i> , <i>requirements.txt</i> e <i>python-dotenv</i>	Isolamento, reprodução do ambiente e gestão de configurações.
Implantação	Ubuntu Server, VPS e OpenLiteSpeed	Execução e disponibilização da aplicação web.

Fonte: Elaborada pelo autor (2026).

A Tabela 6 demonstra que a solução combina componentes de backend, frontend, integração, persistência e implantação. Essa distribuição evidencia a função de cada tecnologia e mantém separadas as responsabilidades de processamento, apresentação, acesso aos serviços Google, armazenamento, configuração e disponibilização da aplicação.

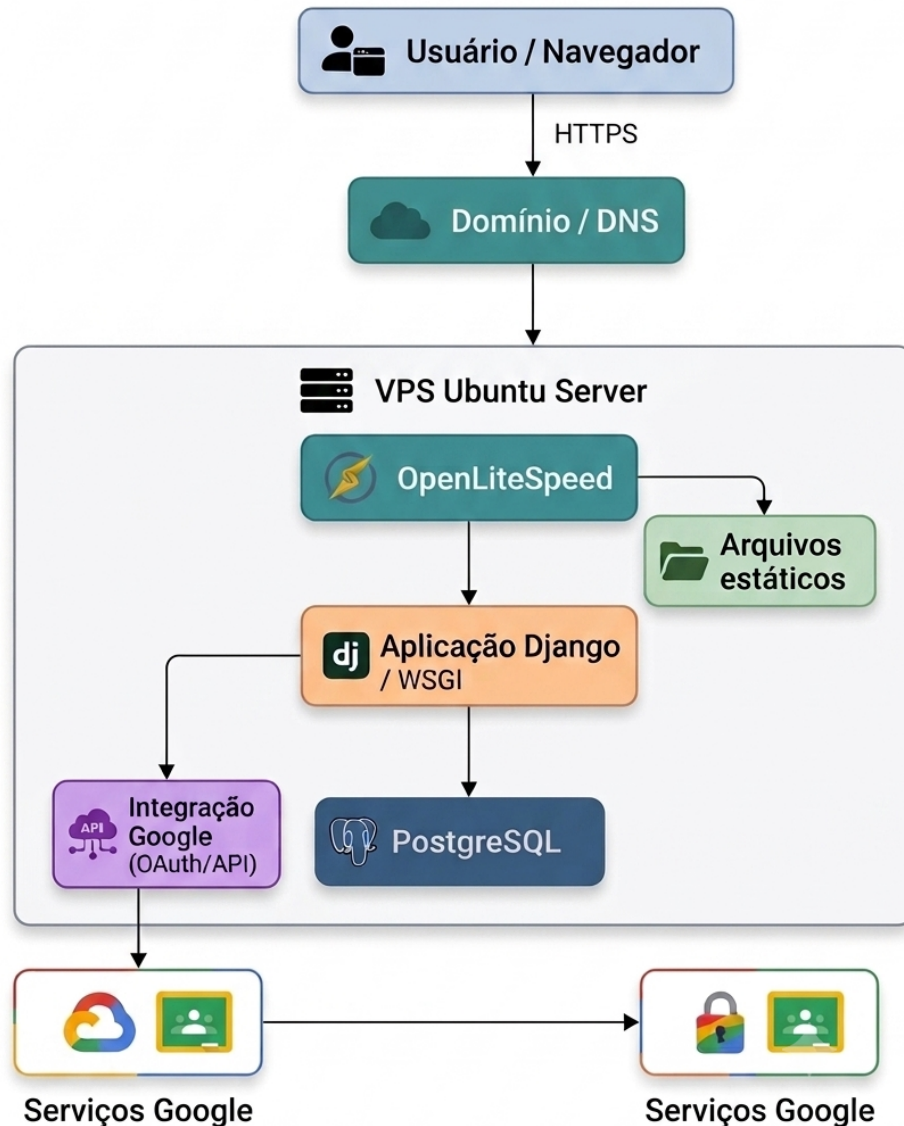
4.7 INFRAESTRUTURA DE IMPLANTAÇÃO

Esta seção descreve o ambiente em que a plataforma foi publicada, abrangendo o servidor utilizado, a preparação da VPS, a configuração do servidor web e do Django, o banco de dados e o domínio de acesso.

4.7.1 Infraestrutura

A aplicação foi hospedada em um Servidor Privado Virtual (VPS) da Hostinger com Ubuntu Server 24.04 LTS. O OpenLiteSpeed recebe as requisições destinadas ao endereço de hospedagem, encaminha o processamento dinâmico ao ponto de entrada WSGI do Django e entrega diretamente os arquivos estáticos. O PostgreSQL mantém os dados persistidos, e a aplicação comunica-se com os serviços Google por OAuth 2.0 e pela Google Classroom API. A Figura 4 apresenta a disposição desses componentes e seus fluxos de comunicação.

Figura 4 – Topologia de implantação da aplicação



Fonte: Imagem gerada com Gemini (2026).

Conforme ilustrado na Figura 4, as requisições externas chegam ao servidor web, que diferencia a entrega de arquivos estáticos do processamento encaminhado à aplicação Django. A aplicação, por sua vez, acessa o PostgreSQL para persistência e os serviços Google para autenticação e sincronização dos dados acadêmicos.

4.7.2 Preparação da VPS

O servidor foi administrado por SSH e recebeu uma chave própria para acesso ao repositório privado. Após a obtenção do código-fonte, foi criado um ambiente virtual com Python 3.12 e foram instaladas as dependências declaradas no projeto. O Código 4.1 apresenta a sequência geral de comandos empregada na preparação da VPS.

Código 4.1 – Sequência geral de preparação da VPS

```

1 ssh root@eduquestify.com.br
2 cd /usr/local/lsws/Example/html
3 git clone git@github.com:DiegoSousaOliveira/gamificationProject.git
4 cd gamificationProject
5 python3 -m venv .venv
6 source .venv/bin/activate
7 pip install -r requirements.txt
8 python manage.py migrate
9 python manage.py collectstatic

```

Fonte: Elaborada pelo autor (2026).

O Código 4.1 registra as etapas de acesso, obtenção do repositório, criação e ativação do ambiente virtual, instalação das dependências, aplicação das migrações e coleta dos arquivos estáticos. A sequência prepara os componentes necessários para que o servidor web possa encaminhar as requisições à aplicação implantada.

