

DOCUMENTAÇÃO TÉCNICA

Diagramas

Sistema de Gestão de Aulas

ISTEC

Minimum Viable Product (MVP)

Versão 1.0

*Produzido pela equipa técnica: João Figueira, Rodrigo Rodrigues, Ricardo
Pinto, Diogo Ferreira*

Índice

- 1. Visão Geral do Sistema**
- 2. Diagrama de Casos de Uso**
- 3. Diagrama de Classes**
- 4. Diagrama de Fluxo de Dados (DFD)**
- 5. Diagrama de Atividades**

1. Visão Geral do Sistema

O Sistema de Gestão de Aulas é uma aplicação desenvolvida em Python que tem como objetivo digitalizar e automatizar o processo de marcação e controlo de presenças em contexto académico.

O sistema diferencia dois tipos de utilizadores: **Aluno** e **Professor**.

Cada um com permissões e funcionalidades distintas, autenticados através de credenciais guardadas num ficheiro JSON que funciona como base de dados local.

Principais Funcionalidades

- Autenticação de Alunos e Professores com credenciais validadas.
- Abertura e fecho de aulas com cronómetro automático.
- Marcação de presença por parte do Aluno apenas quando a aula está aberta.
- Ordenação das presenças com dois algoritmos: Bubble Sort e Tree Sort.
- Exportação de relatório em formato CSV e PDF.
- Testes de desempenho comparativos entre algoritmos de ordenação.
- Persistência de dados via serialização/deserialização JSON.

Arquitetura em Módulos

O projeto segue uma arquitetura modular, separando responsabilidades em ficheiros distintos:

Ficheiro	Responsabilidade
main.py	Ponto de entrada. Gere o loop principal, o menu de login e o estado global da aula.
classAluno.py	Define a classe Aluno com validação de dados e o menu de ações do aluno.
classProfessor.py	Define a classe Professor com menus, exportação CSV/PDF e gestão da aula.
algoritmos.py	Implementa Bubble Sort e a árvore binária de busca para Tree Sort.
leitorDados.py	Carrega e guarda dados do ficheiro JSON.
testeStress.py	Gera dados aleatórios e compara o desempenho dos dois algoritmos.
database.json	Base de dados local em formato JSON com alunos e professores.

2. Diagrama de Casos de Uso

2.1 Descrição

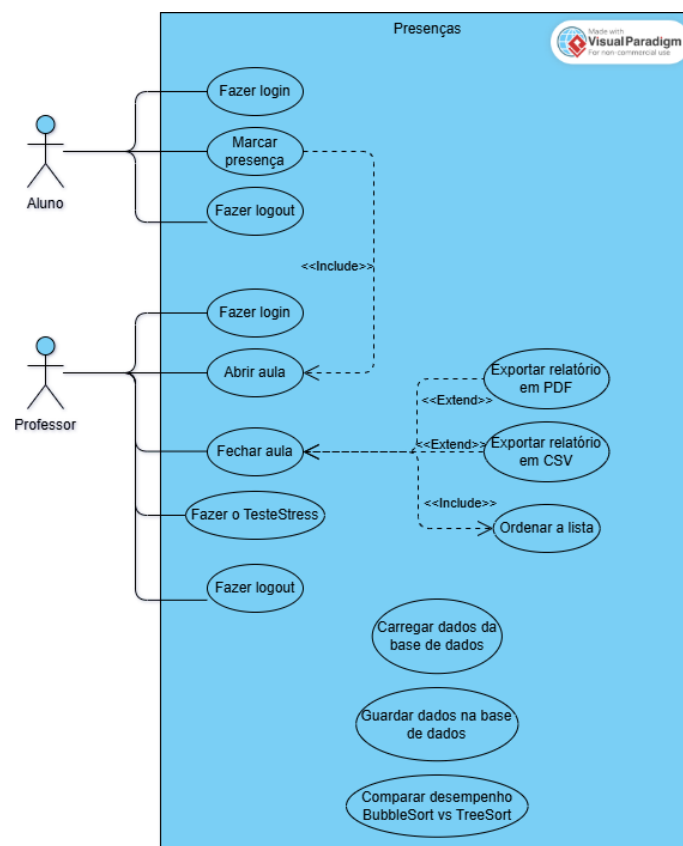
O Diagrama de Casos de Uso representa as interações entre os atores externos (utilizadores) e o sistema, identificando as funcionalidades disponíveis para cada tipo de utilizador.

