Associação de Pais e Amigos dos Excepcionais de Florestópolis

ProAMP

Glossário do projeto

Versão 2.0

Luis Afonso Mineo

Histórico de Revisão

Data	Versão	Descrição	Autor
26/02/2025	1.0	Primeira versão do documento	Luis Afonso Mineo
05/06/2025	2.0	Segunda versão do documento	Luis Afonso Mineo

©Associação de Pais e Amigos
dos Excepcionais de
Florestópolis 2025

Índice

ProAMP | Glossário do Projeto..... 4

1. Termos Gerais do Projeto..... 4
2. Termos Técnicos e de Diagramas..... 6
3. Acrônimos..... 12
4. Traduções dos campos em inglês..... 14

©Associação de Pais e Amigos dos Excepcionais de Florestópolis 2025

ProAMP | Glossário do Projeto

Este glossário define termos chave, conceitos e acrônimos relevantes para o projeto ***ProAMP***, prontuário digital para a escola de educação especial Anna Maria Piettá.

---

1. Termos Gerais do Projeto

1.1. Aluno (Student)

Indivíduo matriculado na escola e paciente do setor clínico, cujos dados cadastrais e de saúde são gerenciados pelo sistema. Nos diagramas, é referido como Student.

1.2. Backend

A parte do sistema que lida com a lógica de negócios, processamento de dados e interações com o banco de dados. No projeto, está sendo desenvolvido com Django e Django REST framework.

1.3. Biblioteca (Programação)

Conjunto de funções, classes e módulos reutilizáveis que fornecem funcionalidades específicas. Em linguagens de programação, bibliotecas ajudam a acelerar o desenvolvimento.

1.4. Controle de Acesso

Mecanismo que define permissões para cada usuário (ator) conforme seu perfil (role) – administrador (sa_admin), gestor (sa_manager) ou profissional da saúde (sa_health_profissional) – restringindo o acesso a funcionalidades e dados específicos do sistema.

1.5. Frontend

A parte do sistema em que o usuário interage diretamente, como botões, menus e imagens. Em outras palavras, é a interface do usuário, a camada que o usuário vê e com a qual interage.

©Associação de Pais e Amigos dos Excepcionais de Florestópolis 2025

1.6. Interface Responsiva

Interface do sistema que se adapta a diferentes tamanhos de tela, como computadores, tablets e dispositivos móveis, garantindo uma boa experiência de usuário em todas as plataformas.

1.7. Prontuário Digital (Medical Record)

Sistema informatizado para registro, armazenamento, consulta e gerenciamento seguro e eficiente das informações clínicas dos alunos/pacientes.

1.8. Registro de Atendimentos (Medical Entry / Medical Record)

Funcionalidade que permite aos profissionais da saúde inserir, visualizar e atualizar dados sobre as consultas, avaliações, evoluções e outros procedimentos realizados com os alunos. No projeto, as entradas de prontuário dos alunos são referidas como MedicalEntry, e a reunião destas entradas, agrupadas pelo id de cada aluno, formam o prontuário completo do mesmo, referido como MedicalRecord.

1.9. Logs de geração de relatório (ReportLog)

Funcionalidade que registra metadados sobre a geração de relatórios de atendimentos (ex: quem gerou, quando, tipo de relatório), sem armazenar o conteúdo do relatório em si no banco de dados principal, para otimizar o armazenamento.

1.10. Usuário (User)

Indivíduo que interage com o sistema, podendo ser um administrador, gestor ou profissional da saúde. Cada usuário possui credenciais de acesso e um perfil (role) que define suas permissões. Nos diagramas, referido como User.

©Associação de Pais e Amigos dos Excepcionais de Florestópolis 2025

2. Termos Técnicos e de Diagramas

2.1. AbstractUser (Django)

Classe base fornecida pelo Django que oferece a implementação fundamental de um sistema de usuário (autenticação, campos como username, password, email, etc.), permitindo customizações como a adição do campo role para diferentes perfis de usuário no projeto.

2.2. API (Application Programming Interface)

Interface que define como diferentes componentes de software devem interagir. No contexto do projeto, o Django REST framework é usado para construir APIs que permitem a comunicação entre o frontend e o backend.

2.3. Ator (Actor)

Entidade externa que interage com o sistema. No Diagrama de Casos de Uso, são representados como sa_admin (administrador do sistema), sa_health_profissional (profissional de saúde) e sa_manager (gestor).