4.7.3 Configuração do Servidor Web

O OpenLiteSpeed foi configurado com um Virtual Host associado à aplicação. O servidor encaminha as requisições dinâmicas ao arquivo `wsgi.py` e mapeia as rotas de arquivos estáticos e de mídia para os diretórios definidos pelo Django.

4.7.4 Configuração do Django

As configurações de implantação abrangem hosts permitidos, banco de dados, arquivos estáticos, mídia, autenticação social e variáveis de ambiente. O Código 4.2 destaca as configurações requeridas para restringir os hosts aceitos e exigir conexões seguras no ambiente público.

Código 4.2 – Configurações requeridas para ambiente público

```

1 DEBUG = False
2 ALLOWED_HOSTS = [
3     'eduquestify.com.br',
4     'www.eduquestify.com.br'
5 ]
6 SECURE_SSL_REDIRECT = True
7 SESSION_COOKIE_SECURE = True
8 CSRF_COOKIE_SECURE = True

```

Fonte: Elaborada pelo autor (2026).

4.7.5 Banco de Dados e Migrações

O PostgreSQL é configurado por variáveis de ambiente. O comando `python manage.py migrate` aplica a estrutura dos modelos ao banco, e a persistência é acessada pela aplicação por meio do ORM do Django.

4.7.6 Domínio e DNS

O domínio `eduquestify.com.br` é mencionado neste relatório exclusivamente como endereço de hospedagem da plataforma Gamify. O DNS associa esse endereço ao IP público da VPS, permitindo que as requisições sejam encaminhadas ao Virtual Host do OpenLiteSpeed.

4.8 ARQUITETURA DA SOLUÇÃO

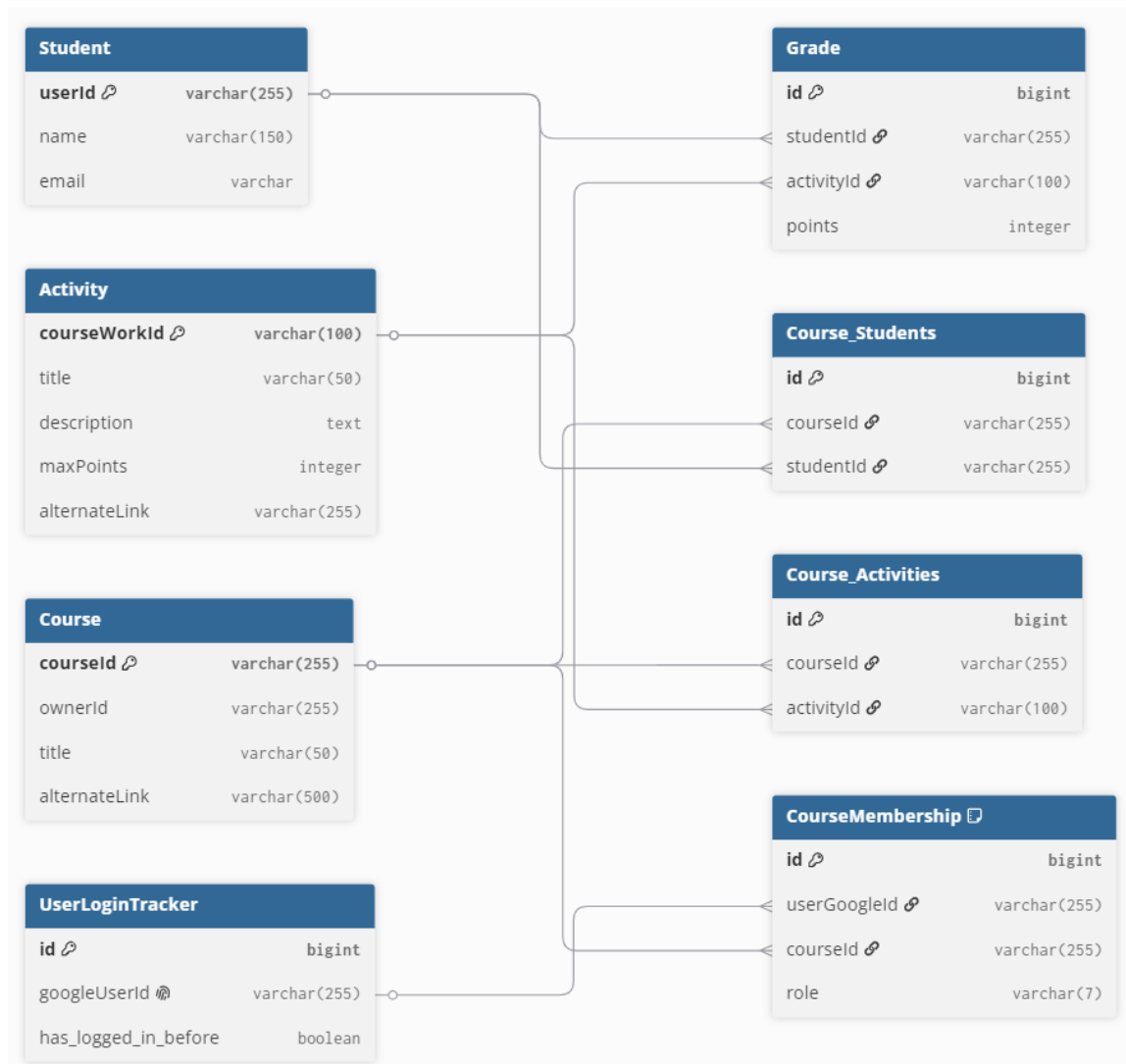
A arquitetura da solução segue o modelo cliente-servidor e o padrão MVT do Django. A camada de apresentação reúne templates, HTML, CSS, JavaScript e Bootstrap; a camada de aplicação concentra views, autenticação, sessões e regras de negócio; a integração externa encapsula OAuth 2.0 e a Classroom API; a persistência utiliza ORM e PostgreSQL; e a camada REST expõe dados em JSON.

O diagrama da Figura 3, apresentado no processo de desenvolvimento, evidencia o fluxo entre essas camadas. A separação reduz o acoplamento entre interface, regras de gamificação, acesso a dados e comunicação externa, permitindo que cada componente seja mantido sem transferir suas responsabilidades aos demais.

