



**INSTITUTO SUPERIOR DE EDUCAÇÃO DO RIO DE JANEIRO -
ISERJ/FAETEC**

TÉCNICO EM INFORMÁTICA

Gabriel B. Casanova, Gabriel da S. Ferreira, Gabriel H. V. da Cunha, Matheus
G. F. Menezes

EDUSYST

Sistema de Gestão Escolar para Escolas de Ensino Médio

RIO DE JANEIRO

2024

GABRIEL B. CASANOVA, GABRIEL DA S. FERREIRA, GABRIEL H. V.
DA CUNHA, MATHEUS G. F. MENEZES

EDUSYST: Sistema de Gestão Escolar para Escolas de Ensino Médio

Trabalho de Conclusão de Curso apresentado
ao Instituto Superior de Educação do Rio
de Janeiro - ISERJ/FAETEC como parte dos
requisitos para a obtenção do título de Técnico
em Informática.

Orientador: Prof. Dr. Marcus Vinicius
dos Santos Claro

RIO DE JANEIRO

2024

AGRADECIMENTOS

O desenvolvimento do EduSyst foi o resultado de nove meses de muita luta, dedicação e aprendizado, marcando a culminação dos três anos de formação técnica no ISERJ. Mais que um projeto, a experiência de trabalho em equipe foi forjada graças a um companheirismo que só o convívio e a parceria de três anos de estudo podem proporcionar.

O EduSyst foi idealizado para ir além de um simples trabalho de conclusão de curso. Nosso objetivo sempre foi desenvolver um projeto verdadeiramente open-source que se conectasse profundamente às necessidades reais e aos desafios do ensino público no Brasil. Nosso sonho é que este projeto se expanda, receba contribuições e alcance patamares que vão além do que qualquer um de nós poderia imaginar.

A produção deste trabalho de conclusão de curso contou com a ajuda de diversas pessoas, das quais agradecemos:

Aos professores do corpo docente de informática, que durante todo esse processo nos acompanharam pontualmente, dando todo o auxílio necessário para a elaboração do projeto.

Aos nossos amigos e colegas de turma, que estiveram do nosso lado apoiando e incentivando todas as ideias malucas que surgiam em nossas cabeças.

Às nossas famílias, que nos incentivaram a cada momento, acreditando nas nossas escolhas e oferecendo apoio sempre que precisamos.

A todos que, de alguma forma, contribuíram para que o EduSyst fosse muito além de uma ideia, nosso sincero e profundo agradecimento.

"A tecnologia não é nada. O importante é ter fé nas pessoas, acreditar que elas são boas e inteligentes, e que, se você lhes der as ferramentas, elas farão coisas maravilhosas com elas."

– STEVE JOBS

SUMÁRIO

1	Logo	1
2	Introdução	1
3	Descrição	2
3.1	Objetivo Geral	2
3.2	Abrangência do Sistema	2
3.3	Atores	2
4	Impactos na Implantação do Sistema.....	3
5	Metodologia do Desenvolvimento	3
6	Minimundo	4
7	Regras do Negócio	5
7.1	Requisitos Funcionais	5
7.2	Requisitos Não Funcionais	6
8	Diagrama de Caso de Uso Geral.....	8
9	Desktop.....	9
9.1	Diagrama de Caso de Uso Desktop	9
9.2	Capturas de Tela Desktop	18
10	Web	20
10.1	Diagrama de Caso de Uso Web	20
10.2	Capturas de Tela Web	33
11	Android	35
11.1	Diagrama de Caso de Uso Android	35
11.2	Descrição do Sistema “EduSYST” Android	36
11.3	Capturas de Tela Android	37
12	Banco de Dados.....	38
12.1	Diagrama de Entidade-Relacionamento	38

12.2 Diagrama Lógico	39
12.3 Diagrama de Classes	40
12.4 Observações Sobre a Entidade Notas	41
12.5 Backup e Particionamento dos Dados	41
12.6 Amostragem de Dados Fictícios	42
13 Prints do Núcleo do Sistema	43
13.1 Java Web / JSP	43
13.2 Android Studio / Java / PHP	49
13.3 Java Desktop	52
13.4 MySQL	55
14 Conclusão	56

LISTA DE FIGURAS

1	Logo EduSyst	1
2	Ícone EduSyst	1
3	DCU Geral.....	8
4	DCU Desktop	9
5	Telas - Login, Cadastro e Alteração de Alunos e Consulta de Turmas	18
6	Telas - Enturmação, Desenturmação, Consulta e Exclusão de Alunos	19
7	DCU WEB.....	20
8	Páginas - Página de usuário, Alunos, Turmas e Configurações do Perfil.....	33
9	Páginas - Grade de Horários, Aplicação de Notas, Listagem de Professores e Chat	34
10	DCU Android	35
11	Telas - Login, Opções, Matérias, Professores, Horários, Turma e Boletim	37
12	DER.....	38
13	Diagrama Lógico.....	39
14	Diagrama de Classes	40
15	Tabela Administradores.....	42
16	Tabela Responsáveis	42
17	Tabela Alunos	42
18	Tabela Professores	42
19	Tabela Matérias.....	43
20	Tabela Horários.....	43
21	Conexão com o Banco de Dados MySQL	43
22	DAO do Aluno	44
23	Lançamento de Nota pelos Professores (print n.1)	45
24	Lançamento de Nota pelos Professores (print n.2)	46

25	Ambiente do Usuário (print n.1)	47
26	Ambiente do Usuário (print n.2)	48
27	Conexão com o Banco de Dados MySQL	49
28	Obter Informações de um Professor.....	49
29	Login Responsável	50
30	Geração do Boletim	51
31	Enturmação	52
32	Cadastro de Alunos	53
33	Cadastro de Turmas	53
34	Cadastro de Professores.....	53
35	Listagem de Turmas.....	54
36	Backup das Tabelas	55

1. LOGO



Figura 1: Logo EduSyst



Figura 2: Ícone EduSyst

2. INTRODUÇÃO

A complexidade na organização de dados, a comunicação deficiente entre alunos, professores e pais, juntamente com as dificuldades na gestão acadêmica e de recursos, além dos desafios de transparência e segurança de dados, são problemas enfrentados pelas escolas no dia a dia. Para enfrentar esses desafios e otimizar a administração escolar, o EduSyst foi desenvolvido como uma solução abrangente, eficiente e segura, que integra todas as funções necessárias para melhorar o gerenciamento acadêmico e a comunicação no ambiente escolar.

3. DESCRIÇÃO

3.1. OBJETIVO GERAL

O objetivo do EduSyst é oferecer um sistema integrado que permita aos alunos acessar suas notas, matérias, horários e atividades virtuais; aos professores, atribuir notas, publicar atividades e acompanhar o desempenho de suas turmas; aos responsáveis, monitorar o progresso dos alunos; e aos administradores, gerenciar de forma eficiente professores, alunos, responsáveis e turmas. O sistema busca simplificar a administração escolar, melhorar a comunicação entre todos os envolvidos e promover um gerenciamento mais eficiente e transparente.

3.2. ABRANGÊNCIA DO SISTEMA

O EduSyst abrange a gestão de dados acadêmicos, permitindo a administração de informações dos alunos, como notas, turmas, matérias, e horários. A plataforma inclui ferramentas para o cadastro e gerenciamento de usuários (alunos, professores, responsáveis e administradores), turmas e matérias, comunicação via chat entre usuários do sistema, geração de boletins e criação de grade horária.

Os responsáveis só possuem associação com 1 (um) aluno no sistema. Por motivos de segurança, o sistema não possui recuperação de senha ao nível usuário (Responsável, Professor e Aluno). O usuário precisa solicitar ao administrador a mudança de sua senha. Também incorpora mecanismos para os administradores atualizarem dinamicamente as informações institucionais e gerenciarem aspectos operacionais da escola, incluindo matrículas, turmas e a atualização de conteúdo no ambiente web. O sistema não inclui criptografia de senhas por padrão, ficando sob responsabilidade do implementador assegurar a implementação de um mecanismo de criptografia adequado.

3.3. ATORES

Os atores são os diferentes tipos de usuários que interagem com o sistema e desempenham funções específicas. Cada ator possui responsabilidades e permissões distintas para garantir o funcionamento adequado da plataforma.

- Administradores
 - Descrição: Usuários responsáveis pela gestão do sistema, incluindo a criação, atuali-

- zação e exclusão de dados relacionados a usuários, turmas e disciplinas.
- Atributos: ID, Nome de Usuário, Senha, E-Mail.
 - Alunos
 - Descrição: Alunos matriculados na escola que utilizam o sistema.
 - Atributos: ID, CPF, E-Mail, Nome, Data de Nascimento, Gênero, Endereço, Telefone, Curso, Turmas associadas.
 - Responsáveis
 - Descrição: Pais ou responsáveis pelos alunos, que acompanham o desenvolvimento acadêmico de seu aluno associado.
 - Atributos: ID, CPF, E-Mail, Nome, Data de Nascimento, Gênero, Endereço, Telefone, Curso, Aluno associado.
 - Professores
 - Descrição: Professores que utilizam o sistema para gerenciar suas turmas e atividades acadêmicas.
 - Atributos: ID, CPF, E-Mail, Nome, Data de Nascimento, Gênero, Endereço, Telefone, Horários e Disciplinas associadas.

4. IMPACTOS NA IMPLANTAÇÃO DO SISTEMA

O EduSyst promove uma transformação social significativa ao modernizar a gestão educacional nas escolas públicas. Ao melhorar a organização e comunicação, o sistema facilita uma educação mais acessível e personalizada, reduzindo desigualdades e fortalecendo o suporte parental.

5. METODOLOGIA DO DESENVOLVIMENTO

Para o desenvolvimento do EduSyst, adotamos uma abordagem colaborativa e eficiente utilizando ferramentas de gestão e controle de versão. O Google Drive foi empregado para o compartilhamento e organização de documentos, facilitando o armazenamento e o acesso aos materiais do projeto, como especificações, relatórios e documentos de design. O GitHub foi utilizado para o controle de versão e a colaboração no código-fonte, permitindo a integração contínua das alterações e a gestão de diferentes versões do projeto. Essa combinação de ferra-

mentas garantiu uma coordenação eficaz entre os membros da equipe, possibilitando um desenvolvimento ágil e bem documentado do sistema. O desenvolvimento do EduSyst foi realizado em três etapas distintas, cada uma com avanços significativos:

- Na primeira etapa, criamos um site estático com tela de login, utilizando Bootstrap para a construção de uma interface básica, e desenvolvemos um aplicativo desktop com funcionalidades iniciais de login para administradores e uma interface básica para operações CRUD.
- Na segunda etapa, implementamos a conexão com um banco de dados MySQL e transformamos o site em uma aplicação dinâmica utilizando JSP. Introduzimos o login para alunos e a capacidade de listar matérias. O aplicativo desktop foi aprimorado para incluir login de administradores, recuperação de senhas, listagem de alunos, além de permitir o cadastro, alteração e exclusão de alunos.
- Na terceira e última etapa, concluímos o projeto com a integração de todos os requisitos funcionais previstos. O sistema agora inclui um site público com login para professores, alunos e responsáveis, um aplicativo desktop completo para a administração do sistema escolar, e um aplicativo móvel para o acesso exclusivo de alunos e responsáveis.

6. MINIMUNDO

O EduSyst enfrenta o desafio de simplificar a complexidade na organização de dados e melhorar a comunicação entre alunos, professores e pais, buscando solucionar as dificuldades na gestão acadêmica das escolas públicas de ensino médio no estado do Rio de Janeiro. Seu propósito primordial é desenvolver e implementar sistemas escolares que ofereçam uma plataforma abrangente e intuitiva para atender às variadas demandas do ambiente escolar.

No sistema escolar, anualmente dividido em quatro bimestres, os alunos podem participar de múltiplas turmas. Eles têm acesso à lista de professores e matérias atribuídas, além de gerar boletins bimestrais baseados nas notas definidas pelos professores, e acessar materiais de estudo disponibilizados pelo sistema. Os alunos possuem os atributos ID, CPF, E-Mail, Nome, Data de Nascimento, Gênero, Endereço, Telefone, Curso, além das turmas na qual ele está associado.

Os professores têm a possibilidade de listar todas as turmas que lecionam, juntamente com seus respectivos alunos e matérias atribuídas. Eles também necessitam da capacidade de

atribuir notas individualmente a cada aluno em suas respectivas matérias. Eles também podem aplicar atividades para suas turmas, avaliando o desempenho de cada aluno de forma personalizada. Os professores possuem os atributos ID, CPF, E-Mail, Nome, Data de Nascimento, Gênero, Endereço, Telefone, além das turmas e disciplinas na qual ele está associado.

O EduSyst facilita a comunicação entre usuários por meio de chats de turmas. Essa funcionalidade permite que professores compartilhem informações importantes e respondam a dúvidas dos alunos de forma rápida e direta.

Os Responsáveis dos alunos têm a possibilidade de acompanhar o seu desenvolvimento, através do registro dos resultados e notas de avaliações e atividades, além da possibilidade de contactar os administradores do ambiente escolar. Os Responsáveis possuem os atributos ID, CPF, E-Mail, Nome, Data de Nascimento, Gênero, Endereço, Telefone e o aluno na qual está associado.

Por fim, os administradores do sistema desempenham um papel crucial ao gerenciar matrículas, professores, alunos, turmas, matérias, horários e notas. Além disso, têm a responsabilidade de atualizar dinamicamente as informações do site do sistema escolar. Ao cadastrar usuários (alunos e professores), os administradores definem um login de identificação para acesso à seção institucional do site.

Com funcionalidades adaptadas às necessidades de cada usuário e uma interface intuitiva, o sistema contribui para a transparência, o acompanhamento acadêmico e a otimização da administração escolar, alinhando-se aos objetivos de modernização e eficiência das escolas públicas de ensino médio no Rio de Janeiro. O EduSyst apresenta uma solução abrangente e eficiente para os desafios da gestão escolar, promovendo a organização de dados e a comunicação entre alunos, professores, responsáveis e administradores.

7. REGRAS DO NEGÓCIO

7.1. REQUISITOS FUNCIONAIS

- Aplicativo desktop para acesso de Administradores.
- Ambiente web para acesso de Professores, Alunos e Responsáveis.
- Aplicativo móvel para acesso exclusivo de Professores, Alunos e Responsáveis, com funcionalidades limitadas.
- Professores:

- Consultar matérias, alunos e turmas atribuídas.
 - Atribuir atividades on-line.
 - Atribuir notas aos alunos.
- Alunos:
 - Consultar matérias, notas, grade de horários, atividades, turmas e professores.
 - Realizar atividades.
 - Administradores:
 - Gerenciar matrículas, professores, alunos, turmas, matérias, responsáveis, notas e horários.
 - Imprimir informações como listagem de alunos e matérias.

Responsáveis:

- Acompanhar o desenvolvimento dos alunos
- Observar matérias, notas, grade de horários, turmas atividades e professores do aluno.