2.4. Atributo (Attribute)

Uma característica ou propriedade de uma entidade ou classe. Por exemplo, name (Nome), dob (Data de Nascimento), email são atributos da entidade Student (Estudante).

2.5. Boundary (Interface)

Elemento em diagramas (como o de Sequência) que representa a interface com a qual o ator interage, como uma tela do sistema (ex: Student Page no diagrama de sequência).

2.6. Cardinalidade (Cardinality)

Indica o número de instâncias de uma entidade que podem estar relacionadas a instâncias de outra entidade (ex: (0,n) – zero para muitos; (1,1) – um para um). Visível nos Diagramas de Entidade-Relacionamento, Lógico e de Classe.

©Associação de Pais e Amigos dos Excepcionais de Florestópolis 2025

2.7. Caso de Uso (Use Case)

Descreve uma sequência de ações que o sistema pode realizar. Exemplos do Diagrama de Casos de Uso: suc_manage_students (Gerenciar Alunos), suc_generate_report (Gerar Relatório).

2.8. Chave Estrangeira (Foreign Key - FK)

Um campo (ou conjunto de campos) em uma tabela que estabelece um link entre os dados em duas tabelas. Ela referencia a chave primária de outra tabela. Indicada com fk_ no Diagrama Lógico (ex: fk_User_uuid na tabela HealthProfile).

2.9. Chave Primária (Primary Key - PK)

Um campo (ou conjunto de campos) que identifica unicamente cada registro em uma tabela do banco de dados. Indicada com um ícone de chave no Diagrama Lógico (ex: uuid na tabela User).

2.10. Classe (Class)

Modelo para criar objetos em programação orientada a objetos. Define atributos e métodos que os objetos terão. Presente no Diagrama de Classes (ex: Student, User, ReportService).

2.11. Controlador (Controller)

Representa a lógica de negócios de uma determinada classe do sistema. No diagrama de sequência, a classe Student (referenciada como Student Entity) manipula a lógica de negócio, através de métodos para operações de estudante. O termo é comumente usado em sistemas MVC, estrutura do qual o Django REST Framework faz uso, porém neste a controller é renomeada como *VIEW*, assim constituindo, em Django o modelo MVT.

2.12. Diagrama de Caso de Uso (Use Case Diagram)

Representação visual das interações entre os atores e o sistema, mostrando as principais funcionalidades (casos de uso) que o sistema oferece.

©Associação de Pais e Amigos dos Excepcionais de Florestópolis 2025

2.13. Diagrama de Classe (Class Diagram)

Tipo de diagrama da UML que descreve a estrutura de um sistema mostrando suas classes, atributos, métodos e os relacionamentos entre elas.

2.14. Diagrama de Entidade-Relacionamento (Entity-Relationship Diagram)

Modelo conceitual que descreve a estrutura de dados de um banco de dados, mostrando entidades, seus atributos e os relacionamentos entre elas.

2.15. Diagrama de Sequência (Sequence Diagram)

Diagrama UML que mostra a ordem temporal das interações entre objetos (ou atores e o sistema) para realizar uma funcionalidade específica ou parte dela.

2.16. Diagrama Lógico (Logic Diagram)

Representação da estrutura do banco de dados como ele será implementado, mostrando tabelas, colunas, tipos de dados, chaves primárias e estrangeiras.

2.17. Django

Framework de desenvolvimento web, escrito em Python. Utilizado no backend do projeto.

2.18. Django App

Um submódulo de um projeto Django, projetado para realizar uma função específica. O projeto está estruturado com apps como "estudantes" e "autenticação".

2.19. Django REST framework (DRF)

Uma biblioteca para construir APIs Web com Django. Utilizada no projeto para criar os endpoints da API.

2.20. Endpoint

URL específica onde uma API pode ser acessada por uma aplicação cliente. Por exemplo, /api/students/ é o endpoint usado para listar ou criar alunos.

©Associação de Pais e Amigos dos Excepcionais de Florestópolis 2025

2.21. Entidade (Entity)

Objeto do mundo real que é relevante para o sistema e sobre o qual os dados são armazenados. No DER, exemplos são Student, User, MedicalEntry.

2.22. Extensão (<<extend>>)

Relacionamento em Diagramas de Caso de Uso onde um caso de uso pode opcionalmente estender o comportamento de outro caso de uso base. Ex: suc_generate_report estende suc_view_students_history.

2.23. HealthProfile (Perfil de Saúde)

Entidade/Classe que armazena informações específicas de profissionais da saúde, como especialidade (speciality) e número do conselho (council_number), vinculada a um User (usuário).