4.9 MODELO DE DADOS

As entidades persistidas e seus relacionamentos foram organizados para representar cursos, estudantes, atividades, notas, usuários autenticados e participações. A Figura 5 apresenta o diagrama entidade-relacionamento utilizado para sintetizar essa estrutura de dados.

Figura 5 – Diagrama entidade-relacionamento da plataforma



Fonte: Imagem gerada com ChatGPT (2026).

A Figura 5 evidencia a separação de estudantes, atividades, cursos, notas, usuários rastreados e participações em entidades próprias. Nos modelos, essas estruturas correspondem a Student, Activity, Course, Grade, UserLoginTracker e CourseMembership. Grade associa estudante, atividade e pontuação.

UserLoginTracker registra o identificador Google utilizado no fluxo de autenticação. CourseMembership relaciona esse usuário a um curso e registra seu papel como estudante ou professor. A separação das entidades reduz duplicidade e sustenta a sincronização, a atualização de notas e os cálculos de XP, nível, progresso e ranking.

4.10 INTERFACES E FLUXOS DE UTILIZAÇÃO

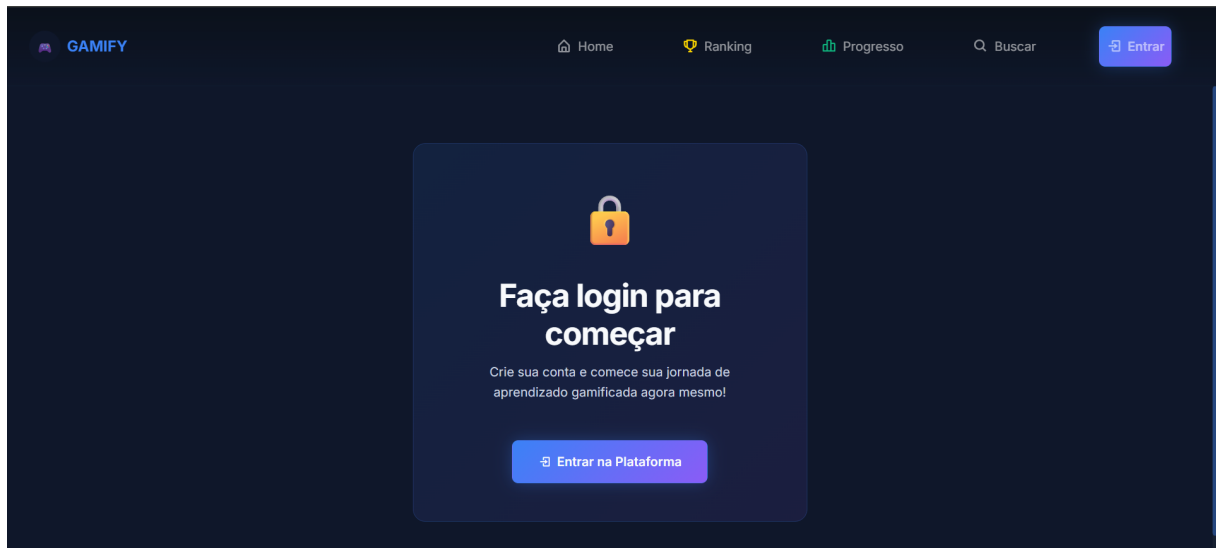
As capturas a seguir apresentam as interfaces implementadas e os principais fluxos disponíveis. Os dados pessoais visíveis foram substituídos por identificações genéricas na versão

final do relatório.

4.10.1 Acesso e Autenticação

O acesso à plataforma inicia em uma página pública que antecede a autorização dos dados acadêmicos. A Figura 6 apresenta essa interface inicial e o direcionamento oferecido ao usuário.

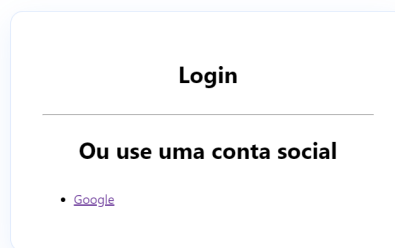
Figura 6 – Tela inicial antes da autenticação



Fonte: Elaborada pelo autor (2026).

A Figura 6 mostra a apresentação da plataforma e o acesso ao fluxo de autenticação. As funcionalidades dependentes do Google Classroom permanecem restritas até a identificação do usuário. A etapa seguinte é apresentada na Figura 7, que registra a interface de login por conta Google.

Figura 7 – Tela de autenticação por conta Google



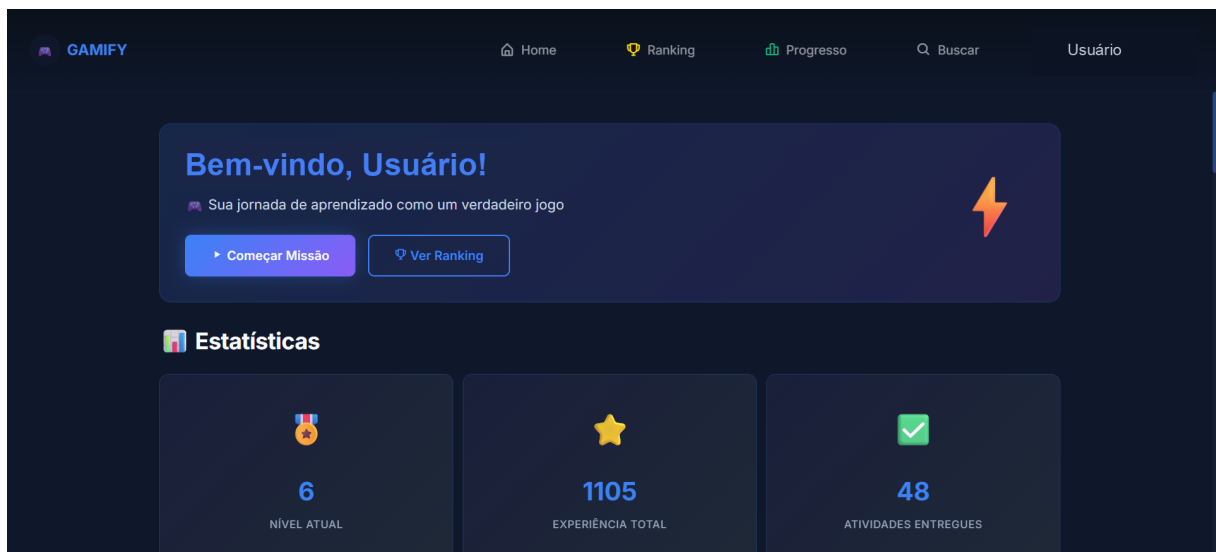
Fonte: Elaborada pelo autor (2026).