Existem dois atores principais no sistema: o **Aluno** e o **Professor**. Ambos interagem com o sistema através de menus de autenticação e, após login, acedem às suas funcionalidades exclusivas.

2.2 Atores

- Aluno: utilizador com acesso restrito às funcionalidades do professor, apenas pode marcar a sua presença quando uma aula está ativa.
- Professor: utilizador com controlo total da aula, acesso a relatórios de aulas e testes de stress.

2.3 Representação do Diagrama



Feito com: Visual Paradigm

2.4 Casos de Uso Detalhados

Caso de Uso	Ator	Descrição
Login no Sistema	Aluno / Professor	Autenticação com ID numérico e password. Valida credenciais na base de dados JSON.
Marcar Presença	Aluno	Regista a presença do aluno (número + nome) na lista da aula ativa. Previne duplicados.
Abrir Aula	Professor	Ativa a aula, inicia o cronómetro e reinicia a lista de presenças.
Fechar Aula	Professor	Encerra a aula, calcula a duração e ordena as presenças antes de exibir e exportar.
Exportar CSV	Professor	Serializa a lista de presenças ordenadas num ficheiro CSV.
Exportar PDF	Professor	Gera um relatório formatado em PDF com os dados da aula (requer fpdf2).
Teste de Stress	Professor	Gera N registos aleatórios e executa Bubble Sort e Tree Sort para medir e comparar tempos.

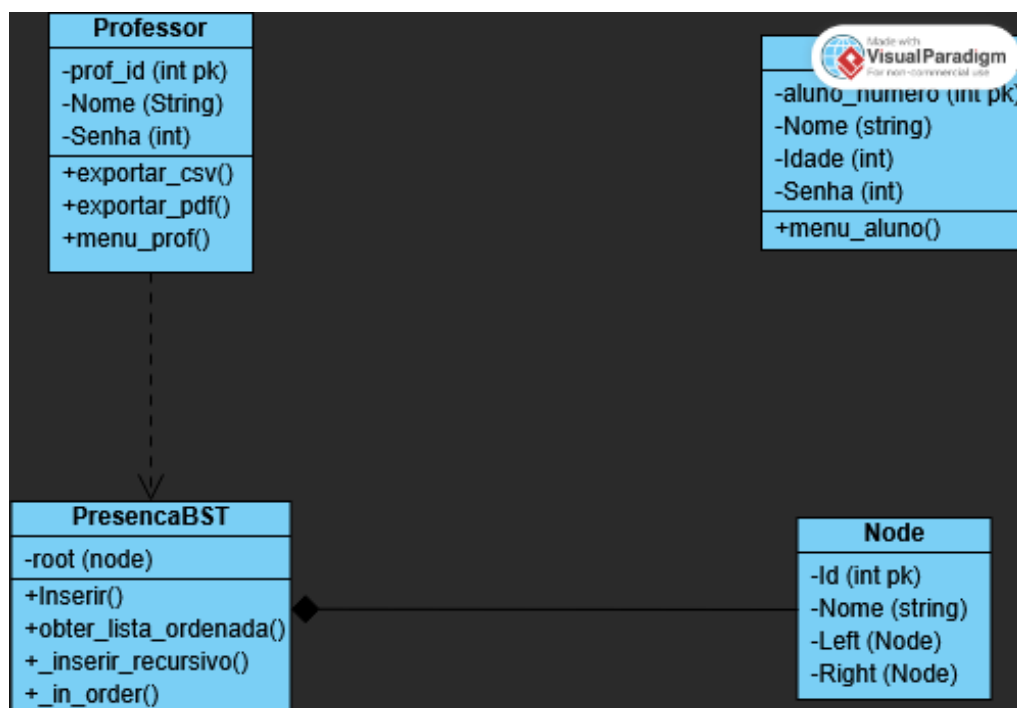
3. Diagrama de Classes

3.1 Descrição

O Diagrama de Classes representa a estrutura estática do sistema, mostrando as classes existentes, os seus atributos, métodos e as relações entre elas.