7.2. REQUISITOS NÃO FUNCIONAIS

- API Externa para sistema de chat.
- Software:
 - Utilizamos os sistemas operacionais Windows 10 e Windows 11 para o desenvolvimento do EduSyst e o projeto foi feito com eles em mente.
 - Execução do sistema:
 - Java Development Kit (JDK) 22 para execução do sistema desktop.
 - XAMMP 8.2.12 com Tomcat 8.5.96 e PHP 8.2.12 para criação do servidor web local do website.
 - Java Runtime Environment (JRE) 22 para execução do website no servidor web.
 - MariaDB 10.4.32 e PhpMyAdmin 5.2.1 para execução e manutenção do banco de dados MySQL no servidor web.
 - Google Chrome 128 ou similar para exibição ideal do website.
 - Ambiente de Desenvolvimento:
 - Apache NetBeans 22 para desenvolvimento Java Desktop, Java Web, PHP, HTML5 e CSS.
 - Android Studio 2023.3.1 Patch 2 para desenvolvimento Android.
 - Bibliotecas:

- MySQL Connector (JDBC) 9 para conexão do site e do sistema desktop Java com o banco de dados MySQL.
 - O'Reilly Servlet Package (cos.jar) para envio de arquivos pro servidor
 - Bootstrap para design HTML.
 - API do MeuMural para sistema de chat.
 - Visual Paradigm 17 Community Edition para design de diagramas de classe.
 - Astah UML 9.2 para design de diagramas de caso de uso (DCUs).
 - Draw.io para design do diagrama de entidade relacionamento (DER).
 - Overleaf (LaTeX) para preparação, edição e formatação do texto do TCC.
 - GitHub e Google Drive Desktop para desenvolvimento em equipe, backup e versionamento.
- Hardware:
 - Processador Intel® Core™ i5-5200U.
 - 8GB Memória Ram.
 - Monitor com resolução mínima de 1366 x 768 (desktop).
 - Tela com resolução mínima de 720 x 1280 (mobile).

8. DIAGRAMA DE CASO DE USO GERAL

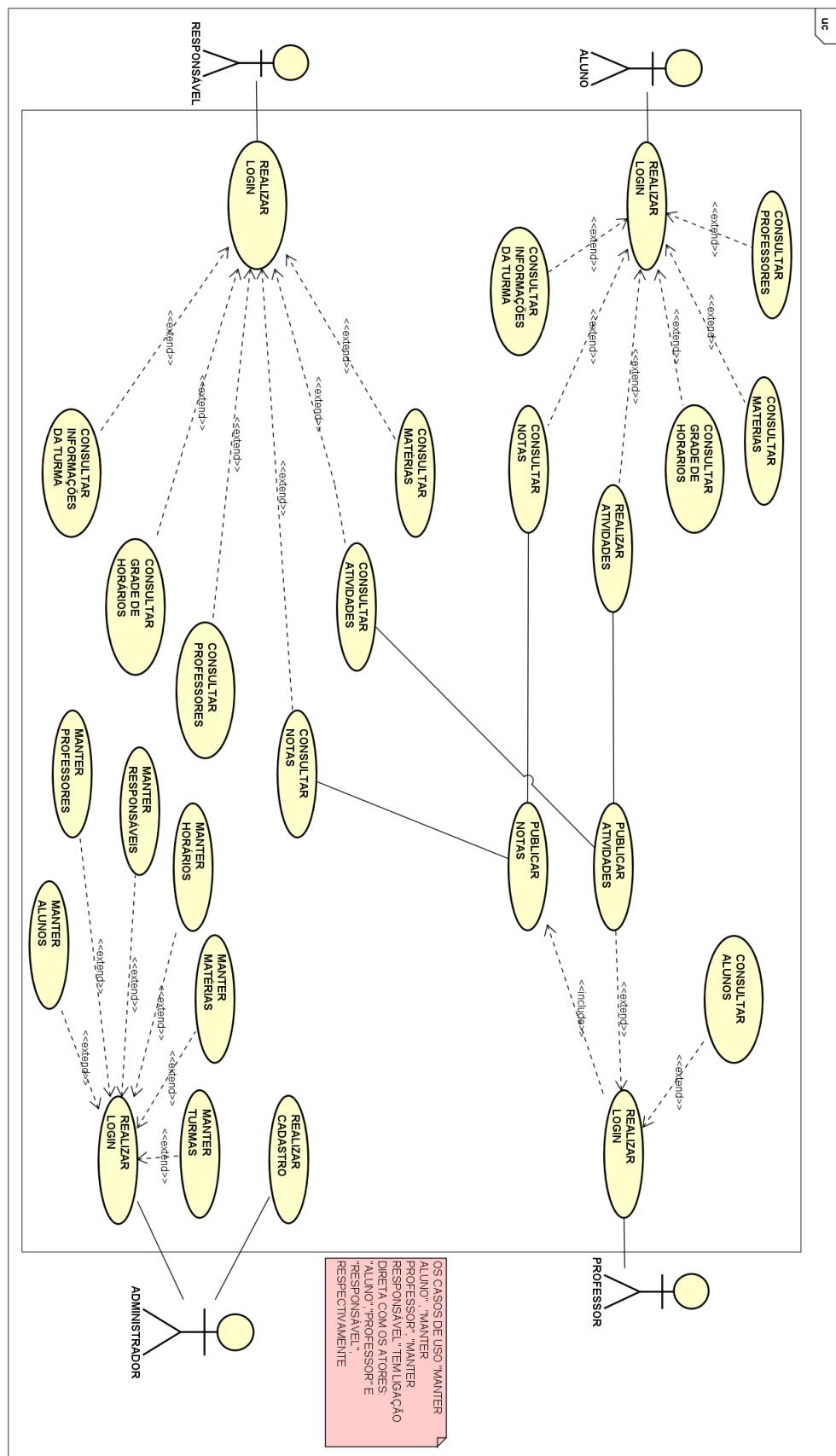


Figura 3: DCU Geral

9. DESKTOP

9.1. DIAGRAMA DE CASO DE USO DESKTOP

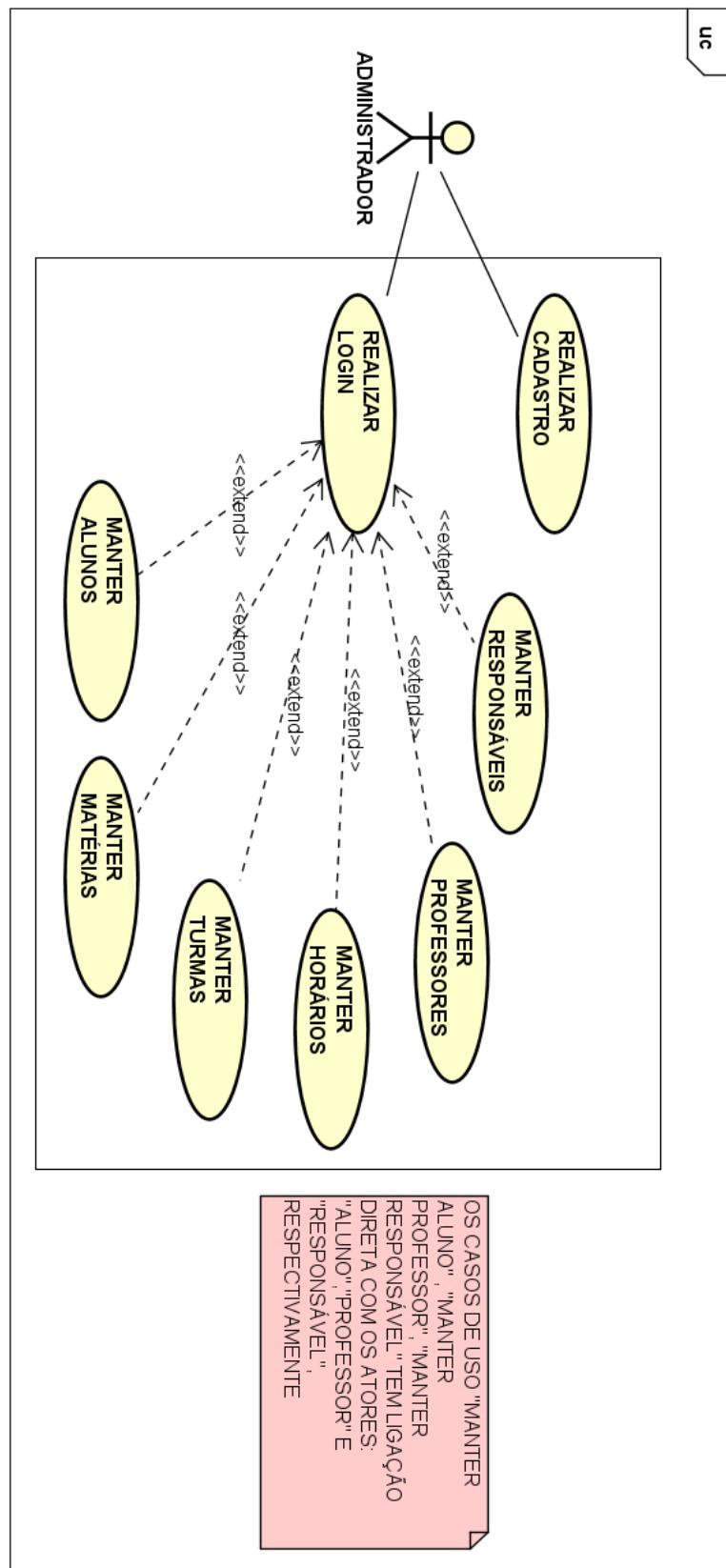


Figura 4: DCU Desktop

Código	Nome
CdU-1	Realizar login
Sumário	
O administrador precisa realizar o login no sistema para acessar todas as funcionalidades administrativas.	
Autor Principal	Autor Secundário
Administrador	Nenhum
Pré-Condições	
Para realizar o Login o Administrador deve estar cadastrado.	

Fluxo Principal	
Atores	Sistema
1. O administrador insere suas credenciais na tela inicial.	2. Se forem validadas com sucesso, ele é direcionado ao painel principal.

Fluxo Alternativo - Usuário Não Encontrado	
Atores	Sistema
1. O administrador insere suas credenciais na tela inicial.	2. Se o usuário não for encontrado, o sistema exibe uma mensagem de erro.

Pós-Condições
Fluxo Principal:
<ul style="list-style-type: none"> O usuário, agora logado como administrador, tem acesso a todas as funcionalidades do sistema.
Fluxo Alternativo:
<ul style="list-style-type: none"> O usuário permanecerá na tela de login e poderá escolher entre tentar novamente, recuperar a senha ou realizar o cadastro no sistema..

Código	Nome
CdU-2	Realizar Cadastro
Sumário	
O administrador pode criar uma conta para si mesmo no sistema.	
Autor Principal	Autor Secundário
Administrador	Responsável
Pré-Condições	
Para realizar o cadastro o administrador deve informar a senha secreta.	

Fluxo Principal	
Atores	Sistema
1. O administrador insere a senha secreta exclusiva para criação de novos administradores. 3. Preenche os dados requisitados e confirma	2. Solicita os dados necessários para a criação de um novo usuário. 4. Grava o novo administrador no banco de dados e o usuário é direcionado ao painel de administração.

Fluxo Alternativo - Dados Incorretos	
Atores	Sistema
1. O administrador insere a senha secreta exclusiva para criação de novos administradores. 3. Preenche os dados requisitados e confirma	2. Solicita os dados necessários para a criação de um novo usuário. 4. Se houver dados incorretos ou faltantes, o sistema solicita correções antes de permitir a criação do registro.

Fluxo Exceção - Senha Secreta Incorreta	
Atores	Sistema
1. O administrador insere a senha secreta exclusiva para criação de novos administradores.	2. Se a senha secreta estiver errada, o sistema não permite o usuário iniciar o processo de cadastro.

Pós-Condições
Fluxo Principal:
<ul style="list-style-type: none"> O usuário, agora logado como administrador, tem acesso a todas as funcionalidades do sistema.
Fluxo Alternativo:
<ul style="list-style-type: none"> O usuário permanecerá na tela de cadastro e poderá escolher entre tentar novamente, modificando ou adicionando novos dados.
Fluxo Exceção:
<ul style="list-style-type: none"> O usuário deverá informar a senha secreta novamente.

Código	Nome
CdU-3	Manter Responsável
Sumário	
O administrador pode realizar as operações de CRUD (Criar, Ler, Atualizar e Deletar) sobre os registros de responsáveis. Isso inclui criar novos registros, visualizar detalhes, atualizar informações e excluir registros de responsáveis.	
Autor Principal	Autor Secundário
Administrador	Responsável
Pré-Condições	
O usuário deve informar qual opção deseja: Criar, Ler Atualizar ou Deletar um responsável.	

Fluxo Principal	
Atores	Sistema
1. O administrador acessa a funcionalidade de gerenciamento de responsáveis. 3. Seleciona a opção desejada: criar, visualizar, atualizar ou excluir registros de responsáveis.	2. Exibe o formulário ou a lista de responsáveis cadastrados. 4. Realiza a operação solicitada (CRUD) e confirma a ação ao administrador.

Fluxo Alternativo - Dados Incorretos	
Atores	Sistema
1. O administrador acessa a funcionalidade de gerenciamento de responsáveis. 3. Seleciona a opção desejada: criar, visualizar, atualizar ou excluir registros de responsáveis.	2. Exibe o formulário ou a lista de responsáveis cadastrados, conforme a operação escolhida. 4. Se houver dados incorretos ou inconsistências, o sistema solicita que o administrador corrija os erros antes de completar a operação.

Fluxo Exceção - Falha Inesperada	
Atores	Sistema
1. O administrador acessa a funcionalidade de gerenciamento de responsáveis.	2. Se ocorrer uma falha inesperada no processo de criação, atualização ou exclusão, o sistema notifica o administrador sobre o erro e sugere tentar novamente.

Pós-Condições	
<ul style="list-style-type: none"> Fluxo Principal: O responsável foi cadastrado, atualizado ou excluído com sucesso, e o administrador é redirecionado para a tela principal do gerenciamento de responsáveis. Fluxo Alternativo: O sistema exibe o formulário com os campos destacados para correção, permitindo que o administrador revise os dados e tente novamente. Fluxo Exceção: O sistema exibe uma mensagem de erro informando o problema, permitindo que o administrador reinicie o processo. 	

Código	Nome
CdU-4	Manter aluno
Sumário	
O administrador pode realizar as operações de CRUD (Criar, Ler, Atualizar e Deletar) sobre os registros de alunos. Isso inclui criar novos registros, visualizar detalhes, atualizar informações e excluir registros de alunos.	
Autor Principal	Autor Secundário
Administrador	Aluno
Pré-Condições	
O usuário deve informar qual opção deseja: Criar, Ler Atualizar ou Deletar um aluno.	

Fluxo Principal	
Atores	Sistema
1. O administrador acessa a funcionalidade de gerenciamento de alunos. 3. Seleciona a opção desejada: criar, visualizar, atualizar ou excluir registros de alunos.	2. Exibe o formulário ou a lista de alunos cadastrados. 4. Realiza a operação solicitada (CRUD) e confirma a ação ao administrador.

Fluxo Alternativo - Dados Incorretos	
Atores	Sistema
1. O administrador acessa a funcionalidade de gerenciamento de alunos. 3. Seleciona a opção desejada: criar, visualizar, atualizar ou excluir registros de alunos.	2. Exibe o formulário ou a lista de alunos cadastrados, conforme a operação escolhida. 4. Se houver dados incorretos ou inconsistências, o sistema solicita que o administrador corrija os erros antes de completar a operação.