Conforme mostrado na Figura 7, o login social inicia a autenticação e a autorização dos escopos necessários à sincronização com o Google Classroom. Essa etapa estabelece a identidade utilizada para consultar e relacionar os dados acadêmicos do usuário.

4.10.2 Dashboard e Cursos

Após a autenticação, os indicadores gerais são reunidos em uma interface central. A Figura 8 apresenta a primeira parte do dashboard disponibilizado ao usuário autenticado.

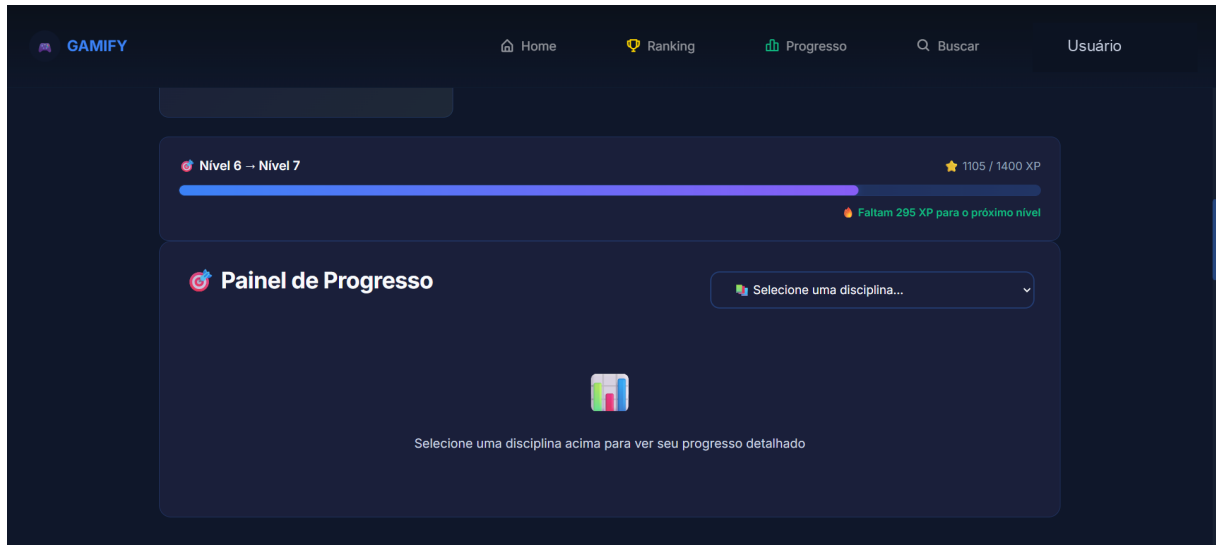
Figura 8 – Dashboard do usuário autenticado



Fonte: Elaborada pelo autor (2026).

A Figura 8 reúne nível, experiência acumulada e quantidade de atividades processadas, apresentando uma visão sintética dos indicadores calculados. Para detalhar a progressão, a Figura 9 apresenta os componentes relacionados ao nível e ao desempenho por disciplina.

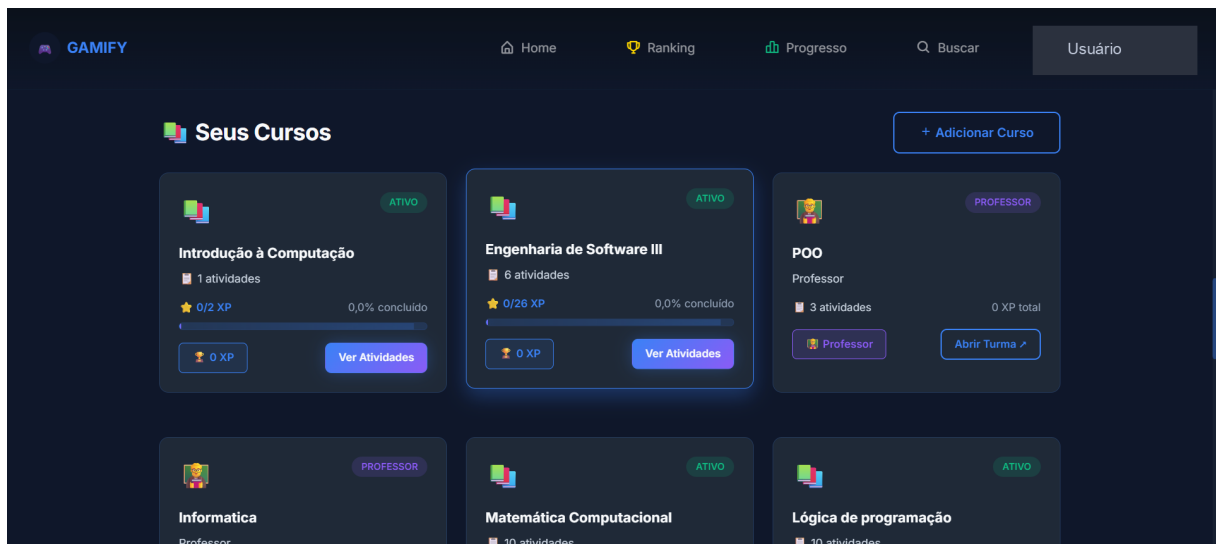
Figura 9 – Indicadores de nível e progresso



Fonte: Elaborada pelo autor (2026).

A Figura 9 apresenta o XP acumulado, a meta do nível seguinte e o painel de progresso por disciplina. A continuidade do dashboard é registrada na Figura 10, que mostra os cursos recuperados pela integração.