Neste sistema existem duas classes principais: o **Aluno** e o **Professor**, que agregam os dados e comportamentos dos dois tipos de utilizadores. Adicionalmente, as classes Node e PresencaBST representam uma conexão e suporte à estrutura de dados.

3.2 Representação do Diagrama



Feito com: Visual Paradigm

3.3 Descrição das Classes

Classe Aluno

Representa um estudante inscrito no sistema. É instanciada com dados provenientes do ficheiro JSON e valida os tipos de dados no construtor, lançando `ValueError` em caso de dados inválidos. O método `menu_aluno()` gere o ciclo de interação do aluno, incluindo a marcação de presença com verificação de duplicados.

Classe Professor

Representa um docente com privilégios de gestão da aula. Além dos atributos de identificação, possui métodos de exportação (CSV e PDF) e o método `menu_professor()` que oferece um painel completo de controlo: abrir/fechar aula, exportar relatórios e executar testes de stress.

Classe PresencaBST

Implementa uma Árvore Binária de Busca (Binary Search Tree) para armazenar e ordenar presenças. A propriedade fundamental da BST garante que todos os nós à esquerda têm ID menor que o pai, e todos à direita têm ID maior. O percurso in-order (esquerda → raiz → direita) produz os elementos ordenados de forma crescente sem necessidade de algoritmo de ordenação adicional.

Classe Node

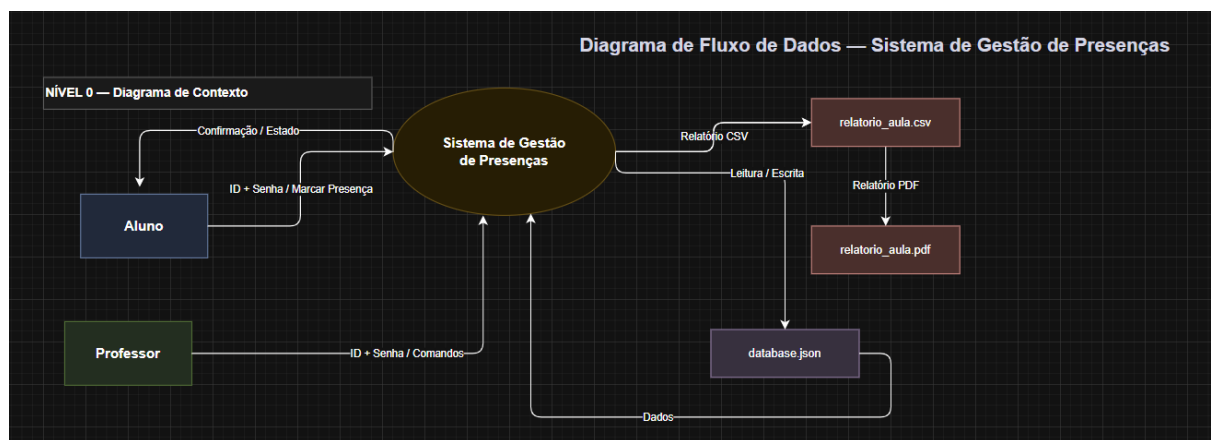
Nó elementar da árvore BST. Armazena o ID e nome do aluno, e mantém referências para os filhos esquerdo e direito. É exclusivamente utilizada pela classe `PresencaBST`.