Fluxo Exceção - Falha Inesperada	
Atores	Sistema
1. O administrador acessa a funcionalidade de gerenciamento de alunos.	2. Se ocorrer uma falha inesperada no processo de criação, atualização ou exclusão, o sistema notifica o administrador sobre o erro e sugere tentar novamente.

Pós-Condições
<ul style="list-style-type: none"> Fluxo Principal: O aluno foi cadastrado, atualizado ou excluído com sucesso, e o administrador é redirecionado para a tela principal do gerenciamento de alunos. Fluxo Alternativo: O sistema exibe o formulário com os campos destacados para correção, permitindo que o administrador revise os dados e tente novamente. Fluxo Exceção: O sistema exibe uma mensagem de erro informando o problema, permitindo que o administrador reinicie o processo.

Código	Nome
CdU-5	Manter Professores
Sumário	
O administrador pode realizar as operações de CRUD (Criar, Ler, Atualizar e Deletar) sobre os registros de professores. Isso inclui criar novos registros, visualizar detalhes, atualizar informações e excluir registros de professores.	
Autor Principal	Autor Secundário
Administrador	Professor
Pré-Condições	
O usuário deve informar qual opção deseja: Criar, Ler, Atualizar ou Deletar um professor.	

Fluxo Principal	
Atores	Sistema
1. O administrador acessa a funcionalidade de gerenciamento de professores. 3. Seleciona a opção desejada: criar, visualizar, atualizar ou excluir registros de professores.	2. Exibe o formulário ou a lista de professores cadastrados. 4. Realiza a operação solicitada (CRUD) e confirma a ação ao administrador.

Fluxo Alternativo - Dados Incorretos	
Atores	Sistema
1. O administrador acessa a funcionalidade de gerenciamento de professores. 3. Seleciona a opção desejada: criar, visualizar, atualizar ou excluir registros de professores.	2. Exibe o formulário ou a lista de professores cadastrados, conforme a operação escolhida. 4. Se houver dados incorretos ou inconsistências, o sistema solicita que o administrador corrija os erros antes de completar a operação.

Fluxo Exceção - Falha Inesperada	
Atores	Sistema
1. O administrador acessa a funcionalidade de gerenciamento de professores.	2. Se ocorrer uma falha inesperada no processo de criação, atualização ou exclusão, o sistema notifica o administrador sobre o erro e sugere tentar novamente.

Pós-Condições
<ul style="list-style-type: none"> Fluxo Principal: O professor foi cadastrado, atualizado ou excluído com sucesso, e o administrador é redirecionado para a tela principal do gerenciamento de professores. Fluxo Alternativo: O sistema exibe o formulário com os campos destacados para correção, permitindo que o administrador revise os dados e tente novamente. Fluxo Exceção: O sistema exibe uma mensagem de erro informando o problema, permitindo que o administrador reinicie o processo.

Código	Nome
CdU-6	Manter matéria
Sumário	
O administrador pode realizar as operações de CRUD (Criar, Ler, Atualizar e Deletar) sobre os registros de matérias. Isso inclui criar novos registros, visualizar detalhes, atualizar informações e excluir registros de matérias.	
Autor Principal	Autor Secundário
Administrador	
Pré-Condições	
O usuário deve informar qual opção deseja: Criar, Ler Atualizar ou Deletar uma matéria.	

Fluxo Principal	
Atores	Sistema
1. O administrador acessa a funcionalidade de gerenciamento de matérias. 3. Seleciona a opção desejada: criar, visualizar, atualizar ou excluir registros de matérias.	2. Exibe o formulário ou a lista de matérias cadastrados. 4. Realiza a operação solicitada (CRUD) e confirma a ação ao administrador.

Fluxo Alternativo - Dados Incorretos	
Atores	Sistema
1. O administrador acessa a funcionalidade de gerenciamento de matérias. 3. Seleciona a opção desejada: criar, visualizar, atualizar ou excluir registros de matérias.	2. Exibe o formulário ou a lista de matérias cadastradas, conforme a operação escolhida. 4. Se houver dados incorretos ou inconsistências, o sistema solicita que o administrador corrija os erros antes de completar a operação.

Fluxo Exceção - Falha Inesperada	
Atores	Sistema
1. O administrador acessa a funcionalidade de gerenciamento de matérias.	2. Se ocorrer uma falha inesperada no processo de criação, atualização ou exclusão, o sistema notifica o administrador sobre o erro e sugere tentar novamente.

Pós-Condições	
<ul style="list-style-type: none"> Fluxo Principal: A matéria foi cadastrada, atualizado ou excluída com sucesso, e o administrador é redirecionado para a tela principal do gerenciamento de matérias. Fluxo Alternativo: O sistema exibe o formulário com os campos destacados para correção, permitindo que o administrador revise os dados e tente novamente. Fluxo Exceção: O sistema exibe uma mensagem de erro informando o problema, permitindo que o administrador reinicie o processo. 	

Código	Nome
CdU-7	Manter turma
Sumário	
O administrador pode realizar as operações de CRUD (Criar, Ler, Atualizar e Deletar) sobre os registros de turmas. Isso inclui criar novos registros, visualizar detalhes, atualizar informações e excluir registros de turmas.	
Autor Principal	Autor Secundário
Administrador	Nenhum
Pré-Condições	
O usuário deve informar qual opção deseja: Criar, Ler Atualizar ou Deletar uma turma.	

Fluxo Principal	
Atores	Sistema
1. O administrador acessa a funcionalidade de gerenciamento de turmas. 3. Seleciona a opção desejada: criar, visualizar, atualizar ou excluir registros de turmas.	2. Exibe o formulário ou a lista de turmas cadastrados. 4. Realiza a operação solicitada (CRUD) e confirma a ação ao administrador.

Fluxo Alternativo - Dados Incorretos	
Atores	Sistema
1. O administrador acessa a funcionalidade de gerenciamento de turmas. 3. Seleciona a opção desejada: criar, visualizar, atualizar ou excluir registros de turmas.	2. Exibe o formulário ou a lista de turmas cadastradas, conforme a operação escolhida. 4. Se houver dados incorretos ou inconsistências, o sistema solicita que o administrador corrija os erros antes de completar a operação.

Fluxo Exceção - Falha Inesperada	
Atores	Sistema
1. O administrador acessa a funcionalidade de gerenciamento de turmas.	2. Se ocorrer uma falha inesperada no processo de criação, atualização ou exclusão, o sistema notifica o administrador sobre o erro e sugere tentar novamente.

Pós-Condições
<ul style="list-style-type: none"> Fluxo Principal: A turma foi cadastrada, atualizado ou excluída com sucesso, e o administrador é redirecionado para a tela principal do gerenciamento de turmas. Fluxo Alternativo: O sistema exibe o formulário com os campos destacados para correção, permitindo que o administrador revise os dados e tente novamente. Fluxo Exceção: O sistema exibe uma mensagem de erro informando o problema, permitindo que o administrador reinicie o processo.

Código	Nome
CdU-8	Manter horários
Sumário	
O administrador pode realizar as operações de CRUD (Criar, Ler, Atualizar e Deletar) sobre os registros de horários. Isso inclui criar novos horários, visualizar detalhes, atualizar informações e excluir registros de turmas	
Autor Principal	Autor Secundário
Administrador	Nenhum
Pré-Condições	
O usuário deve informar qual opção deseja: Criar, Ler Atualizar ou Deletar um horário.	

Fluxo Principal	
Atores	Sistema
1. O administrador acessa a funcionalidade de gerenciamento de horários. 3. Seleciona a opção desejada: criar, visualizar, atualizar ou excluir registros de horários.	2. Exibe o formulário ou a lista de horários cadastrados. 4. Realiza a operação solicitada (CRUD) e confirma a ação ao administrador.

Fluxo Alternativo - Dados Incorretos	
Atores	Sistema
1. O administrador acessa a funcionalidade de gerenciamento de horários. 3. Seleciona a opção desejada: criar, visualizar, atualizar ou excluir registros de horários.	2. Exibe o formulário ou a lista de horários cadastrados, conforme a operação escolhida. 4. Se houver dados incorretos ou inconsistências, o sistema solicita que o administrador corrija os erros antes de completar a operação.

Fluxo Exceção - Falha Inesperada	
Atores	Sistema
1. O administrador acessa a funcionalidade de gerenciamento de horários.	2. Se ocorrer uma falha inesperada no processo de criação, atualização ou exclusão, o sistema notifica o administrador sobre o erro e sugere tentar novamente.

Pós-Condições	
<ul style="list-style-type: none"> Fluxo Principal: O horário foi cadastrado, atualizado ou excluído com sucesso, e o administrador é redirecionado para a tela principal do gerenciamento de horários. Fluxo Alternativo: O sistema exibe o formulário com os campos destacados para correção, permitindo que o administrador revise os dados e tente novamente. Fluxo Exceção: O sistema exibe uma mensagem de erro informando o problema, permitindo que o administrador reinicie o processo. 	

9.2. CAPTURAS DE TELA DESKTOP

Segue um excerto das páginas presentes no ambiente web, capturadas com o usuário "Administrador" logado no sistema.

Login

eduSYST

Administrador

CPF: _____
E-mail: _____
Senha: _____
Nascimento: / /
Endereço: _____
Telefone: _____
Gênero: Masculino
Responsável: Maria da Silva
Esqueceu a senha? Entrar Cadastrar

Insera o Nome ou CPF do aluno: _____
PESSOAR

Cadastro

eduSYST

CADASTRAR ALUNO

CPF: _____
Nome: _____
E-mail: _____
Senha: _____
Nascimento: / /
Endereço: _____
Telefone: _____
Gênero: Feminino
Responsável: _____
LIMPAR

Alterar

eduSYST

ALTERAR ALUNO

CPF: 901-001-0001-0001
Nome: Pedro Silva
E-mail: pedro.silva@escola.com
ID_Turma: 1901
Classe: Ano 2024
Educação: Ensino Fundamental
Nome: Ana Souza
E-mail: ana.souza@escola.com
ID_Turma: 1901
Classe: Ano 2024
Educação: Ensino Fundamental
Nome: Felipe Lima
E-mail: felipe.lima@escola.com
ID_Turma: 1901
Classe: Ano 2024
Educação: Ensino Fundamental
Nome: João Pereira
E-mail: joao.pereira@escola.com
ID_Turma: 1901
Classe: Ano 2024
Educação: Ensino Fundamental
Nome: Isabela Santos
E-mail: isabela.santos@escola.com
ID_Turma: 1901
Classe: Ano 2024
Educação: Ensino Fundamental

LIMPAR

Consultar

eduSYST

CADASTRAR ALUNO

CPF: _____
Nome: _____
E-mail: _____
ID_Turma: _____
Classe: _____
Ano: _____
Educação: _____
LIMPAR

Figura 5: Telas - Login, Cadastro e Alteração de Alunos e Consulta de Turmas

edusYST

Consultar

Alunos						
ID	Nome	E-mail	Data_Nasc	Endereço	Telefone	Genero
1	Petra Silva	pedro.silva@escola.com	2010-03-10	Rua Secundino, 456	1192345678	Masculino
2	Ariane Souza	ari.souza@escola.com	2011-07-19	Rua Fernano, 789	11923456789	Feminino
3	Felipe Lima	felipe.mir@escola.com	2010-03-14	Rua Vagner Morato, 89	119234567890	Masculino
4	Jéssica Ferreira	jessica.fern@escola.com	2011-08-15	Sala Júlio, 900	119234567891	Masculino
5	Isabela Sales	isabela.sales@escola.com	2010-09-12	Sala 100, 707	119234567892	Masculino
6	Bárbara Góes	bárbara.goes@escola.com	2010-09-14	Rua Santo Antônio, 90	119234567893	Feminino
7	Luís Henrique	luis.henrique@escola.com	2010-05-20	Rua Santo Antônio, 507	119234567894	Masculino
8	Lucas Ferreira	lucas.fern@escola.com	2010-07-14	Rua São Pedro, 100	119234567895	Masculino
9	Mariana Souza	mariana.souza@escola.com	2010-10-14	Rua Antônio, 102	119234567896	Masculino
10	Ana Costa	ana.costa@escola.com	2011-11-18	Rua Gaspar Dutra, 102	119234567897	Masculino

edusYST

Enturmar

Insira o nome ou CPF do aluno a ser enturmado:	<input type="text"/>	<input type="button" value="Pesquisar"/>
Nome do aluno	Rafael Almeida	
ID do Aluno	6	
Seleção uma turma:	<input type="button" value="Enturmar Aluno"/> <input type="button" value="Desenturmar Aluno"/>	
ID Turma	2	Classe do Aluno
Classe	101	

Desenturmar

Insira o CPF ou nome do aluno a ser desenturmado:	<input type="text"/>	<input type="button" value="Pesquisar"/>
ID Turma	2	Classe
Rafael Almeida	<input type="button" value="Desenturmar Aluno"/>	
ID Turma	2	Classe
Classe	101	

Figura 6: Telas - Enturmação, Desenturmação, Consulta e Exclusão de Alunos

10. WEB

10.1. DIAGRAMA DE CASO DE USO WEB

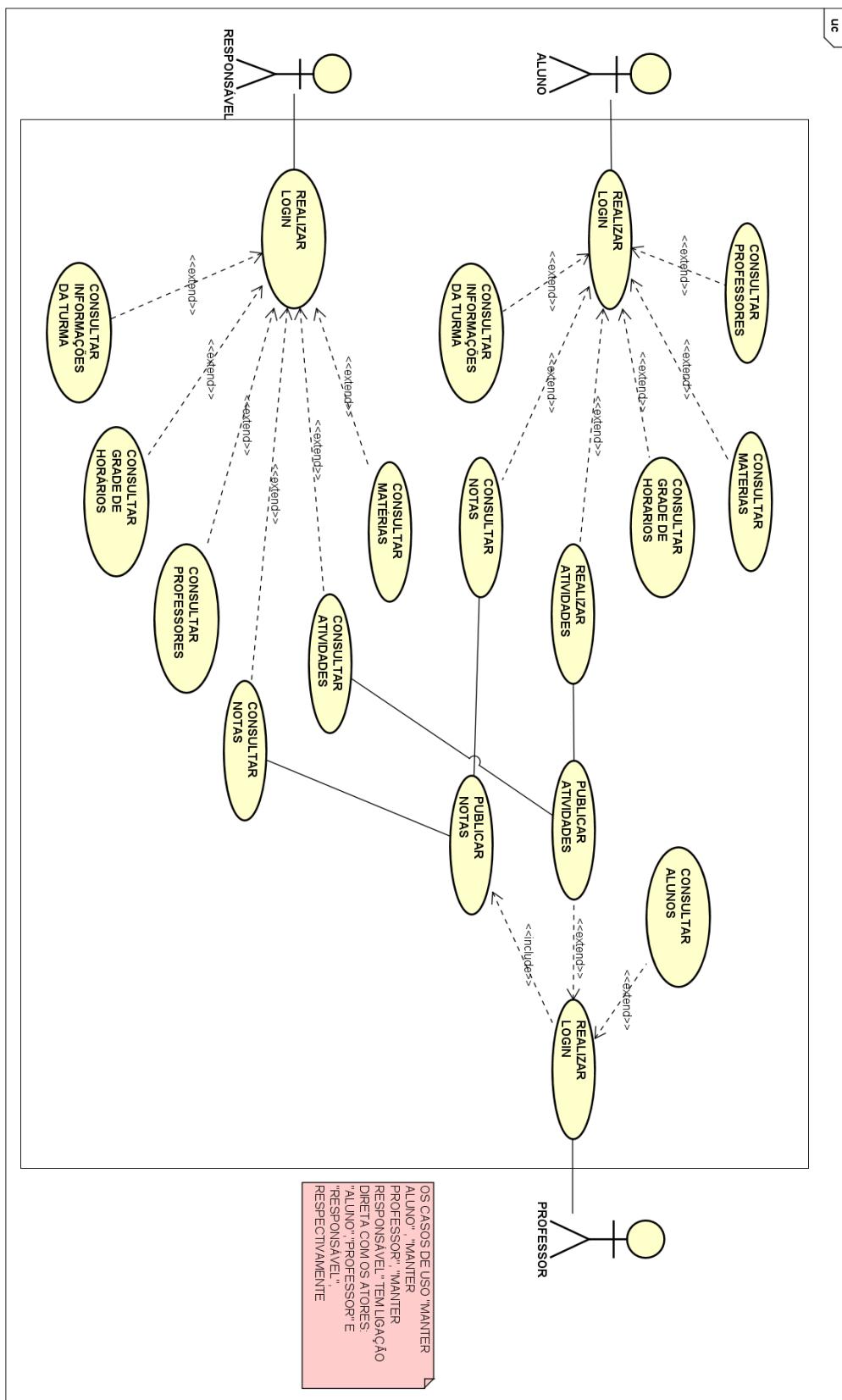


Figura 7: DCU WEB

Código	Nome
CdU-9	Realizar Login
Sumário	
O Aluno precisa realizar o login no sistema para acessar todas as funcionalidades.	
Autor Principal	Autor Secundário
Aluno	Responsável
Pré-Condições	
Para realizar o Login o aluno deve estar cadastrado e deve estar associado com um responsável. Este é o primeiro passo que o aluno realiza para acessar o sistema. Sem o login, o aluno não poderá acessar outras funcionalidades	

Fluxo Principal	
Atores	Sistema
1. O aluno insere suas credenciais na tela de login.	2. Se forem validadas com sucesso, ele é direcionado ao painel principal.