Figura 10 – Cursos sincronizados do usuário



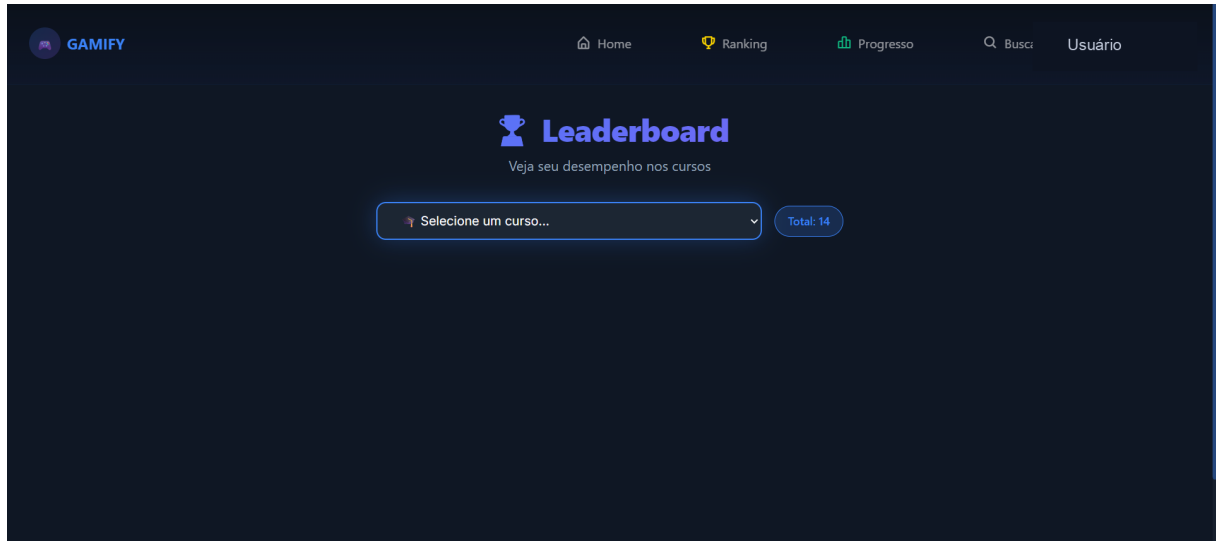
Fonte: Elaborada pelo autor (2026).

Conforme apresentado na Figura 10, os cursos sincronizados são exibidos em cartões com indicadores e links para consulta das atividades ou abertura da turma externa. Essa organização relaciona cada disciplina aos respectivos dados de acompanhamento e às ações disponíveis.

4.10.3 Ranking por Disciplina

O ranking é consultado no contexto de uma disciplina sincronizada. A Figura 11 apresenta a etapa em que o usuário escolhe o curso cujas pontuações serão comparadas.

Figura 11 – Seleção de disciplina para o ranking



Fonte: Elaborada pelo autor (2026).

A Figura 11 delimita a classificação ao curso selecionado, evitando a comparação de pontuações pertencentes a disciplinas distintas. Após essa escolha, a Figura 12 apresenta o destaque visual destinado às maiores pontuações encontradas.

Figura 12 – Destaque das maiores pontuações

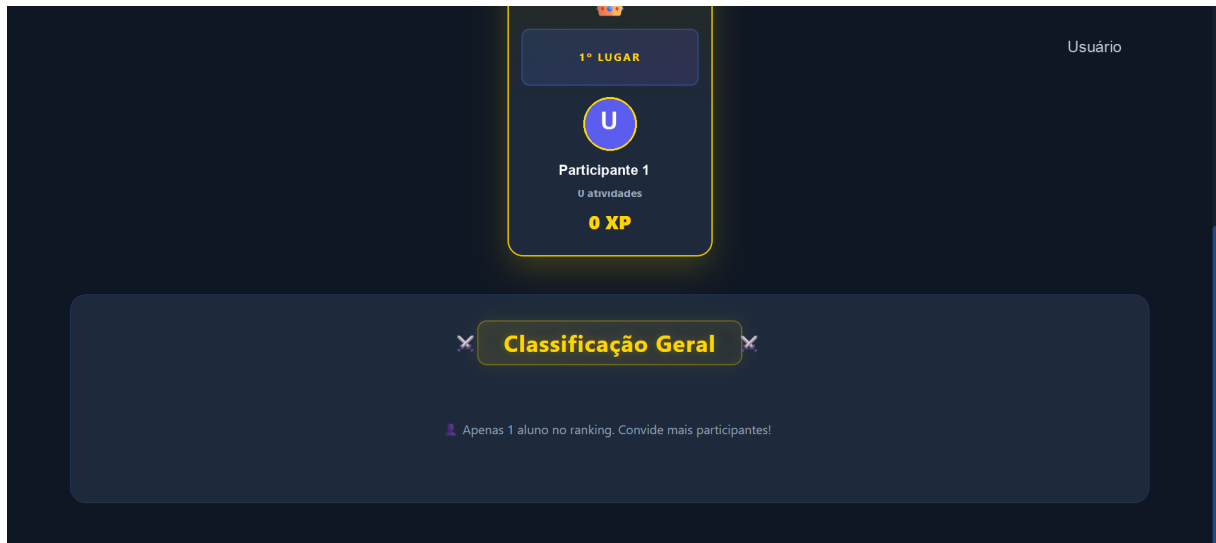


Fonte: Elaborada pelo autor (2026).

Como mostra a Figura 12, as primeiras posições recebem destaque antes da relação completa dos participantes. A Figura 13 complementa essa visualização ao apresentar a classificação

geral calculada para a disciplina.

Figura 13 – Classificação geral do ranking



Fonte: Elaborada pelo autor (2026).

Conforme apresentado na Figura 13, o fluxo exibe a ordenação dos participantes conforme a pontuação calculada. A interface também comunica quando não há dados suficientes para uma classificação mais ampla, mantendo o resultado condicionado às informações sincronizadas para o curso.

4.10.4 Análise de Desempenho

A análise de desempenho também depende da definição do contexto acadêmico examinado. A Figura 14 apresenta a seleção da disciplina utilizada para gerar os indicadores correspondentes.

Figura 14 – Seleção de disciplina para análise



Fonte: Elaborada pelo autor (2026).

A Figura 14 permite restringir os cálculos aos dados de uma disciplina específica. Após a seleção, a Figura 15 apresenta o painel numérico e a representação gráfica produzidos pela aplicação.