4. Diagrama de Fluxo de Dados

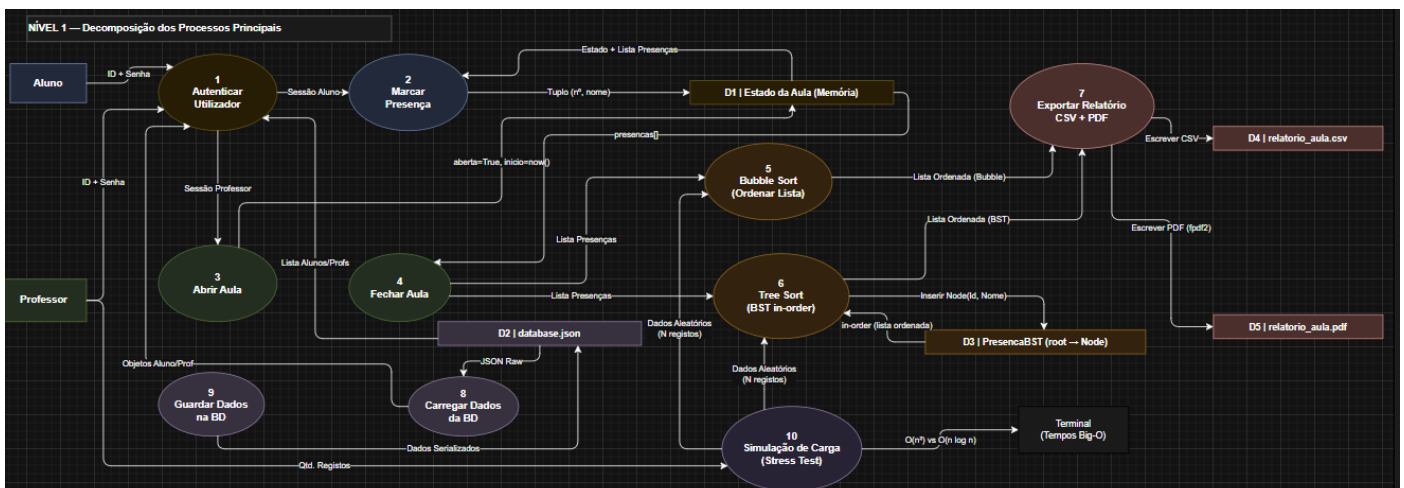
4.1 Descrição

O Diagrama de Fluxo de Dados representa como os dados fluem entre os processos, armazenamentos e entidades externas do sistema.