Fluxo Alternativo - Usuário Não Encontrado	
Atores	Sistema
1. O aluno insere suas credenciais na tela inicial.	2. Se o usuário não for encontrado, o sistema exibe uma mensagem de erro.

Pós-Condições
Fluxo Principal:
<ul style="list-style-type: none"> • O usuário, agora logado como aluno, tem acesso a todas as funcionalidades do sistema.

Fluxo Alternativo:

- O usuário permanecerá na tela de login e poderá tentar novamente.

Código	Nome
CdU-10	Consultar Professores
Sumário	
Permite ao aluno visualizar a lista de professores responsáveis pelas disciplinas disponíveis pela escola	
Autor Principal	Autor Secundário
Aluno	Professor
Pré-Condições	
Aluno precisa estar logado no sistema.	

Fluxo Principal	
Atores	Sistema
1. O aluno seleciona a opção para consultar professores no painel principal	2. Exibe uma lista com os nomes e informações dos professores.

Fluxo Alternativo – Informação Não Encontrada	
Atores	Sistema
1. O aluno seleciona a opção para consultar professores no painel principal	2. Se não houver professores cadastrados, o sistema exibe uma mensagem indicando que não há informações disponíveis no momento.

Pós-Condições
Fluxo Principal:
<ul style="list-style-type: none"> O aluno visualizou as informações dos professores cadastrados no sistema.
Fluxo Alternativo:
<ul style="list-style-type: none"> O sistema apresentou uma mensagem indicando a ausência de professores cadastrados, garantindo que o aluno foi informado da situação atual.

Código	Nome
CdU-11	Consultar matérias
Sumário	
Permite ao aluno visualizar a lista de matérias disponíveis pela escola	
Autor Principal	Autor Secundário
Aluno	Nenhum
Pré-Condições	
Aluno precisa estar logado no sistema.	

Fluxo Principal	
Atores	Sistema
1. O aluno seleciona a opção para consultar matérias no painel principal	2. Exibe uma lista com os nomes e informações das matérias.

Fluxo Alternativo – Informação Não Encontrada	
Atores	Sistema
1. O aluno seleciona a opção para consultar matérias no painel principal	2. Se não houver matérias cadastradas, o sistema exibe uma mensagem indicando que não há informações disponíveis no momento.

Pós-Condições	
Fluxo Principal:	<ul style="list-style-type: none"> O aluno visualizou as informações das matérias cadastradas no sistema.
Fluxo Alternativo:	<ul style="list-style-type: none"> O sistema apresentou uma mensagem indicando a ausência de matérias cadastradas, garantindo que o aluno foi informado da situação atual.

Código	Nome
CdU-12	Consultar grade de horários
Sumário	
Permite ao aluno visualizar a grade de horários disponível na escola	
Autor Principal	Autor Secundário
Aluno	Nenhum
Pré-Condições	
Aluno precisa estar logado no sistema.	

Fluxo Principal	
Atores	Sistema
1. O aluno seleciona a opção para consultar grade de horários no painel principal	2. Exibe uma lista com informações e conteúdo da grade de horários.

Fluxo Alternativo – Informação Não Encontrada	
Atores	Sistema
1. O aluno seleciona a opção para consultar grade de horários no painel principal	2. Se não houver grade de horários cadastradas, o sistema exibe uma mensagem indicando que não há informações disponíveis no momento.

Pós-Condições
Fluxo Principal:
<ul style="list-style-type: none"> O aluno visualizou as informações das grade de horários cadastradas no sistema.
Fluxo Alternativo:
<ul style="list-style-type: none"> O sistema apresentou uma mensagem indicando a ausência de grade de horários cadastradas, garantindo que o aluno foi informado da situação atual.

Código	Nome
CdU-13	Consultar atividades
Sumário	
Permite ao aluno visualizar a lista de atividades disponíveis pela escola	
Autor Principal	Autor Secundário
Aluno	Nenhum
Pré-Condições	
Aluno precisa estar logado no sistema.	

Fluxo Principal	
Atores	Sistema
1. O aluno seleciona a opção para consultar atividades no painel principal 3. O aluno seleciona a atividade que deseja realizar, preenche as respostas e submete a atividade.	2. Exibe uma lista com os nomes e informações das atividades. 4. Atividade é confirmada e computada.

Fluxo Alternativo – Informação Não Encontrada	
Atores	Sistema
1. O aluno seleciona a opção para consultar atividades no painel principal	2. Se não houver atividades cadastradas, o sistema exibe uma mensagem indicando que não há informações disponíveis no momento.

Pós-Condições
Fluxo Principal:
<ul style="list-style-type: none"> O aluno visualizou as informações das atividades cadastradas no sistema.
Fluxo Alternativo:
<ul style="list-style-type: none"> O sistema apresentou uma mensagem indicando a ausência de atividades cadastradas, garantindo que o aluno foi informado da situação atual.

Código	Nome
CdU-14	Consultar notas
Sumário	
Permite ao aluno visualizar a lista de notas disponíveis pela escola	
Autor Principal	Autor Secundário
Aluno	Nenhum
Pré-Condições	
Aluno precisa estar logado no sistema.	

Fluxo Principal	
Atores	Sistema
1. O aluno seleciona a opção para consultar notas no painel principal.	2. Exibe uma lista com os nomes e informações das notas e gera uma média anual com base nas notas do aluno em cada matéria. Se essa média for maior que 5,6, o aluno está aprovado. Se não, ele estará reprovado na determinada matéria.

Fluxo Alternativo 1 – Informação Não Encontrada	
Atores	Sistema
1. O aluno seleciona a opção para consultar notas no painel principal	2. Se não houver notas cadastradas, o sistema exibe uma mensagem indicando que não há informações disponíveis no momento.

Fluxo Alternativo 2 – Nota Cão Computada	
Atores	Sistema
1. O aluno seleciona a opção para consultar notas no painel principal.	2. Exibe uma lista com os nomes e informações das notas. Se não possuir todas as notas dos quatro bimestres, ele não computará a média final, e dirá que a situação está indisponível.

Pós-Condições	
Fluxo Principal:	<ul style="list-style-type: none"> O aluno visualizou as informações das notas cadastradas no sistema.
Fluxo Alternativo 1:	<ul style="list-style-type: none"> O sistema apresentou uma mensagem indicando a ausência de notas cadastradas, garantindo que o aluno foi informado da situação atual.
Fluxo Alternativo 2:	<ul style="list-style-type: none"> O aluno visualizou as informações das notas cadastradas no sistema. O sistema apresentou uma mensagem indicando indisponibilidade da média, garantindo que o aluno foi informado da situação atual.

Código	Nome
CdU-15	Consultar informações da turma
Sumário	
Permite ao aluno visualizar as informações da turma que o aluno faz parte.	
Autor Principal	Autor Secundário
Aluno	Nenhum
Pré-Condições	
Aluno precisa estar logado no sistema.	

Fluxo Principal	
Atores	Sistema
1. O aluno seleciona a opção para consultar informações da turma no painel principal	2. Exibe uma lista com informações e conteúdo da turma.

Fluxo Alternativo – Informação Não Encontrada	
Atores	Sistema
1. O aluno seleciona a opção para consultar informações da turma no painel principal	2. Se o aluno não estiver associado com nenhuma turma, o sistema exibe uma mensagem indicando que não há informações disponíveis no momento.

Pós-Condições
Fluxo Principal:
<ul style="list-style-type: none"> O aluno visualizou as informações das informações da turma cadastradas no sistema.
Fluxo Alternativo:
<ul style="list-style-type: none"> O sistema apresentou uma mensagem indicando a ausência de informações da turma, garantindo que o aluno foi informado da situação atual.

Os casos de uso "realizar login", "consultar professores", "consultar matérias", "consultar grade de horários", "consultar notas" e "consultar informações da turma" do ator Responsável possuem funcionamento idêntico aos casos de uso CdU-9, CdU-10, CdU-11, CdU-12, CdU-14 e CdU-15, do ator Aluno, respectivamente.

Código	Nome
CdU-16	Consultar atividades
Sumário	
Permite ao responsável visualizar as atividades que o aluno associado faz parte.	
Autor Principal	Autor Secundário
Responsável	Aluno
Pré-Condições	
Responsável precisa estar logado no sistema.	

Fluxo Principal	
Atores	Sistema
1. O Responsável seleciona a opção para consultar atividades no painel principal	2. Exibe uma lista com informações e conteúdo da turma.

Fluxo Alternativo – Informação Não Encontrada	
Atores	Sistema
1. O Responsável seleciona a opção para consultar atividades no painel principal	2. Se o aluno do responsável não possuir atividades, o sistema exibe uma mensagem indicando que não há informações disponíveis no momento.

Pós-Condições
Fluxo Principal:
<ul style="list-style-type: none"> O responsável visualizou as informações das atividades cadastradas no sistema.
Fluxo Alternativo:
<ul style="list-style-type: none"> O sistema apresentou uma mensagem indicando a ausência de atividades, garantindo que o responsável foi informado da situação atual.

Código	Nome
CdU-17	Realizar Login
Sumário	
O professor precisa realizar o login no sistema para acessar todas as funcionalidades.	
Autor Principal	Autor Secundário
Aluno	Nenhum
Pré-Condições	
Para realizar o Login o professor deve estar cadastrado. Este é o primeiro passo que o professor realiza para acessar o sistema. Sem o login, o aluno não poderá acessar outras funcionalidades	

Fluxo Principal	
Atores	Sistema
1. O professor insere suas credenciais na tela de login.	2. Se forem validadas com sucesso, ele é direcionado ao painel principal.

Fluxo Alternativo - Usuário Não Encontrado	
Atores	Sistema
1. O professor insere suas credenciais na tela inicial.	2. Se o usuário não for encontrado, o sistema exibe uma mensagem de erro.

Pós-Condições
Fluxo Principal:
<ul style="list-style-type: none"> O usuário, agora logado como professor, tem acesso a todas as funcionalidades do sistema.

Fluxo Alternativo:

- O usuário permanecerá na tela de login e poderá tentar novamente.

Código	Nome
CdU-18	Publicar Atividades
Sumário	
Permite ao professor criar e publicar atividades para os alunos, como exercícios que serão realizados online.	
Autor Principal	Autor Secundário
Aluno	Nenhum
Pré-Condições	
O professor precisa estar logado no sistema.	

Fluxo Principal	
Atores	Sistema
1. O professor seleciona a opção para publicar atividades no painel principal 3. Após preencher os dados, ele confirma a atividade.	2. Solicita o título, descrição, prazo e conteúdo. 4. Se os dados estiverem preenchidos corretamente, a atividade é publicada e se torna acessível aos alunos.

Fluxo Alternativo – Informações Incorretas ou Faltantes	
Atores	Sistema
1. O professor seleciona a opção para publicar atividades no painel principal 3. Após preencher os dados, ele confirma a atividade.	2. Solicita o título, descrição, prazo e conteúdo. 4. Se os dados estiverem preenchidos incorretamente ou informações cruciais estiverem faltando, o professor tem a oportunidade de tentar novamente.

Pós-Condições
Fluxo Principal:
<ul style="list-style-type: none"> • Atividade Publicada: A atividade está visível e acessível aos alunos no sistema. • Registro Atualizado: O sistema atualizou os registros com os detalhes da nova atividade.
Fluxo Alternativo:
<ul style="list-style-type: none"> • Dados Não Salvos: A atividade não foi publicada, pois os dados estavam incorretos ou incompletos. • Orientação Fornecida: O sistema forneceu uma mensagem para o professor corrigir as informações e tentar novamente.

Código	Nome				
CdU-19	Publicar notas				
Sumário					
Permite ao professor criar e publicar notas para os alunos, como exercícios que serão realizados online.					
<table border="1"> <tr> <td>Autor Principal</td> <td>Autor Secundário</td> </tr> <tr> <td>Aluno</td> <td>Aluno</td> </tr> </table>		Autor Principal	Autor Secundário	Aluno	Aluno
Autor Principal	Autor Secundário				
Aluno	Aluno				
Pré-Condições					
O professor precisa estar logado no sistema.					

Fluxo Principal	
Atores	Sistema
1. O professor seleciona a opção para publicar notas no painel principal 2. O professor seleciona o aluno e a matéria. 5. Informa a nota e confirma.	2. Solicita a turma, o aluno e a matéria específicas. 4. Solicita qual bimestre e a nota, sendo ela obrigatoriamente igual ou maior a 0 e menor que 10. 6. Se estiver condizente, a nota é computada.

Fluxo Alternativo – Informações Incorretas ou Faltantes	
Atores	Sistema
1. O professor seleciona a opção para publicar notas no painel principal 2. O professor seleciona o aluno e a matéria. 5. O professor insere uma nota que não atende ao critério (fora do intervalo permitido).	2. Solicita a turma, o aluno e a matéria específicas. 4. Solicita qual bimestre e a nota, sendo ela obrigatoriamente igual ou maior a 0 e menor que 10. 6. O sistema exibe uma mensagem de erro informando que a nota deve estar entre 0 e 10 e solicita correção.