Figura 15 – Painel e gráfico de desempenho



Fonte: Elaborada pelo autor (2026).

Conforme demonstrado na Figura 15, a página apresenta quantidades, médias, percentuais e um gráfico que compara os pontos obtidos com a pontuação disponível nas atividades. A combinação dos indicadores permite examinar os dados processados sem alterar as informações originais recuperadas do Google Classroom.

4.10.5 Sincronização de Disciplinas, Atividades e Notas

As ações de sincronização foram reunidas em uma interface que permite consultar novos dados e atualizar registros já persistidos. A Figura 16 apresenta o estado inicial dessa área antes da escolha de uma operação.

Figura 16 – Tela inicial de busca de atividades

Fonte: Elaborada pelo autor (2026).

A Figura 16 organiza separadamente as ações relacionadas a disciplinas, atividades e notas. Para registrar a consulta de atividades ainda não importadas, a Figura 17 apresenta o fluxo de busca associado aos cursos sincronizados.

Figura 17 – Busca de novas atividades

Fonte: Elaborada pelo autor (2026).

Como apresentado na Figura 17, a aplicação permite iniciar uma nova consulta ao serviço externo e incorporar as atividades retornadas ao conjunto local. A Figura 18 apresenta a operação complementar de atualização da nota vinculada a uma atividade já persistida.

Figura 18 – Atualização de nota por atividade

The screenshot shows a web interface for 'GAMIFY'. At the top, there is a navigation bar with links for 'Home', 'Ranking', 'Progresso', 'Busc', and 'Usuário'. The main heading is 'Buscar Missões' with a magnifying glass icon, and a subtitle 'Encontre novas atividades e acompanhe suas tarefas'. Below this, there is a form with three dropdown menus: 'Selecione um Curso' (set to 'teste'), 'O que deseja verificar?' (set to 'Nota Atualizada'), and 'Atividade' (set to 'Quiz de Revisão'). At the bottom of the form are two buttons: 'Pesquisar Atividades' (blue) and '+ Verificar Novas Disciplinas' (green).

Fonte: Elaborada pelo autor (2026).

Conforme mostrado na Figura 18, a atualização seleciona uma atividade existente e solicita novamente sua nota. Em conjunto, as três telas demonstram que a área denominada visualmente “Buscar Missões” permite verificar novas disciplinas, importar atividades e consultar novamente a nota de uma atividade já persistida. Nesse contexto, “missão” é apenas o nome apresentado na interface para uma atividade do Google Classroom.

5 VERIFICAÇÃO TÉCNICA DO SOFTWARE

A verificação técnica foi realizada por meio de testes automatizados implementados com os recursos de teste do Django e da biblioteca `unittest.mock`. Os testes foram organizados em dois conjuntos: aplicação web e API REST. Foram avaliados modelos, rotas, views, regras de gamificação, serialização, endpoints e rotinas de integração. As chamadas à Google Classroom API foram substituídas por objetos simulados, permitindo reproduzir respostas conhecidas sem acessar dados reais ou depender da disponibilidade do serviço externo.

Os resultados apresentados indicam que os cenários incluídos na suíte de testes produziram as respostas esperadas. Entretanto, a aprovação desses testes não representa uma verificação integral do sistema, pois não foram avaliados experiência dos usuários, desempenho sob carga, acessibilidade, segurança dinâmica ou impacto sobre o engajamento.

5.1 CONFIGURAÇÃO DO AMBIENTE DE TESTES

A documentação técnica registra o uso de Python 3.12, Django 5.x e dos recursos nativos de teste do framework. Os componentes que dependem do Google Classroom utilizaram objetos criados com `unittest.mock`, substituindo o cliente externo e suas respostas durante a execução. Essa estratégia torna os cenários determinísticos e evita o uso de credenciais ou dados acadêmicos reais.

O comando padrão para reprodução da suíte em um ambiente com as dependências instaladas é apresentado no Código 5.1. A saída completa do executor e as versões exatas das dependências não foram incluídas nos arquivos fornecidos para elaboração deste relatório.

Código 5.1 – Comando para execução da suíte de testes

```
1 python manage.py test
```

Fonte: Elaborada pelo autor (2026).

O Código 5.1 aciona o executor de testes do Django, responsável por localizar e executar os cenários definidos nas aplicações do projeto. A reprodução pressupõe que o ambiente esteja configurado com as dependências e variáveis necessárias à inicialização do sistema.

5.2 TESTES DA APLICAÇÃO WEB

Os testes da aplicação abrangeram modelos, views, URLs, utilitários e regras de negócio. Foram verificados o rastreamento de usuários, os vínculos com cursos, a renderização das páginas, a proteção de rotas, as respostas JSON de progresso e ranking, a atualização de notas, a sincronização simulada e os cálculos de XP, níveis e posição no ranking. A Tabela 7 apresenta os cenários documentados, a evidência esperada e o resultado registrado para cada verificação.

Tabela 7 – Cenários documentados da aplicação web

Código	Categoria	Funcionalidade verificada	Evidência esperada	Resultado
T01	Modelos	Rastreamento de login e vínculo entre usuário e curso	Criação, representação e integridade dos modelos de gamificação.	Aprovado
T03	Views	Página inicial e contexto do dashboard	Renderização da view home com o contexto esperado.	Aprovado
T04	Views	Proteção do ranking	Redirecionamento do acesso não autenticado.	Aprovado
T05	Views	Página de busca de atividades	Renderização da interface e do conteúdo esperado.	Aprovado
T06	Views	Respostas JSON de progresso e ranking	Estrutura e valores esperados nas respostas.	Aprovado
T07	Views	Atualização de notas	Atualização de Grade com dados simulados.	Aprovado
T08	Integração	Sincronização com o Google Classroom	Criação de rastreador, estudante e curso a partir de respostas simuladas.	Aprovado
T09	URLs	Rotas da aplicação web	Resolução das URLs principais.	Aprovado
T10	Utilitários	Regras de XP, níveis e ranking	Cálculos de experiência, nível e posição.	Aprovado
T11	Utilitários	Funções auxiliares do Classroom	Estruturas geradas a partir das respostas simuladas.	Aprovado

Fonte: Elaborada pelo autor (2026).