2.24. HTTP Status Codes

Códigos de resposta padrão dados por um servidor web a uma requisição HTTP:

- HTTP_200_OK: Requisição bem-sucedida.
- HTTP_201_CREATED: Requisição bem-sucedida e um novo recurso foi criado.
- HTTP_204_NO_CONTENT: Requisição bem-sucedida, mas não há conteúdo para retornar (usado em operações de exclusão, por exemplo).

2.25. Método (Method)

Uma função associada a uma classe, que define um comportamento ou ação que um objeto daquela classe pode realizar. Ex: save(student), soft_delete(student) na classe Student.

©Associação de Pais e Amigos dos Excepcionais de Florestópolis 2025

2.26. Relacionamentos (Relationships)

Conexões entre classes ou entidades que descrevem como elas interagem ou se associam. Exemplos dos diagramas:

- Generate (Gera): Relação entre User e Report/ReportLog.
- Has (Possui): Relação entre User e HealthProfile.
- Manage (Gerencia): Relação entre User e Student.
- Own (Possui): Relação entre Student e MedicalEntry.
- Receive (Recebe): Relação entre Student e MedicalRecord.
- Register (Registra): Relação entre User e MedicalEntry.
- Uses (Usa): Relação entre Report e ReportService.

2.27. Request (Requisição)

Uma mensagem enviada por um cliente (ex: navegador) para um servidor solicitando uma ação ou recurso.

2.28. Response (Resposta)

Uma mensagem enviada por um servidor a um cliente em resposta a uma requisição.

2.29. Role (Papel/Perfil)

Campo associado à entidade User para definir o perfil do usuário no sistema, determinando suas permissões e acesso.

2.30. Soft Delete (Exclusão Lógica)

Marca um registro como inativo (ex: is_active = false) em vez de removê-lo fisicamente do banco de dados. Permite a recuperação de dados (restore_inactive_student).

2.31. SQL Data Types (Tipos de Dados SQL)

Definem o tipo de dado que uma coluna em uma tabela de banco de dados pode armazenar. Vistos no Diagrama Lógico:

- VARCHAR: Sequência de caracteres de tamanho variável.
- DATE: Armazena uma data (ano, mês, dia).
- BOOLEAN: Armazena valores verdadeiros ou falsos.
- INTEGER: Armazena números inteiros.

©Associação de Pais e Amigos dos Excepcionais de Florestópolis 2025

2.32. UUID (Universally Unique Identifier)

Identificador universalmente único usado como chave primária (pk) para muitas entidades no projeto (ex: Student, User), garantindo que cada registro tenha um ID único globalmente.

2.33. View (Django)

Como mencionado anteriormente, em Django as regras de negócio são definidas em uma camada de controle chamada de *VIEW*.

©Associação de Pais e Amigos dos Excepcionais de Florestópolis 2025

3. Acrônimos

3.1. API (Application Programming Interface)

Interface de Programação de Aplicações.

3.2. CEP (Código de Endereçamento Postal)

Número de identificação postal.

3.3. CGM (Código Geral Matrícula)

Código de identificação único a cada aluno (funciona como um ID).

3.4. CSV (Comma-Separated Values)

Valores Separados por Vírgula. Formato de arquivo para armazenamento de dados estruturados em tabelas.

3.5. DER (Diagrama de Entidade-Relacionamento)

Modelo de dados que representa entidades e seus relacionamentos.

3.6. DOB (Date of Birth)

Data de nascimento.

3.7. DRF (Django REST Framework)

Framework para construir APIs Web com Django.

3.8. FK (Foreign Key)

Chave Estrangeira.

3.9. HTTP (Hypertext Transfer Protocol)

Protocolo de Transferência de Hipertexto.

3.10. HTTPS (Hypertext Transfer Protocol Secure)

Protocolo de Transferência de Hipertexto Seguro. Protocolo seguro de comunicação utilizado entre o navegador e o servidor.

©Associação de Pais e Amigos dos Excepcionais de Florestópolis 2025

3.11. MTTR (Mean Time to Repair)

Tempo Médio para Reparo. Tempo médio necessário para corrigir falhas no sistema.

3.12. MVC (Model-View-Controller)

Padrão de arquitetura de software que separa a aplicação em três camadas: Model (dados), View (interface do usuário) e Controller (gerencia a lógica de negócio). Django segue um padrão similar chamado MVT (Model-View-Template).