Pós-Condições
Fluxo Principal: <ul style="list-style-type: none"> • Nota Publicada: A nota está visível e acessível aos alunos no sistema. • Registro Atualizado: O sistema atualizou os registros com os detalhes da nova nota.
Fluxo Alternativo: <ul style="list-style-type: none"> • Dados Não Salvos: A nota não foi publicada, pois os dados estavam incorretos ou incompletos. • Orientação Fornecida: O sistema forneceu uma mensagem para o professor corrigir as informações e tentar novamente.

Código	Nome
CdU-20	Consultar alunos
Sumário	
Permite ao professor visualizar a lista de alunos responsáveis pelas disciplinas disponíveis pela escola	
Autor Principal	Autor Secundário
Professor	Aluno
Pré-Condições	
Professor precisa estar logado no sistema.	

Fluxo Principal	
Atores	Sistema
1. O professor seleciona a opção para consultar alunos no painel principal	2. Exibe uma lista com os nomes e informações dos alunos.

Fluxo Alternativo – Informação Não Encontrada	
Atores	Sistema
1. O professor seleciona a opção para consultar alunos no painel principal	2. Se não houver alunos cadastrados, o sistema exibe uma mensagem indicando que não há informações disponíveis no momento.

Pós-Condições	
Fluxo Principal:	<ul style="list-style-type: none"> O professor visualizou as informações dos alunos cadastrados no sistema.
Fluxo Alternativo:	<ul style="list-style-type: none"> O sistema apresentou uma mensagem indicando a ausência de alunos cadastrados, garantindo que o professor foi informado da situação atual.

10.2. CAPTURAS DE TELA WEB

Segue um excerto das páginas presentes no ambiente web, capturadas com o usuário “Professor” logado no sistema.

Screenshot 1: Página inicial do professor

O topo da tela exibe o logo "eduSyst" e uma barra de navegação com links para "Home", "Sobre Nós", "Fale Conosco" e "Área Institucional". O nome do usuário logado é "Prof. Alberto". A seção principal "Seja bem-vindo ao ambiente do professor!" contém uma foto de perfil, uma barra de navegação com "Configurações" e "Sair da Sessão", e uma lista de links para "Meus Alunos", "Minhas Turmas", "Chat Escolar", "Meu Horários", "Aplicar Notas" e "Aplicar Atividades".

Screenshot 2: Minhas Turmas

Aqui, o professor pode visualizar suas turmas. A tabela mostra os seguintes dados:

Ano	Turma	Materia
2024	901	Matemática
2024	901	Português
2024	1101	Física
2024	1101	Sociologia

Screenshot 3: Meus Alunos

Aqui, o professor pode gerenciar seus alunos. A tabela mostra os seguintes dados:

Nome	Email	Responsável	Email	Turma	Ano
Ana Souza	ana.souza@escola.com	Carla Souza	carla.souza@domino.com	901	2024
Flávio Lima	flavio.lima@escola.com	Marcelo Jorge	marcelo.jorge@domino.com	901	2024
Isabela Santos	isabela.santos@escola.com	Leandro Ferreira	leandro.fernandes@domino.com	901	2024
José Pereira	jose.pereira@escola.com	Ricardo Matheus	ricardo.matheus@domino.com	901	2024
Fábio Souza	fabio.souza@escola.com	Maria da Silva	maria.silva@domino.com	901	2024
Ana Costa	ana.costa@escola.com	Maria Oliveira	maria.oliveira@domino.com	1101	2024
Bethânia Faria	bethania.faria@escola.com	Jorge Filho	jorge.filho@domino.com	1101	2024
Luis Ferreira	luis.ferreira@escola.com	Carlo Pereira	carlo.pereira@domino.com	1101	2024
Mariana Souza	mariana.souza@escola.com	Patrícia Santos	patricia.santos@domino.com	1101	2024
Rafael Almeida	rafael.almeida@escola.com	Patrícia Rocha	patricia.rocha@domino.com	1101	2024

Screenshot 4: Configurações

Aqui, o professor pode atualizar seu perfil. A seção "Informações Pessoais" permite alterar dados como nome, e-mail, data de nascimento, endereço ou outras informações pessoais. A seção "Atualizar Foto de Perfil" permite carregar uma nova foto de perfil para uso no sistema. A barra de navegação inclui "Configurações" e "Sair da Sessão".

Figura 8: Páginas - Página de usuário, Alunos, Turmas e Configurações do Perfil



EduSyst

Casarão, Ferreira, Henrique e Matheus são os fundadores da EduSyst.

[Voltar ao topo](#)



EduSyst

Todos os direitos reservados.

[Voltar ao topo](#)



EduSyst

Todos os Professores

[Voltar ao ambiente do professor](#)

Name	Email
Prof. Alberto	alberto.prof@ecoclick.com
Prof. Afonso	afonso.prof@ecoclick.com
Prof. Ana Paula	anapaula.prof@ecoclick.com
Prof. Caio Oliveira	caiooliveira.prof@ecoclick.com
Prof. Fernanda	fernandap.prof@ecoclick.com
Prof. Júlio Lima	julio.lima.prof@ecoclick.com
Prof. José	jose.prof@ecoclick.com
Prof. Marcos Oliveira	marcosoliveira.prof@ecoclick.com
Prof. Maria Santos	mariasantos.prof@ecoclick.com
Prof. Roberta Alves	roberta.alves.prof@ecoclick.com

[Recado*](#)

Aceito os termos de uso

[Vai](#)



EduSyst

Meus Horários

[Voltar ao ambiente do professor](#)



EduSyst

Notas Lançadas:

[Voltar ao topo](#)



EduSyst

Aplicar Notas

[Voltar ao ambiente do professor](#)

Figura 9: Páginas - Grade de Horários, Aplicação de Notas, Listagem de Professores e Chat

11. ANDROID

11.1. DIAGRAMA DE CASO DE USO ANDROID

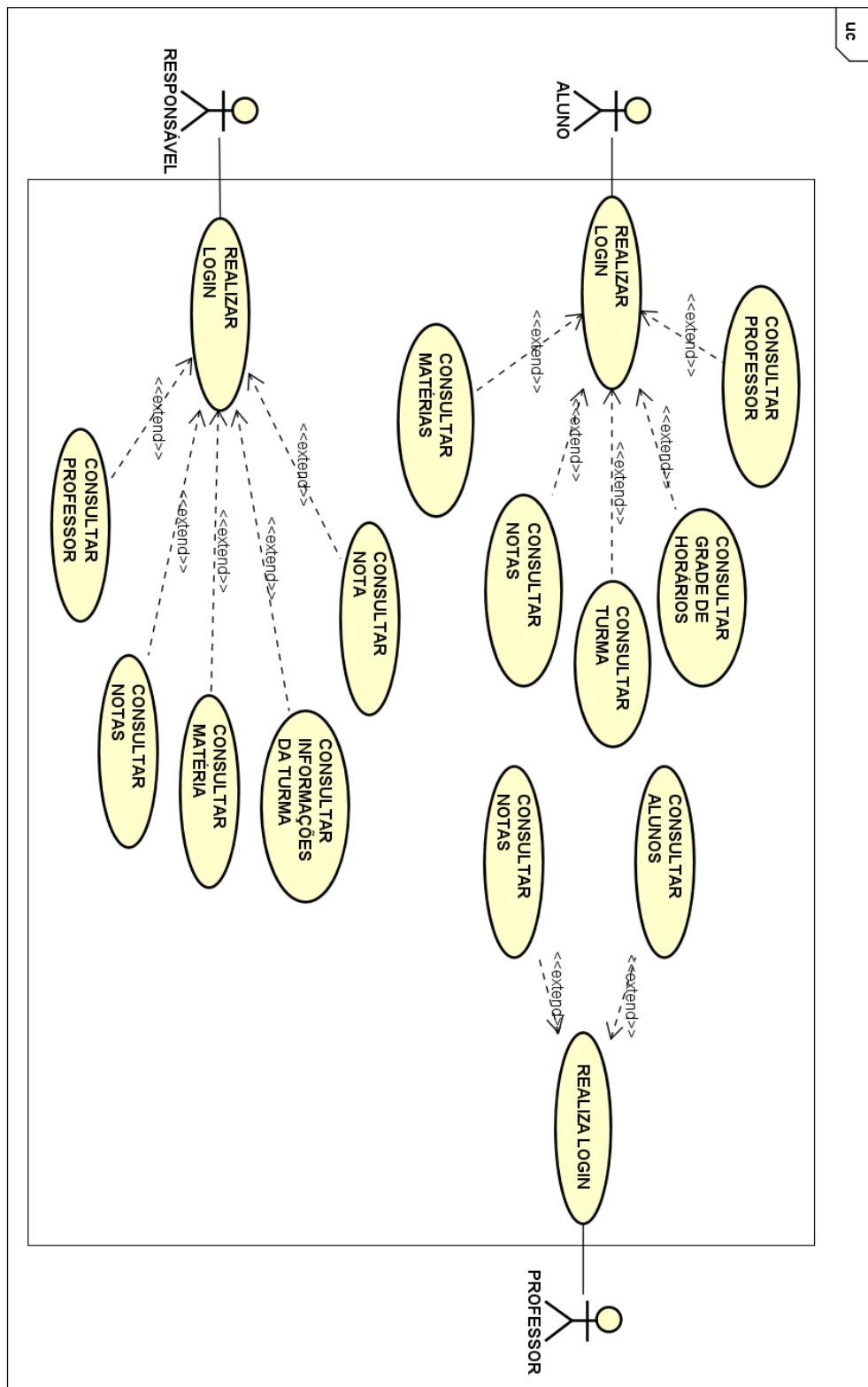


Figura 10: DCU Android

11.2. DESCRIÇÃO DO SISTEMA “EDUSYST” ANDROID

A funcionalidade do sistema Android é idêntica a funcionalidade do sistema WEB, com exceção dos seguintes casos de uso indisponíveis no sistema Android:

- Ator: Aluno
 - 1. Não é possível realizar atividades. (CdU-9 do ator Aluno)
- Ator: Responsável
 - 1. Não é possível consultar atividades do aluno associado ao responsável. (CdU-16 do ator Responsável)
- Ator: Professor
 - 1. Não é possível publicar atividades. (CdU-18 do ator Professor)
 - 2. Não é possível publicar notas. (CdU-19 do ator Professor)

11.3. CAPTURAS DE TELA ANDROID

Segue um excerto das telas presentes no aplicativo Android, capturadas com o usuário "Aluno" logado no sistema.

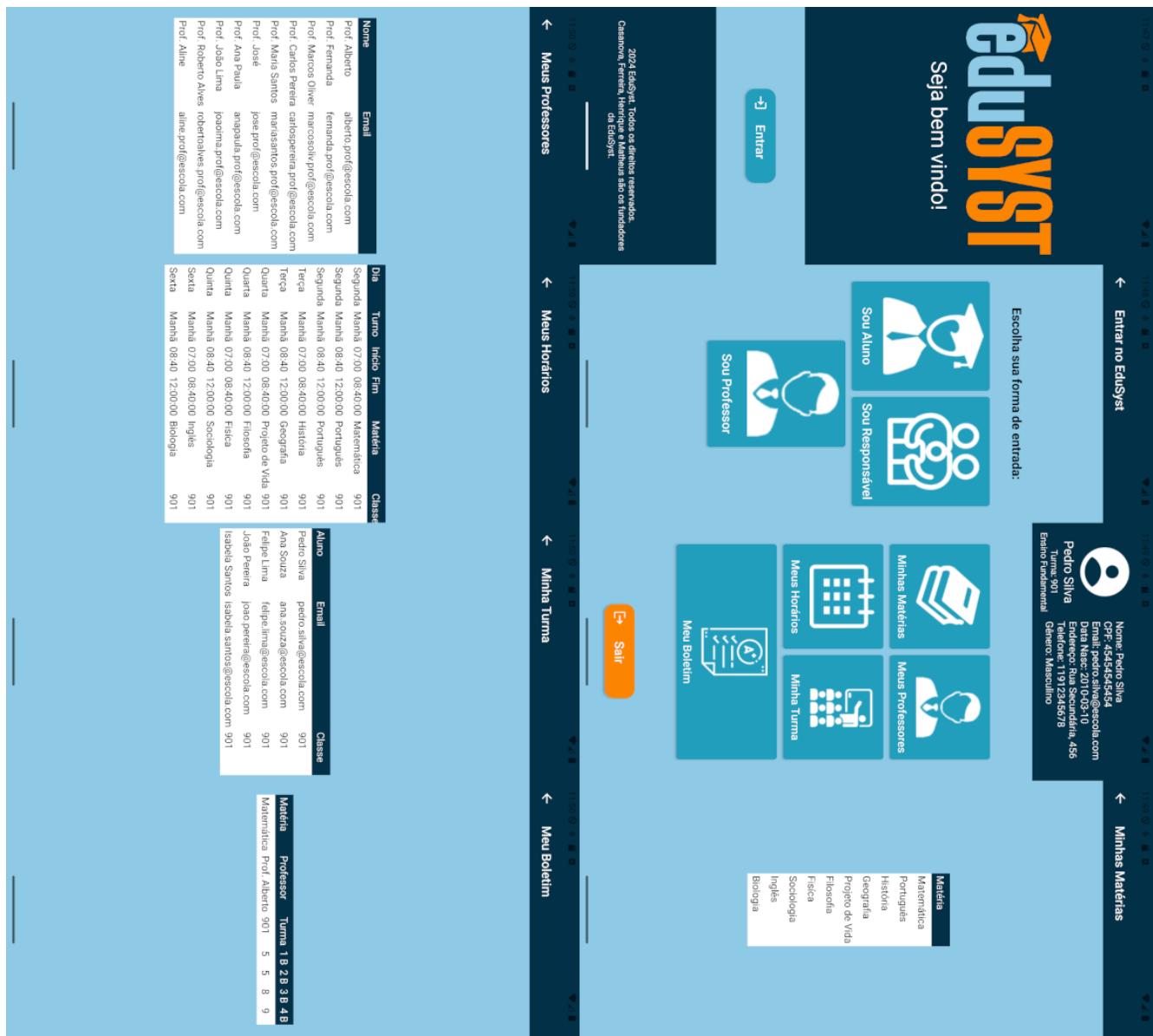


Figura 11: Telas - Login, Opções, Matérias, Professores, Horários, Turma e Boletim