5.3 TESTES DA API REST

Os testes da aplicação `api_rest` abrangeram modelos, serializers, views, URLs e endpoints. Foram verificados os relacionamentos entre as entidades acadêmicas, a representação das participações em curso, a estrutura das respostas JSON e o tratamento de usuários ou cursos inexistentes. A Tabela 8 organiza os cenários correspondentes e os resultados obtidos.

Tabela 8 – Cenários documentados da API REST

Código	Categoria	Funcionalidade verificada	Evidência esperada	Resultado
T02	Modelos	Relações entre estudantes, atividades, cursos e notas	Criação e associação dos modelos acadêmicos.	Aprovado
T12	Serializers	Serialização das participações em curso	Estrutura retornada e filtragem de estudantes.	Aprovado
T13	Views/API	Endpoints de participações e notas	Dados e estrutura JSON esperados.	Aprovado
T14	URLs/API	Rotas da API REST	Resolução dos endpoints implementados.	Aprovado
T15	Endpoints/API	Usuário e curso inexistentes	Respostas HTTP e mensagens de erro esperadas.	Aprovado

Fonte: Elaborada pelo autor (2026).

5.4 RESULTADOS E LIMITAÇÕES DOS TESTES

Os 15 cenários documentados apresentaram o comportamento esperado. As evidências abrangeram criação e relacionamento de modelos, resolução de rotas, renderização de views, proteção de acesso, cálculos gamificados, atualização de notas, sincronização com respostas simuladas, serialização e respostas de endpoints.

Os resultados fornecem evidência somente para os comportamentos incluídos nesses cenários. Não foram documentados testes de carga, desempenho, acessibilidade, segurança dinâmica, compatibilidade ampla entre navegadores ou utilização por professores e estudantes. As chamadas externas simuladas também não verificam indisponibilidade real do Google Classroom,

alterações de contrato da API, expiração de credenciais em ambiente público ou variações de latência da rede.

6 CONSIDERAÇÕES FINAIS

O desenvolvimento do software resultou em uma plataforma web capaz de autenticar usuários por meio de conta Google, sincronizar informações do Google Classroom, armazenar dados acadêmicos em banco relacional e gerar indicadores gamificados de acompanhamento. Entre as funcionalidades implementadas estão a sincronização de cursos, atividades, submissões e notas, o cálculo de pontos de experiência e níveis, a apresentação de barras de progresso, rankings e painéis de desempenho, além da disponibilização de endpoints REST.

6.1 RESULTADOS TÉCNICOS ALCANÇADOS

A estrutura adotada integra Python, Django, Django REST Framework, PostgreSQL e serviços Google em uma arquitetura cliente-servidor organizada em camadas. A aplicação também foi disponibilizada em ambiente web, demonstrando a viabilidade técnica de execução fora do ambiente local de desenvolvimento.

A verificação automatizada demonstrou que os 15 cenários documentados apresentaram o comportamento esperado. Esses resultados fornecem evidências sobre as funcionalidades verificadas, incluindo modelos, rotas, views, regras de gamificação, sincronização simulada, serialização e endpoints. Eles não permitem afirmar que a plataforma aumentou o engajamento, melhorou a motivação ou produziu mudanças no comportamento dos estudantes.

6.2 LIMITAÇÕES

Entre as limitações estão a dependência da Google Classroom API, a necessidade de concluir as configurações de segurança do ambiente público e a ausência de avaliações de desempenho, acessibilidade e segurança dinâmica. Também não foram realizadas avaliações com usuários em contexto real.

DECLARAÇÃO SOBRE USO DE INTELIGÊNCIA ARTIFICIAL

O autor declara que ferramentas de inteligência artificial generativa foram utilizadas como apoio à elaboração deste relatório técnico. O ChatGPT foi empregado na revisão textual, na organização do conteúdo, na verificação da clareza do manuscrito e na criação do diagrama entidade-relacionamento da plataforma. O DeepSeek foi utilizado como apoio à criação do design das páginas do frontend. O Gemini foi empregado na criação das imagens do fluxograma da arquitetura em camadas da solução e da topologia de implantação da aplicação. Todo o conteúdo técnico, as decisões metodológicas e arquiteturais, as referências, os elementos visuais e a versão final do relatório foram revisados, verificados e assumidos integralmente pelo autor.

REFERÊNCIAS

- BASS, Len; CLEMENTS, Paul; KAZMAN, Rick. *Software Architecture in Practice*. 4. ed. [S.l.]: Addison-Wesley Professional, 2021. ISBN 9780136886099.
- BOOTSTRAP TEAM. *Bootstrap Documentation*. [S.l.], 2026. Acesso em: 09 jul. 2026. Disponível em: <<https://getbootstrap.com/docs/>>.
- DETERDING, Sebastian et al. From Game Design Elements to Gamefulness: Defining “Gamification”. In: *PROCEEDINGS of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments*. Tampere, Finland: Association for Computing Machinery, 2011. P. 9–15. ISBN 9781450308168. DOI: [10.1145/2181037.2181040](https://doi.org/10.1145/2181037.2181040).
- DJANGO SOFTWARE FOUNDATION. *Django Documentation*. [S.l.], 2026. Acesso em: 09 jul. 2026. Disponível em: <<https://docs.djangoproject.com/>>.
- DJANGO-ALLAUTH CONTRIBUTORS. *django-allauth Documentation*. [S.l.], 2026. Acesso em: 09 jul. 2026. Disponível em: <<https://docs.allauth.org/>>.
- ELMASRI, Ramez; NAVATHE, Shamkant B. *Fundamentals of Database Systems*. 7. ed. [S.l.]: Pearson, 2016. ISBN 9780133970777.
- ENCODE OSS LTD. *Django REST Framework Documentation*. [S.l.], 2026. Acesso em: 09 jul. 2026. Disponível em: <<https://www.django-rest-framework.org/>>.
- FARDO, Marcelo. A gamificação aplicada em ambientes de aprendizagem. *RENOTE*, v. 11, 2013. DOI: [10.22456/1679-1916.41629](https://doi.org/10.22456/1679-1916.41629).
- GIT. *Git Documentation*. [S.l.], 2026. Acesso em: 09 jul. 2026. Disponível em: <<https://git-scm.com/doc>>.
- GITHUB. *GitHub Docs*. [S.l.], 2026. Acesso em: 09 jul. 2026. Disponível em: <<https://docs.github.com/>>.
- GOOGLE. *Google API Client Library for Python Documentation*. [S.l.], 2026. Acesso em: 09 jul. 2026. Disponível em: <<https://googleapis.github.io/google-api-python-client/docs/>>.
- _____. *Google Auth Library for Python*. [S.l.], 2026. Acesso em: 09 jul. 2026. Disponível em: <<https://google-auth.readthedocs.io/>>.
- _____. *Google Classroom API Documentation*. [S.l.], 2026. Acesso em: 09 jul. 2026. Disponível em: <<https://developers.google.com/classroom>>.
- _____. *OAuth 2.0 Playground*. [S.l.], 2026. Acesso em: 09 jul. 2026. Disponível em: <<https://developers.google.com/oauthplayground>>.
- _____. *Using OAuth 2.0 to Access Google APIs*. [S.l.], 2026. Acesso em: 09 jul. 2026. Disponível em: <<https://developers.google.com/identity/protocols/oauth2>>.
- GOOGLE CLOUD. *Google Cloud Documentation*. [S.l.], 2026. Acesso em: 09 jul. 2026. Disponível em: <<https://cloud.google.com/docs>>.