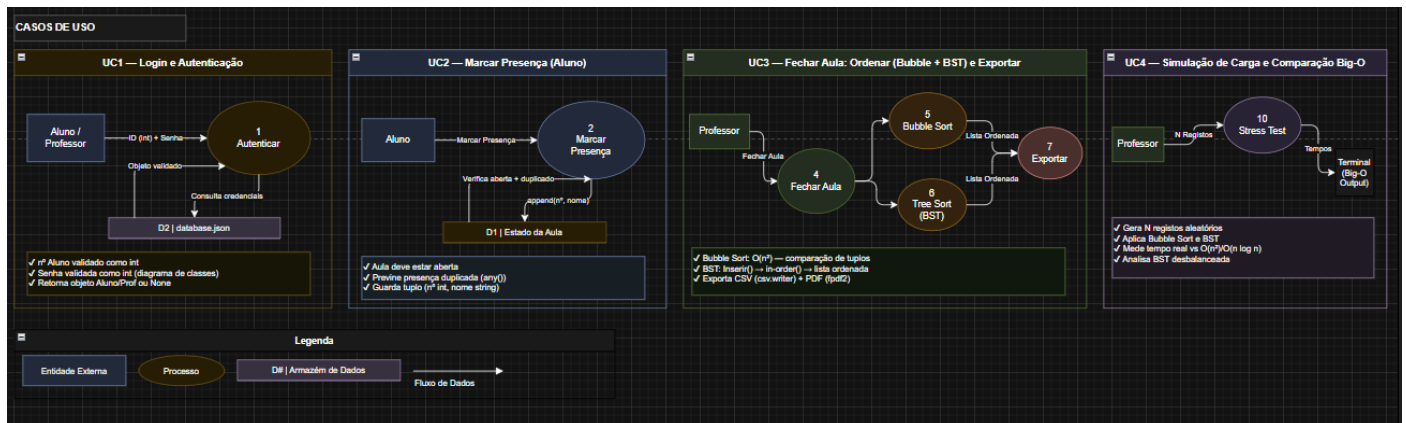
4.2 DFD Nível 0 — Contexto



4.3 DFD Nível 1 — Processos Principais



4.3 DFD — Casos de Uso



5. Diagrama de Atividades

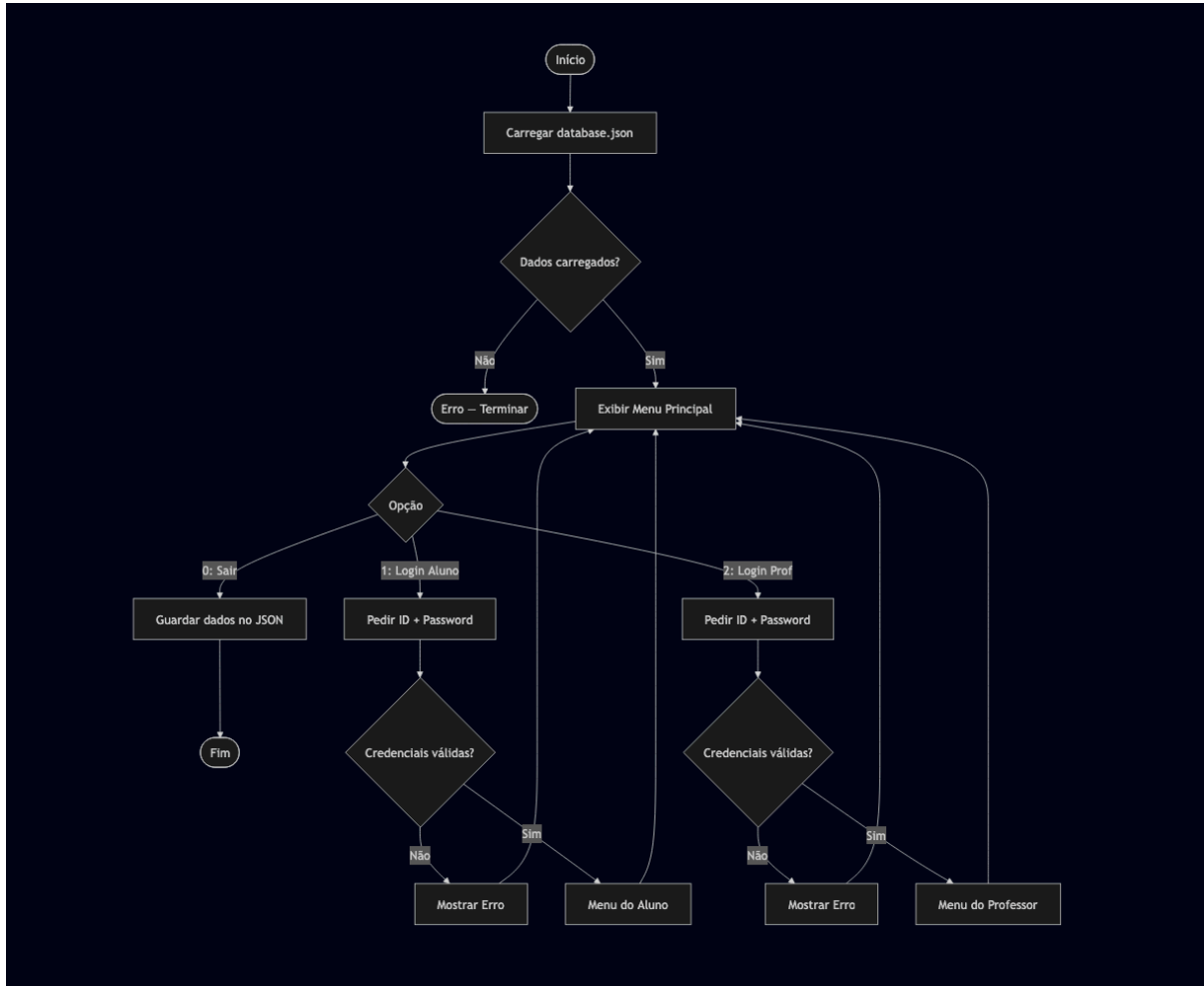
5.1 Descrição

O Diagrama de Atividades representa o fluxo de controlo de um processo, mostrando a sequência de ações, decisões e estados finais.