12. BANCO DE DADOS

12.1. DIAGRAMA DE ENTIDADE-RELACIONAMENTO

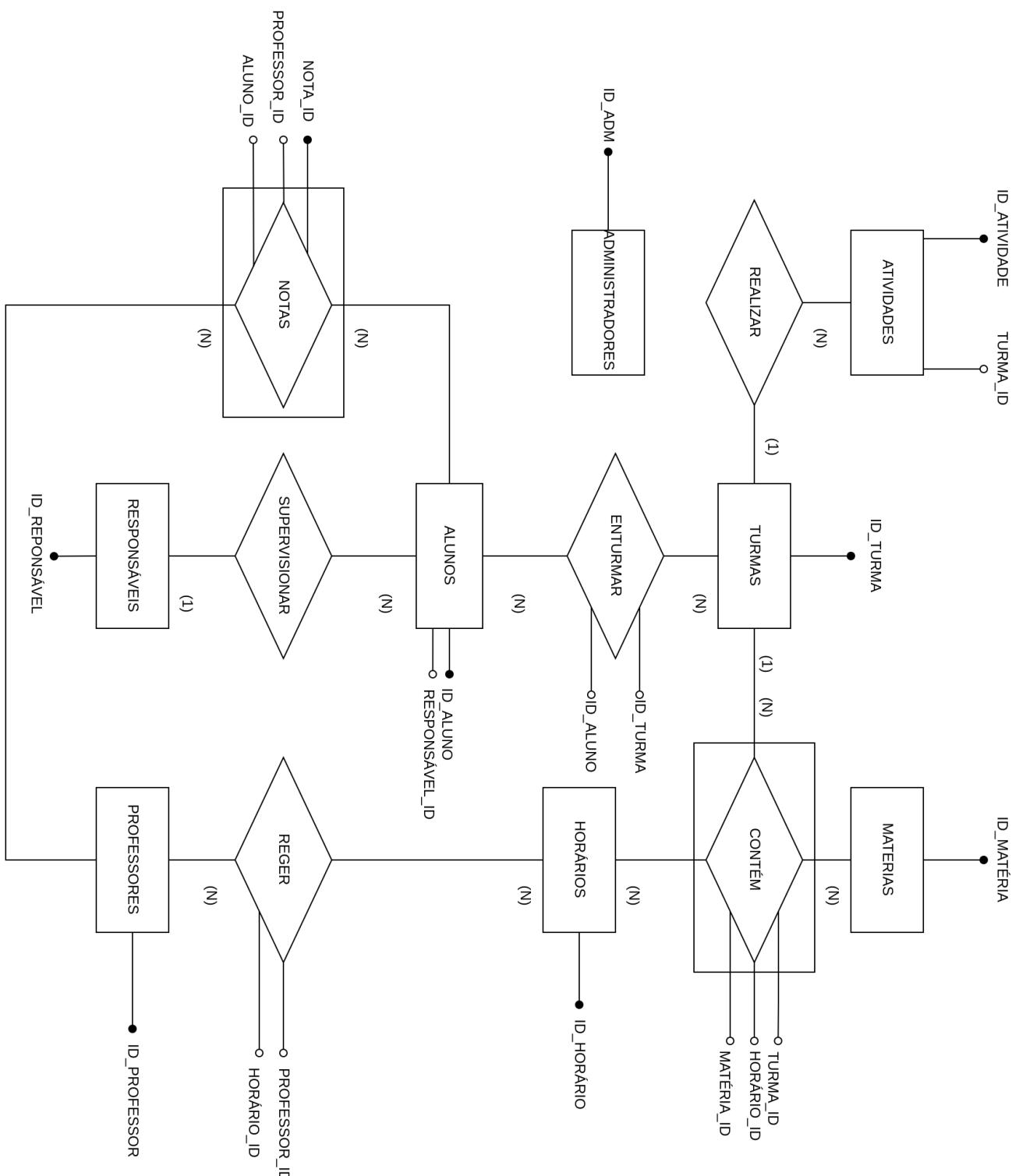


Figura 12: DER

12.2. DIAGRAMA LÓGICO

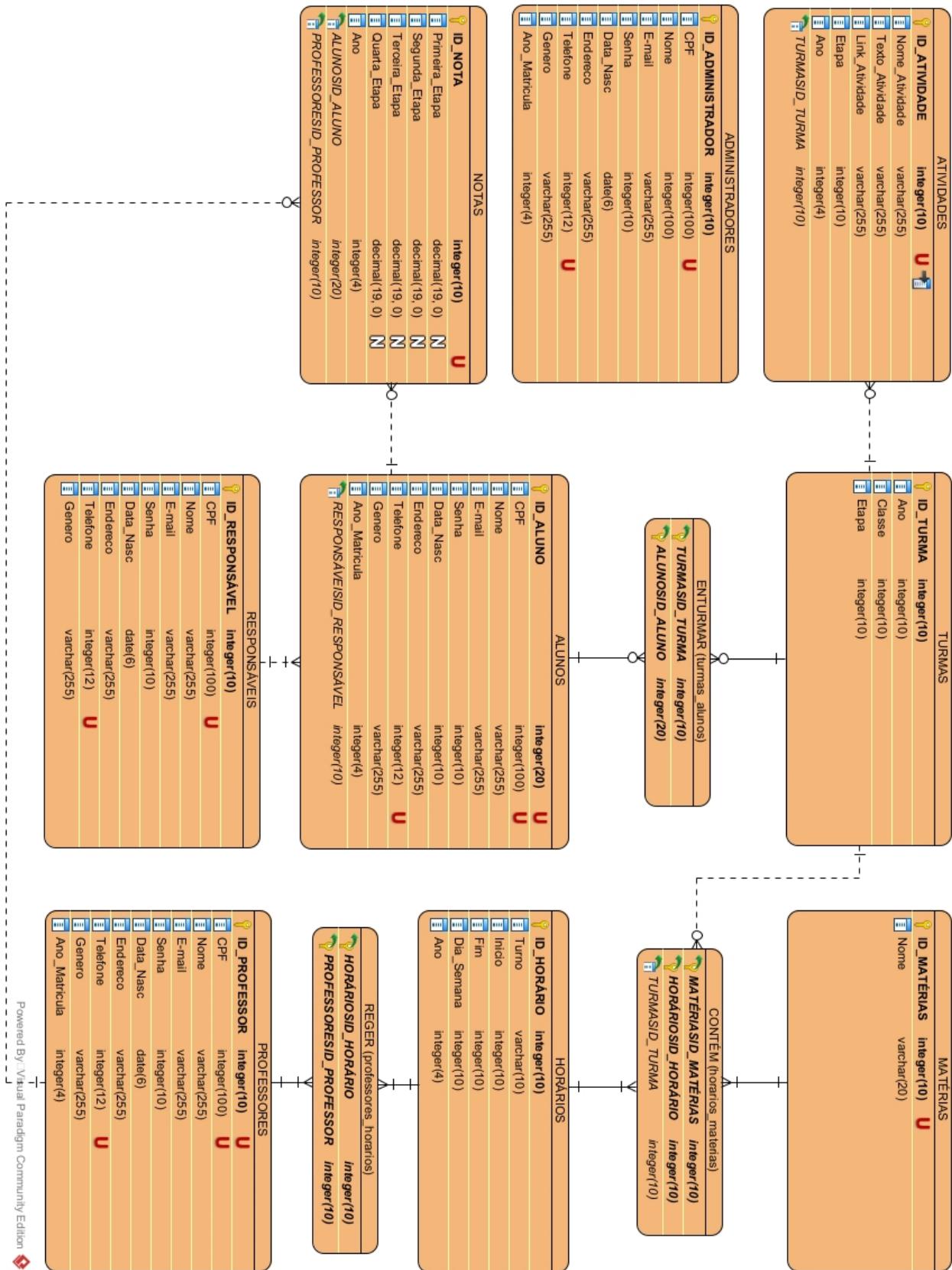
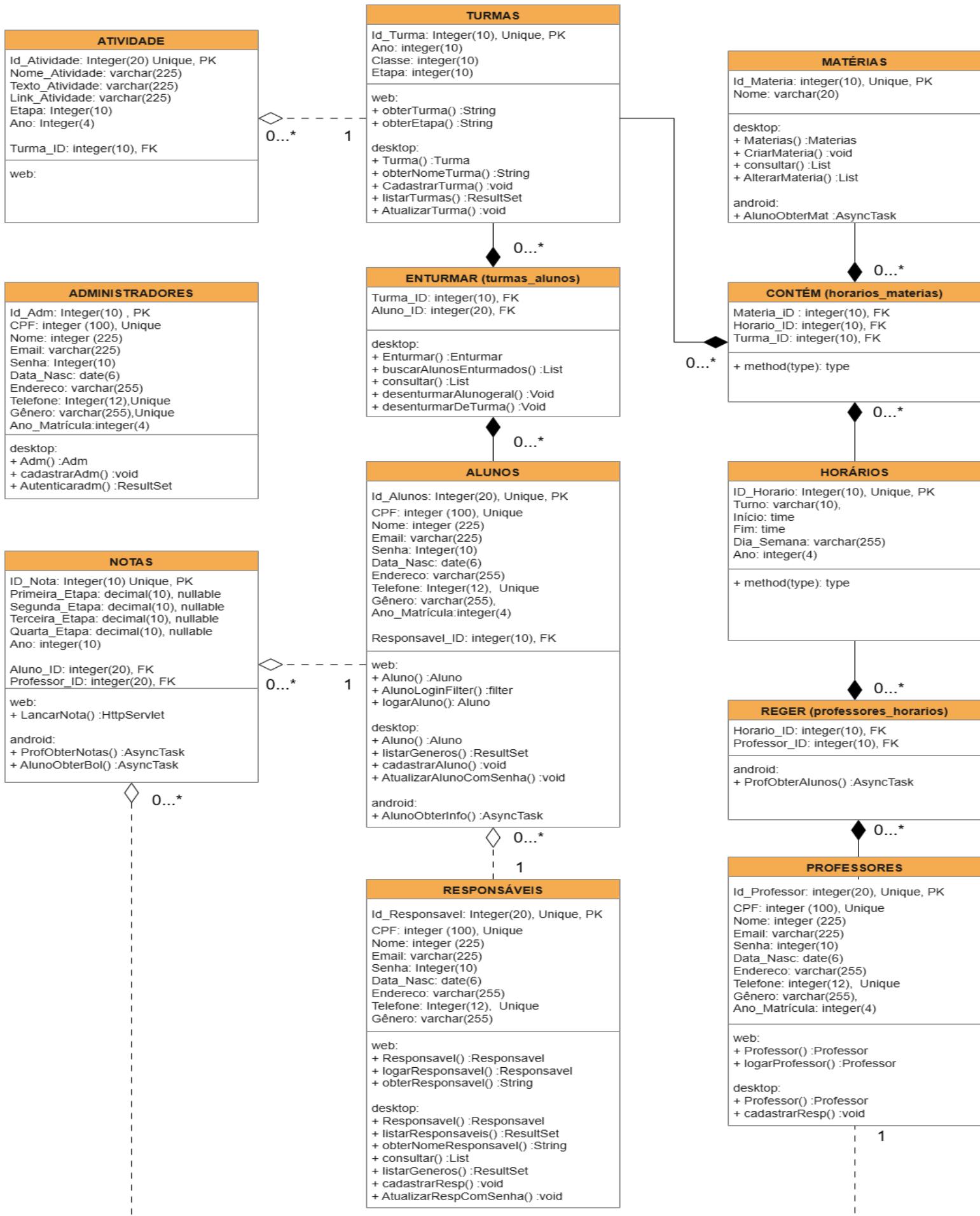


Figura 13: Diagrama Lógico

12.3. DIAGRAMA DE CLASSES



12.4. OBSERVAÇÕES SOBRE A ENTIDADE NOTAS

A tabela Notas no banco de dados do EduSyst estabelece relacionamentos entre os Alunos e os Professores. Embora à primeira vista esses relacionamentos possam parecer circulares, na verdade eles são relacionamentos unidirecionais que seguem uma estrutura hierárquica clara, sem a formação de laços circulares.

Na estrutura, cada nota depende exclusivamente de duas entidades principais: o Aluno que a recebe e o Professor que a atribui. A chave estrangeira Aluno_ID conecta a tabela de Notas à tabela de Alunos, e a chave Professor_ID faz o vínculo com a tabela de Professores. No entanto, esses relacionamentos não são circulares, pois a definição de uma nota não retroalimenta nenhum dos dois lados — os dados de notas fluem unicamente do aluno e do professor para a tabela de Notas.

Em termos práticos, uma relação circular seria se os dados tivessem que voltar ao ponto de origem em um ciclo contínuo, ou se houvesse dependência cíclica para resolver as informações de maneira infinita. No caso da tabela Notas, o relacionamento segue uma hierarquia linear: os professores atribuem notas aos alunos, e essas notas são armazenadas como registros específicos de cada aluno-professor na tabela Notas. Não há necessidade de uma entidade depender do retorno de outra para completar o ciclo.

12.5. BACKUP E PARTICIONAMENTO DOS DADOS

O sistema de backup e armazenamento de dados implementado no banco de dados do EduSyst visa garantir a integridade e preservação das informações dos usuários. Para isso, são utilizadas tabelas “mortas” não representadas no diagrama de classes (TM_Responsaveis, TM_Alunos e TM_Professores), que atuam como repositórios de backup particionados para os dados de usuários, sendo cada tabela configurada com cinco partições proporcionais, definidas pela chave primária dos usuários. As tabelas de backup replicam as estruturas principais e contêm os campos relevantes, como CPF, Nome, Email e outros atributos específicos de cada entidade.

Para garantir que qualquer novo dado seja automaticamente registrado nas tabelas de backup, foram criadas procedures triggers (disparadores) que executam ações após a inserção de registros nas tabelas respectivas. Cada trigger — BackupResponsaveis, BackupAlunos e BackupProfessores — é programado para, após um INSERT, inserir uma cópia exata dos dados

nas respectivas tabelas mortas. Assim, é possível manter um histórico de docentes e alunos, fazer seleções de forma mais prática (graça às partições) e facilitando a recuperação de dados, caso haja necessidade.

12.6. AMOSTRAGEM DE DADOS FICTÍCIOS

Segue um excerto das informações fictícias contidas no banco de dados por padrão, disponíveis no momento de sua criação.