IEEE COMPUTER SOCIETY. *Guide to the Software Engineering Body of Knowledge (SWEBOK Guide): Version 4.0*. Edição: Hironori Washizaki. [S.l.]: IEEE Computer Society, 2024. Acesso em: 02 ago. 2026. Disponível em: <https://www.computer.org/education/bodies-of-knowledge/software-engineering>.

INTERNET CORPORATION FOR ASSIGNED NAMES AND NUMBERS. *The Domain Name System*. [S.l.: s.n.], 2022. Acesso em: 02 ago. 2026. Disponível em: <https://www.icann.org/resources/pages/dns-2022-09-13-en>.

ISO/IEC. *ISO/IEC 25010:2023: Systems and Software Engineering—Systems and Software Quality Requirements and Evaluation (SQuaRE)—Product Quality Model*. [S.l.], 2023. Acesso em: 02 ago. 2026. Disponível em: <https://www.iso.org/standard/78176.html>.

ISO/IEC/IEEE. *ISO/IEC/IEEE 29148:2018: Systems and Software Engineering—Life Cycle Processes—Requirements Engineering*. [S.l.], 2018. Acesso em: 02 ago. 2026. Disponível em: <https://www.iso.org/standard/72089.html>.

LEITÃO, Rui et al. A systematic evaluation of game elements effects on students' motivation. *Education and Information Technologies*, v. 27, 2022. DOI: [10.1007/s10639-021-10651-8](https://doi.org/10.1007/s10639-021-10651-8).

MDN WEB DOCS. *CSS: Cascading Style Sheets*. [S.l.], 2026. Acesso em: 09 jul. 2026. Disponível em: <https://developer.mozilla.org/docs/Web/CSS>.

_____. *HTML: HyperText Markup Language*. [S.l.], 2026. Acesso em: 09 jul. 2026. Disponível em: <https://developer.mozilla.org/docs/Web/HTML>.

_____. *JavaScript*. [S.l.], 2026. Acesso em: 09 jul. 2026. Disponível em: <https://developer.mozilla.org/docs/Web/JavaScript>.

MICROSOFT. *Visual Studio Code Documentation*. [S.l.], 2026. Acesso em: 09 jul. 2026. Disponível em: <https://code.visualstudio.com/docs>.

MOCKAPETRIS, Paul. *Domain Names—Concepts and Facilities*. [S.l.], nov. 1987. Acesso em: 02 ago. 2026. DOI: [10.17487/RFC1034](https://doi.org/10.17487/RFC1034). Disponível em: <https://www.rfc-editor.org/info/rfc1034>.

_____. *Domain Names—Implementation and Specification*. [S.l.], nov. 1987. Acesso em: 02 ago. 2026. DOI: [10.17487/RFC1035](https://doi.org/10.17487/RFC1035). Disponível em: <https://www.rfc-editor.org/info/rfc1035>.

OBJECT MANAGEMENT GROUP. *OMG Unified Modeling Language (OMG UML), Version 2.5.1*. [S.l.], 2017. Acesso em: 02 ago. 2026. Disponível em: <https://www.omg.org/spec/UML/2.5.1>.

POSTGRESQL GLOBAL DEVELOPMENT GROUP. *PostgreSQL Documentation*. [S.l.], 2026. Acesso em: 09 jul. 2026. Disponível em: <https://www.postgresql.org/docs/>.

POSTGRESQL GLOBAL DEVELOPMENT GROUP. *PostgreSQL Documentation: Constraints*. [S.l.], 2026. Acesso em: 02 ago. 2026. Disponível em: <https://www.postgresql.org/docs/current/ddl-constraints.html>.

PRESSMAN, Roger S.; MAXIM, Bruce R. *Software Engineering: A Practitioner's Approach*. 9. ed. [S.l.]: McGraw-Hill Education, 2020. ISBN 9781259872976.

PYTHON SOFTWARE FOUNDATION. *Python Documentation*. [S.l.], 2026. Acesso em: 09 jul. 2026. Disponível em: <<https://docs.python.org/3/>>.

SOMMERVILLE, Ian. *Software Engineering*. 10. ed. [S.l.]: Pearson, 2016. ISBN 9781292096131.

TODA, Armando M. et al. Analysing gamification elements in educational environments using an existing Gamification taxonomy. *Smart Learning Environments*, v. 6, n. 1, 16, 2019. DOI: [10.1186/s40561-019-0106-1](https://doi.org/10.1186/s40561-019-0106-1). Disponível em: <<https://doi.org/10.1186/s40561-019-0106-1>>.

WERBACH, Kevin; HUNTER, Dan. *For the Win: How Game Thinking Can Revolutionize Your Business*. Philadelphia: Wharton Digital Press, 2012. 144 p. ISBN 9781613630235.

ZICHERMANN, Gabe; CUNNINGHAM, Christopher. *Gamification by Design: Implementing Game Mechanics in Web and Mobile Apps*. Sebastopol, CA: O'Reilly Media, 2011. 182 p. ISBN 9781449397678.