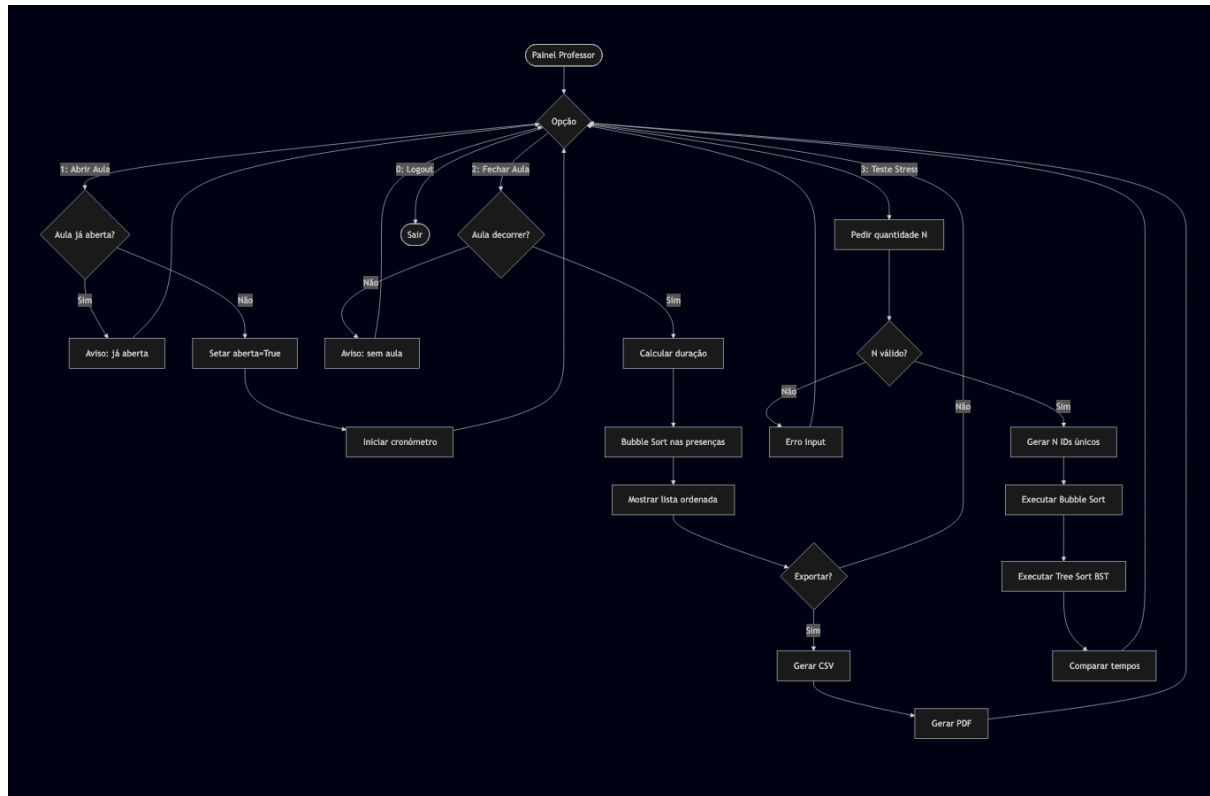
Neste documento são apresentados **três fluxos de atividade**:

- fluxo principal de autenticação e navegação
- fluxo de gestão de aula pelo Professor
- fluxo de marcação do Aluno.

5.2 Fluxo Principal — Login e Navegação



5.3 Fluxo de Atividades — Gestão de Aula pelo Professor



5.3 Fluxo de Atividades — Autenticação Aluno