	Id_Adm	CPF	Nome	Email	Senha	Data_Nasc	Endereco	Telefone	Genero
▶	1	11111111111	Admin Gabriel Henrique	Gabriel.admin@escola.com	admin1111	1981-04-20	Rua Principal, 123	11987654321	Masculino
	2	22222222222	Admin Gabriel Casanova	Casanova.admin@escola.com	admin2222	1981-09-10	Av.Getúlio Vargas, 124	12984654725	Masculino
	3	33333333333	Admin Gabriel Ferreira	Ferreira.admin@escola.com	admin3333	1984-07-19	Av.Nilo Peçanha, 6	15974554925	Masculino
	4	44444444444	Admin Matheus Ferreira	Matheus.admin@escola.com	admin4444	1980-08-29	Av.Brasil, 140	15974565050	Masculino

Figura 15: Tabela Administradores

	Id_Responsavel	CPF	Nome	Email	Senha	Data_Nasc	Endereco	Telefone	Genero
▶	1	11222222222	Maria da Silva	Maria.silva@dominio.com	resp1000	1970-05-14	Rua Secundária, 456	11987654322	Feminino
	2	11333333333	Carlos Souza	Carlos.souza@dominio.com	resp1001	1976-09-04	Rua Tertiária, 789	11986654324	Masculino
	3	11444444444	Mario Jorge	Mario.souza@domirio.com	resp1002	1980-08-01	Rua Vagner Mourão, 89	11986654325	Masculino
	4	11555555555	Ricardo Martins	Ricardo.martins@domirio.com	senha1003	1981-04-18	São José , 606	1190123456	Masculino
	5	11666666666	Larissa Fernandes	Larissa.fernandes@dominio.com	resp1004	1996-09-22	Rua Getúlio Vargas, 707	11901234567	Feminino
	6	11777777777	Patrícia Rocha	Patricia.rocha@dominio.com	resp1005	1986-09-22	Rua Santo Antônio, 507	11901235568	Feminino
	7	11888888888	Jorge Filho	Jorge.filho@dominio.com	resp1006	1981-08-30	Rua Tiradentes, 88	11986654329	Masculino
	8	11999999999	Carlos Pereira	Carlos.Pereira@dominio.com	resp1007	1985-07-15	Rua Sete de Setembro, 100	12986654321	Masculino
	9	12111111111	Pedro Santos	Pedro.Santos@dominio.com	resp1008	1987-09-05	Rua Aracaju, 102	12986654322	Masculino
	10	12222222222	Maria Oliveira	Maria.Oliveira@dominio.com	resp1009	1982-02-24	Rua Gaspar Dutra, 102	12986654323	Feminino

Figura 16: Tabela Responsáveis

	Id_Aluno	CPF	Nome	Email	Senha	Data_Nasc	Endereco	Telefone	Genero	Responsavel_ID
▶	1	45454545454	Pedro Silva	pedro.silva@escola.com	aluno1000	2010-03-10	Rua Secundária, 456	11912345678	Masculino	1
	2	55555555555	Ana Souza	ana.souza@escola.com	aluno1001	2011-07-19	Rua Tertiária, 789	11923456789	Feminino	2
	3	54545454545	Felipe Lima	felipe.lima@escola.com	aluno1002	2010-03-14	Rua Vagner Mourão, 89	11934567890	Masculino	3
	4	78945612300	João Pereira	joao.pereira@escola.com	aluno1003	2011-08-15	São José , 606	11945678901	Masculino	4
	5	34567890123	Isabela Santos	isabela.santos@escola.com	aluno1004	2010-03-14	Rua Getúlio Vargas, 707	11956789012	Masculino	5
	6	44444444444	Rafael Almeida	rafael.almeida@escola.com	aluno1005	2010-05-14	Rua Santo Antônio, 507	11967890123	Masculino	6
	7	44444444444	Beatriz Rocha	beatriz.rocha@escola.com	aluno1006	2010-06-20	Rua Tiradentes, 88	11978901234	Masculino	7
	8	44444444443	Lucas Ferreira	lucas.ferreira@escola.com	aluno1007	2010-07-14	Rua Sete de Setembro, 100	11989012345	Masculino	8
	9	44444444444	Mariana Souza	mariana.souza@escola.com	aluno1008	2010-10-14	Rua Aracaju, 102	11990123456	Masculino	9
	10	44444444445	Ana Costa	ana.costa@escola.com	aluno1009	2011-11-18	Rua Gaspar Dutra, 102	11901234567	Masculino	10

Figura 17: Tabela Alunos

	Id_Profes	CPF	Nome	Email	Senha	Data_Nasc	Endereco	Telefone	Genero
▶	1	66666666661	Prof. Alberto	alberto.prof@escola.com	prof1001	1970-02-10	Rua das Palmeiras, 123	11987654366	Masculino
	2	77777777771	Prof. Fernanda	fernanda.prof@escola.com	prof1002	1972-06-27	Avenida Central, 456	11987654349	Feminino
	3	77777777772	Prof. Marcos Oliver	marcosoliv.prof@escola.com	prof1003	1970-07-25	Rua dos Lírios, 789	11987654347	Masculino
	4	77777777773	Prof. João Lima	joaoima.prof@escola.com	prof1004	1980-08-22	Praça da Alegria, 101	11987654355	Masculino
	5	77777777774	Prof. Maria Santos	mariasantos.prof@escola.com	prof1005	1979-02-21	Rua do Sol, 202	11987654325	Feminino
	6	77777777775	Prof. Roberto Alves	robertoalves.prof@escola.com	prof1006	1982-05-15	Avenida das Flores, 303	11987654324	Masculino
	7	77777777776	Prof. Ana Paula	anapaula.prof@escola.com	prof1007	1981-04-18	Rua da Esperança, 404	11987654323	Feminino
	8	77777777777	Prof. Carlos Pereira	carlosperreira.prof@escola.com	prof1008	1983-03-10	Rua das Orquídeas, 505	11987654322	Masculino
	9	77777777778	Prof. José	jose.prof@escola.com	prof1009	1984-02-09	Avenida dos Saberes, 606	11987654321	Masculino
	10	77777777779	Prof. Aline	aline.prof@escola.com	prof1010	1986-01-08	Rua da Harmonia, 707	11987654319	Feminino

Figura 18: Tabela Professores

	Id_Materia	Nome
▶	1	Matemática
	2	Português
	3	História
	4	Geografia
	5	Projeto de Vida
	6	Filosofia
	7	Física
	8	Sociologia
	9	Inglês
	10	Biologia

Figura 19: Tabela Matérias

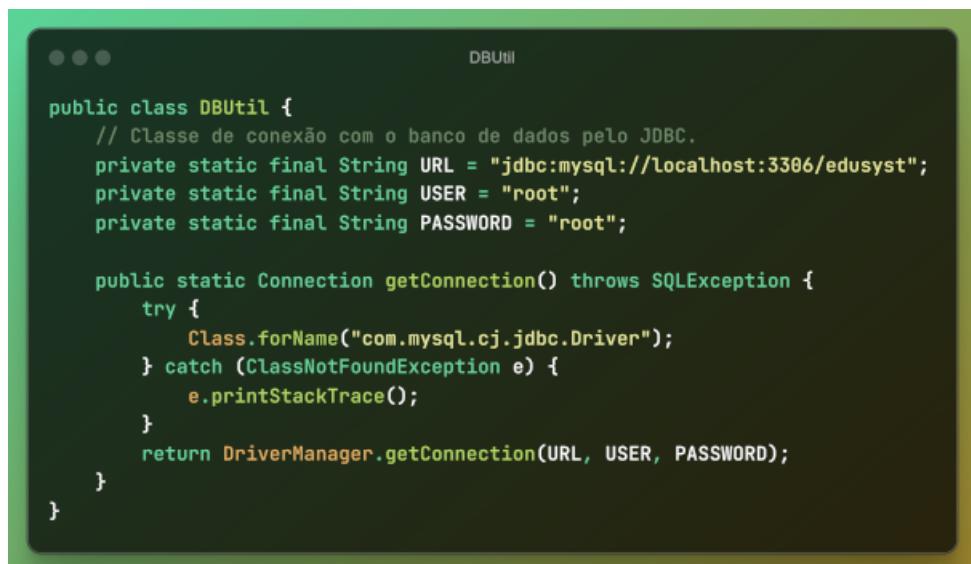
	Id_Horario	Turno	Inicio	Fim	Dia_Semana
▶	1	Manhã	07:00:00	08:40:00	Segunda
	2	Manhã	08:40:00	12:00:00	Segunda
	3	Manhã	07:00:00	08:40:00	Terça
	4	Manhã	08:40:00	12:00:00	Terça
	5	Manhã	07:00:00	08:40:00	Quarta
	6	Manhã	08:40:00	12:00:00	Quarta
	7	Manhã	07:00:00	08:40:00	Quinta
	8	Manhã	08:40:00	12:00:00	Quinta
	9	Manhã	07:00:00	08:40:00	Sexta
	10	Manhã	08:40:00	12:00:00	Sexta

Figura 20: Tabela Horários

13. PRINTS DO NÚCLEO DO SISTEMA

A seguir, são apresentados trechos essenciais do código do sistema, destacando os métodos e classes fundamentais. Esse excerto não representa a totalidade do código, mas sim áreas notáveis de cada parte do sistema.

13.1. JAVA WEB / JSP



```

DBUtil

public class DBUtil {
    // Classe de conexão com o banco de dados pelo JDBC.
    private static final String URL = "jdbc:mysql://localhost:3306/edusyst";
    private static final String USER = "root";
    private static final String PASSWORD = "root";

    public static Connection getConnection() throws SQLException {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
        return DriverManager.getConnection(URL, USER, PASSWORD);
    }
}

```

Figura 21: Conexão com o Banco de Dados MySQL

```

DBUtil

public class AlunoDao {
    public Aluno logarAluno(String cpf, String email, String senha) {
        Aluno aluno = null; // Initialize Aluno as null
        // Query de login aluno.
        String sql = "SELECT * FROM alunos WHERE CPF = ? AND Email = ? AND Senha = ?";
        try (Connection conn = DBUtil.getConnection()) { // Tentar conexão com DB
            PreparedStatement stmt = conn.prepareStatement(sql);
            stmt.setString(1, cpf);           // definir os parametros da query com base
            stmt.setString(2, email);         // no que foi informado pelo login.
            stmt.setString(3, senha);
            ResultSet rs = stmt.executeQuery(); // executar a query MYSQL.
            // Criar objeto aluno com base nas informações do banco de dados
            if (rs.next()) {
                aluno = new Aluno();
                aluno.setId(rs.getInt("Id_Aluno"));
                aluno.setIdResponsavel(rs.getInt("Responsavel_Id"));
                //aluno.setIdTurma(rs.getInt("Turma_Id"));
                aluno.setCpf(rs.getString("CPF"));
                aluno.setNome(rs.getString("nome"));
                aluno.setEmail(rs.getString("email"));
                aluno.setDataNasc(rs.getString("data_nasc"));
                aluno.setEndereco(rs.getString("endereco"));
                aluno.setTelefone(rs.getString("telefone"));
                aluno.setGenero(rs.getString("genero"));
                aluno.setSenha(rs.getString("senha"));
            }
        } catch (SQLException e) {
            e.printStackTrace(); // Printar erro caso uma exception ocorra.
        }
        return aluno; // Retornar o objeto aluno (ou null, se nenhum objeto for encontrado...)
    }

    // Função para checar se o CPF do aluno existe no
    // banco de dados.
    public boolean checarSeExiste(String cpf) {
        try (Connection conn = DBUtil.getConnection()) {
            String sql = "SELECT 1 FROM alunos WHERE CPF = ?";
            PreparedStatement stmt = conn.prepareStatement(sql);
            stmt.setString(1, cpf);
            ResultSet rs = stmt.executeQuery();
            return rs.next();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return false;
    }
}

```

Figura 22: DAO do Aluno



```

public class LancarNota extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

        // Receber dados com base no formulário.
        String ano = Integer.toString(Year.now().getValue());
        String idProfessor = request.getParameter("professorId");
        String turma = request.getParameter("turma");
        String aluno = request.getParameter("aluno");
        String materia = request.getParameter("materia");
        String primeira_etapa = request.getParameter("primeira_etapa");
        String segunda_etapa = request.getParameter("segunda_etapa");
        String terceira_etapa = request.getParameter("terceira_etapa");
        String quarta_etapa = request.getParameter("quarta_etapa");

        Connection conn = null; // MySQL
        Statement stmt = null;
        ResultSet rs = null;

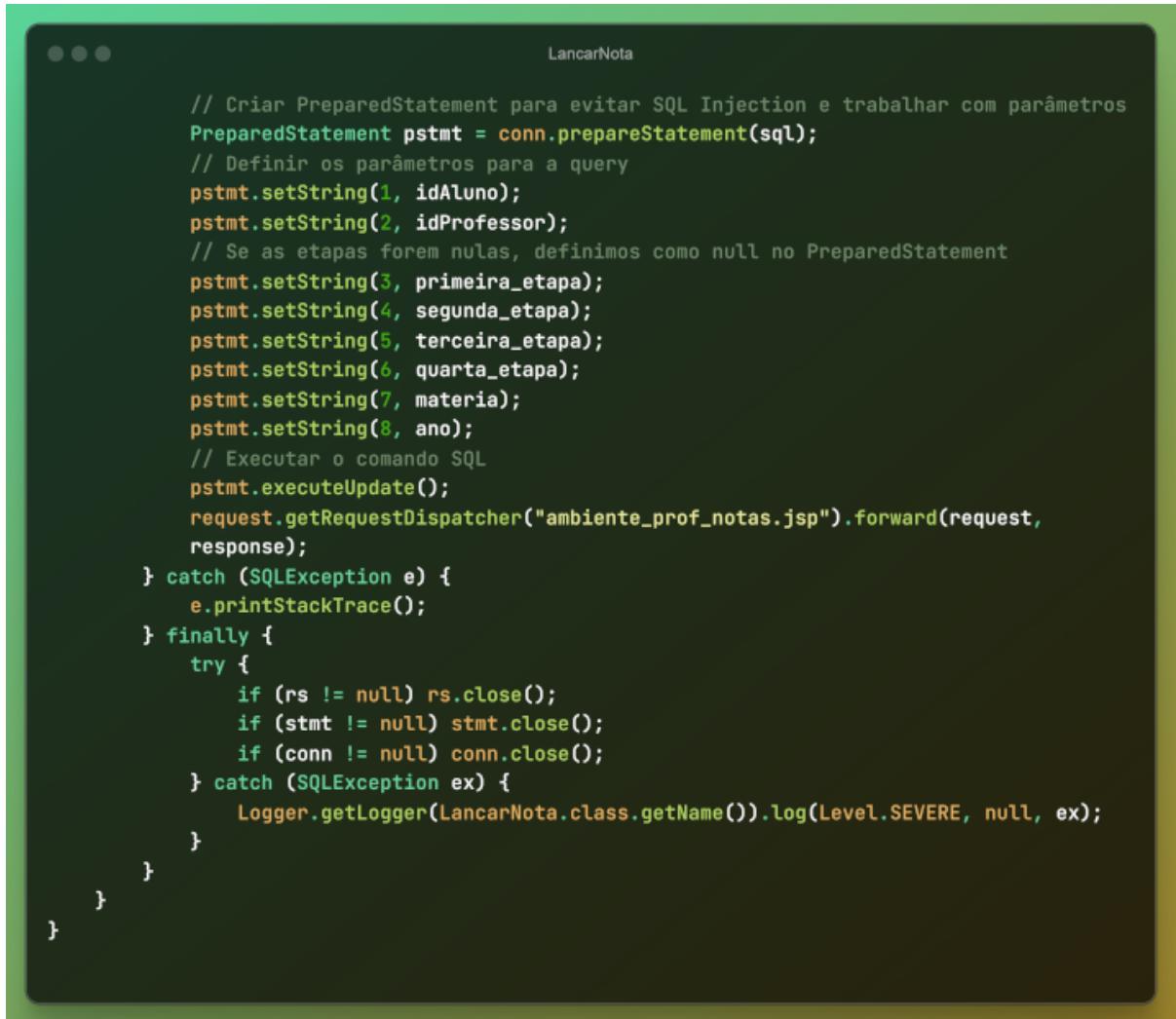
        try {
            conn = DBUtil.getConnection(); // Conectar com o banco de dados.
            stmt = conn.createStatement(); // Preparar um comando SQL.
            // Obter id da turma
            String idTurma = null;
            String sql_turma = "select turmas.id_turma from turmas "
                + "where turmas.classe = " + turma
                + " and turmas.ano = " + Year.now().getValue();
            rs = stmt.executeQuery(sql_turma); // Executar o comando SQL.
            if (rs.next()) {
                idTurma = rs.getString("turmas.id_turma");
            }
            // Obter id do aluno
            String idAluno = null;
            String sql_aluno = "select distinct id_aluno from alunos "
                + "inner join turmas_alunos on turmas_alunos.Aluno_ID = alunos.Id_Aluno "
                + "where turmas_alunos.Turma_ID = " + idTurma
                + " and alunos.nome = '" + aluno + "'";
            rs = stmt.executeQuery(sql_aluno); // Executar o comando SQL.
            if (rs.next()) {
                idAluno = rs.getString("alunos.id_aluno");
            }

            if (primeira_etapa == ""){primeira_etapa=null;};
            if (segunda_etapa == ""){segunda_etapa=null;};
            if (terceira_etapa == ""){terceira_etapa=null;};
            if (quarta_etapa == ""){quarta_etapa=null;};

            String sql = "INSERT INTO Notas (Aluno_ID, Professor_ID, Primeira_Etapa,
                Segunda_Etapa, Terceira_Etapa, Quarta_Etapa, Nome_Materia, Ano) "
                + "VALUES (?, ?, ?, ?, ?, ?, ?, ?) "
                + "ON DUPLICATE KEY UPDATE "
                + "Primeira_Etapa = IF VALUES(Primeira_Etapa) IS NOT NULL, VALUES(Primeira_Etapa),
                    Primeira_Etapa, "
                + "Segunda_Etapa = IF VALUES(Segunda_Etapa) IS NOT NULL, VALUES(Segunda_Etapa),
                    Segunda_Etapa, "
                + "Terceira_Etapa = IF VALUES(Terceira_Etapa) IS NOT NULL, VALUES(Terceira_Etapa),
                    Terceira_Etapa, "
                + "Quarta_Etapa = IF VALUES(Quarta_Etapa) IS NOT NULL, VALUES(Quarta_Etapa),
                    Quarta_Etapa";
        }
    }
}