3.13. MVT (Model-View-Template)

Termo utilizado no contexto Django, para definir as camadas de um típico projeto MVC. Model, modelo de entidades; View, lógica de negócios; Template, representação visual do sistema.

3.14. PK (Primary Key)

Chave Primária. Identificador primário de uma entidade.

3.15. ProAMP

Prontuário *ANNA MARIA PIETTÁ*. Acrônimo usado como nome para a aplicação desenvolvida neste projeto de estágio.

3.16. SQL (Structured Query Language)

Linguagem de Consulta Estruturada, usada para gerenciar e consultar bancos de dados relacionais.

3.17. UML (Unified Modeling Language)

Linguagem de Modelagem Unificada, usada para especificar, visualizar, construir e documentar os artefatos de um sistema de software.

3.18. UUID (Universally Unique Identifier)

Identificador universalmente único.

3.19. XLSX (Excel Open XML Spreadsheet)

Formato Open XML usado pelo Microsoft Excel.

.

©Associação de Pais e Amigos dos Excepcionais de Florestópolis 2025

4. Traduções dos campos em inglês

4.1. Address (Endereço)

Endereço completo, com rua e número do estudante.

4.2. City (Cidade)

Cidade associada ao CEP cadastrado.

4.3. Council_number (Número de conselho)

Campo com número de cadastro no conselho regional, específico a cada tipo de profissional da saúde (ex: CREFITO → Conselho Regional de Fisioterapia e Terapia Ocupacional; CRP → Conselho regional de Psicologia).

4.4. Created_at (Criado em)

Representa a data de criação daquela entrada no banco de dados.

4.5. Created_by (Criado por)

Campo com id do usuário que cadastrou esta entrada.

4.6. Date (Data)

Data de cadastro da entidade.

4.7. Export_to_xlsx (Exportar para XLSX)

Método para exportar a arquivo de relatório no formato XLSX.

4.8. Gender (Gênero)

Campo de escolha entre masculino, feminino ou outro.

4.9. Guardian (Responsável)

Nome completo do responsável legal pelo estudante.

4.10. Guardian_cpf (CPF do responsável)

CPF de cadastro do responsável.

©Associação de Pais e Amigos dos Excepcionais de Florestópolis 2025

4.11. First_name (Primeiro nome)

Herdado a partir do usuário abstrato do Django, representa apenas o primeiro nome do usuário.

4.12. Generate_history (Gerar histórico)

Método para gerar relatório de histórico.

4.13. Generate_montly (Gerar mensal)

Método para gerar relatório mensal.

4.14. Generated_at (Gerado em)

Campo com a data de geração do relatório.

4.15. Generated_by (Gerado por)

Campo de associação à entidade User para definir quem gerou aquele relatório.

4.16. Healthpro_id (Identificador do profissional de saúde)

Identificador único do profissional que registrou a entrada.

4.17. Id (Identificador)

Valor único à cada instância cadastrada no banco.

4.18. Is_active (É ativo)

Representa o campo que define os alunos ativos, ou inativos.

4.19. Last_name (Último nome)

Herdado a partir do usuário abstrato do Django, representa o último nome do usuário (pode conter o nome do meio também).

4.20. Medrec_description (Descrição de recomendações clínicas)

Campo dedicado à descrição de observações do atendimento clínico.

4.21. Name (Nome)

Nome completo do estudante em uma única entrada.

©Associação de Pais e Amigos dos Excepcionais de Florestópolis 2025

4.22. Report_type (Tipo de relatório)

Campo descritivo de qual tipo de relatório foi gerado (ex: Relatório mensal, histórico e etc...).

4.23. Speciality (Especialidade)

Campo descritivo da especialização do profissional.

4.24. State (Estado)

Estado de residência cadastrado.

4.25. Student_id (Identificador do estudante)

Identificador único do estudante associado a aquela entrada.

4.26. Password (Senha)

Campo de senha, herdado a partir do usuário abstrato do Django.

4.27. Patient_evolution (Evolução do paciente)

Campo descritivo do que foi feito no atendimento.

4.28. Role (Papel)

Já descrito nos *termos técnicos*: Campo associado à entidade User para definir o perfil do usuário no sistema.

4.29. Updated_at (Atualizado em)

Representa a data em que aquela instância foi atualizada.

4.30. Updated_by (Atualizado por)

Campo com id do último usuário a alterar esta entrada.

4.31. User (Usuário)

Campo que faz referência a alguma entidade de User.

©Associação de Pais e Amigos dos Excepcionais de Florestópolis 2025