```

Figura 23: Lançamento de Nota pelos Professores (print n.1)



```
LancarNota

    // Criar PreparedStatement para evitar SQL Injection e trabalhar com parâmetros
    PreparedStatement pstmt = conn.prepareStatement(sql);
    // Definir os parâmetros para a query
    pstmt.setString(1, idAluno);
    pstmt.setString(2, idProfessor);
    // Se as etapas forem nulas, definimos como null no PreparedStatement
    pstmt.setString(3, primeira_etapa);
    pstmt.setString(4, segunda_etapa);
    pstmt.setString(5, terceira_etapa);
    pstmt.setString(6, quarta_etapa);
    pstmt.setString(7, materia);
    pstmt.setString(8, ano);
    // Executar o comando SQL
    pstmt.executeUpdate();
    request.getRequestDispatcher("ambiente_prof_notas.jsp").forward(request,
        response);
} catch (SQLException e) {
    e.printStackTrace();
} finally {
    try {
        if (rs != null) rs.close();
        if (stmt != null) stmt.close();
        if (conn != null) conn.close();
    } catch (SQLException ex) {
        Logger.getLogger(LancarNota.class.getName()).log(Level.SEVERE, null, ex);
    }
}
}
```

Figura 24: Lançamento de Nota pelos Professores (print n.2)

```

    Site

<!DOCTYPE html>
<html>
<head>
<%@include file="parts/meta.jsp"%>

<!-- Carregar bibliotecas -->
<link rel="stylesheet" href="styles/bootstrap.min.css">
<script src="js/bootstrap.bundle.min.js"></script>
<script src="https://kit.fontawesome.com/19e40268ee.js" crossorigin="anonymous"></script>

<title>EduSyst</title>
<link rel="stylesheet" href="styles/styles.css">
</head>

<body>
<!-- Barra de Navegação -->
<%@include file="parts/header.jsp"%>

<!-- Página -->
<main>
    <div class="container">
        <div id="pagina" class="row justify-content-md-center min-vh-100">
            <!-- Barra Lateral / Perfil do Usuário -->
            <%@include file="parts/perfil_aluno.jsp">

            <!-- Menu Principal -->
            <div class="col p-3">
                <!-- Título -->
                <div class="row mb-3">
                    <h2 class="text-center">Seja bem-vindo ao ambiente do aluno!</h2>
                </div>

                <!-- Opções -->
                <div class="row justify-content-center"> <!-- Primeira linha -->
                    <div class="col m-2 p-3 botao_ambiente"> <!-- Matérias -->
                        <a href="ambiente_aluno_materias.jsp">
                            
                            <strong class="fs-5">Minhas Matérias</strong>
                        </a>
                    </div>
                    <div class="col m-2 p-3 botao_ambiente"> <!-- Professores -->
                        <a href="ambiente_aluno_professores.jsp">
                            
                            <strong class="fs-5">Meus Professores</strong>
                        </a>
                    </div>
                    <div class="col m-2 p-3 botao_ambiente"> <!-- Horários -->
                        <a href="ambiente_aluno_horarios.jsp">
                            
                            <strong class="fs-5">Meus Horários</strong>
                        </a>
                    </div>
                </div>
            </div>
        </div>
    </div>
</main>

```

Figura 25: Ambiente do Usuário (print n.1)



The screenshot shows a mobile browser window with a dark theme. At the top, there are three dots on the left and the word "Site" on the right. The main content area displays the following HTML code:

```
<div class="row justify-content-center"> <!-- Segunda linha -->
    <div class="col m-2 p-3 botao_ambiente"> <!-- Turma -->
        <a href="ambiente_aluno_turma.jsp">
            
            <strong class="fs-5">Minha Turma</strong>
        </a>
    </div>
    <div class="col m-2 p-3 botao_ambiente"> <!-- Boletim -->
        <a href="ambiente_aluno_boletim.jsp">
            
            <strong class="fs-5">Meu Boletim</strong>
        </a>
    </div>
    <div class="col m-2 p-3 botao_ambiente"> <!-- Atividades -->
        <a href="ambiente_aluno_atividades.jsp">
            
            <strong class="fs-5">Minhas Atividades</strong>
        </a>
    </div>
</div>

<div class="row justify-content-center"> <!-- Terceira linha -->
    <div class="col m-2 p-3 botao_ambiente"> <!-- Chat -->
        <a href="ambiente_aluno_chat.jsp">
            
            <strong class="fs-5">Chat Escolar</strong>
        </a>
    </div>
    </div>
</div>
</div>
</main>

<!-- Rodapé -->
<%@include file="parts/footer.jsp"%>
</body>
</html>
```

Figura 26: Ambiente do Usuário (print n.2)

13.2. ANDROID STUDIO / JAVA / PHP



```

db

// config.php
<?php
    define("DB_HOST", "localhost");
    define("DB_USER", "root");
    define("DB_PASSWORD", "");
    define("DB_NAME", "edusyst");
?>

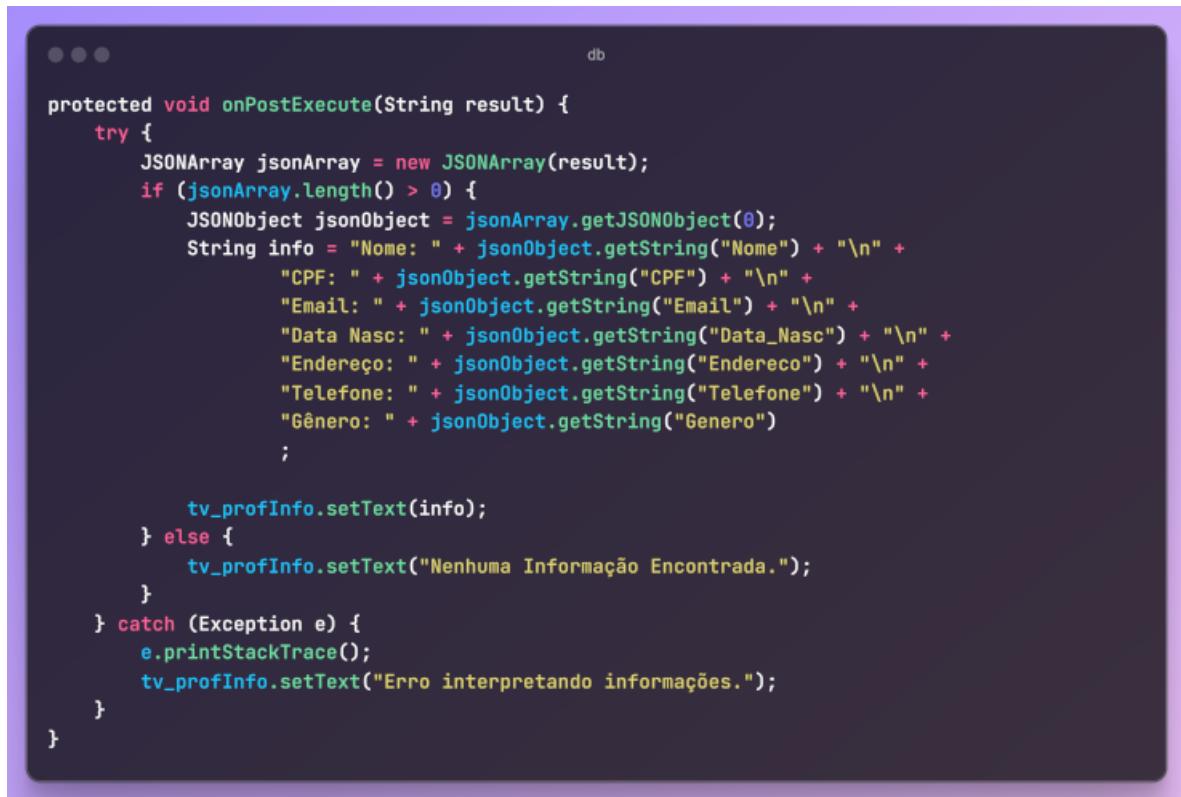
// db-util.php
<?php
    include_once 'config.php';

    // Create connection
    $con = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

    // Check connection
    if (!$con) {
        die("Connection failed: " . mysqli_connect_error());
    }
?>

```

Figura 27: Conexão com o Banco de Dados MySQL



```

protected void onPostExecute(String result) {
    try {
        JSONArray jsonArray = new JSONArray(result);
        if (jsonArray.length() > 0) {
            JSONObject jsonObject = jsonArray.getJSONObject(0);
            String info = "Nome: " + jsonObject.getString("Nome") + "\n" +
                "CPF: " + jsonObject.getString("CPF") + "\n" +
                "Email: " + jsonObject.getString("Email") + "\n" +
                "Data Nasc: " + jsonObject.getString("Data_Nasc") + "\n" +
                "Endereço: " + jsonObject.getString("Endereco") + "\n" +
                "Telefone: " + jsonObject.getString("Telefone") + "\n" +
                "Gênero: " + jsonObject.getString("Genero")
            ;

            tv_profInfo.setText(info);
        } else {
            tv_profInfo.setText("Nenhuma Informação Encontrada.");
        }
    } catch (Exception e) {
        e.printStackTrace();
        tv_profInfo.setText("Erro interpretando informações.");
    }
}

```

Figura 28: Obter Informações de um Professor



The screenshot shows a dark-themed Android application interface. At the top, there are three circular icons on the left and the text "db" on the right. Below this is a scrollable code editor displaying Java code for an `AsyncTask`. The code is as follows:

```
private class LoginTask extends AsyncTask<String, Void, String> {
    @Override
    protected String doInBackground(String... params) {
        String cpf = params[0];
        String email = params[1];
        String senha = params[2];
        String result = "";

        try {
            // Mensagem de debug.
            Log.d("LoginTask", "Attempting to connect to server with CPF: " + cpf);

            // Criar URL e conectar.
            URL url = new URL("http://10.0.2.2/sisescola_web_android/login_resp.php");
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            conn.setRequestMethod("POST");
            conn.setDoOutput(true);
            conn.setDoInput(true);

            // Enviar data POST
            OutputStream os = conn.getOutputStream();
            BufferedWriter writer = new BufferedWriter(new OutputStreamWriter(os, "UTF-8"));
            String postData = "cpf=" + URLEncoder.encode(cpf, "UTF-8") +
                "&email=" + URLEncoder.encode(email, "UTF-8") +
                "&senha=" + URLEncoder.encode(senha, "UTF-8");
            writer.write(postData);
            writer.flush();
            writer.close();
            os.close();

            // Ler resposta
            BufferedReader reader = new BufferedReader(new InputStreamReader(conn.getInputStream()));
            StringBuilder sb = new StringBuilder();
            String line;
            while ((line = reader.readLine()) != null) {
                sb.append(line).append("\n");
            }
            reader.close();
            result = sb.toString();
            conn.disconnect();

            // Loggar resposta para debugar.
            Log.d("LoginTask", "Server response: " + result);
        } catch (Exception e) {
            e.printStackTrace();
        }
        return result;
    }
}
```

Figura 29: Login Responsável

```
<?php
require('db-connect.php');

$id_aluno = $_GET["id_aluno"];

$sql = "SELECT distinct notas.nome_materia AS materia_nome, professores.Nome AS professor_nome, "
    . "turmas.Classe AS turma_classe, notas.Primeira_Etapa as primeira_etapa, "
    . "notas.Segunda_Etapa as segunda_etapa, notas.Terceira_Etapa as terceira_etapa, "
    . "notas.Quarta_Etapa as quarta_etapa "
    . "FROM professores "
    . "INNER JOIN professores_horarios ON professores.Id_Professor =
professores_horarios.Professor_ID "
    . "INNER JOIN horarios ON horarios.Id_Horario = professores_horarios.Horario_ID "
    . "INNER JOIN horarios_materias ON horarios_materias.Horario_ID =
horarios.Id_Horario "
    . "INNER JOIN materias ON materias.Id_Materia = horarios_materias.Materia_ID "
    . "INNER JOIN turmas ON turmas.Id_Turma = horarios_materias.Turma_ID "
    . "INNER JOIN turmas_alunos ON turmas_alunos.Turma_ID = turmas.Id_Turma "
    . "INNER JOIN alunos ON alunos.Id_Aluno = turmas_alunos.Aluno_ID "
    . "INNER JOIN notas ON notas.Aluno_Id = alunos.Id_Aluno "
    . "AND notas.Professor_ID = professores.Id_Professor "
    . "WHERE alunos.Id_Aluno = '$id_aluno'";

$result = mysqli_query($con, $sql);

$data = array();
while ($row = mysqli_fetch_assoc($result)) {
    $data[] = $row;
}

header('Content-Type: application/json');
echo json_encode($data);

mysqli_close($con);
?>
```

Figura 30: Geração do Boletim

13.3. JAVA DESKTOP

```

public class EnturmarDAO {

    public void Enturmar(Enturmar en) throws ClassNotFoundException, SQLException {
        Connection con = ConnectionFactory.getConnection();
        PreparedStatement ps = null;
        try {
            ps = con.prepareStatement("INSERT INTO Turmas_Alunos (Aluno_ID, Turma_ID) VALUES (?, ?)");
            ps.setInt(1, en.getAluno_ID());
            ps.setInt(2, en.getTurma_ID());
            ps.executeUpdate();
            ps.close();
            JOptionPane.showMessageDialog(null, "Aluno enturmado com sucesso.");
        } catch (SQLException e) {
            JOptionPane.showMessageDialog(null, "Erro ao salvar: " + e.getMessage());
        } finally {
            ConnectionFactory.closeConnection(con, ps);
        }
    }

    public List<Aluno> buscarAlunosEnturmados(int turmaId) throws ClassNotFoundException, SQLException {
        Connection con = ConnectionFactory.getConnection();
        PreparedStatement ps = null;
        ResultSet rs = null;
        List<Aluno> alunosEnturmados = new ArrayList<>();

        try {
            String sql = "SELECT a.Id_Aluno, a.NomeA FROM Alunos a "
                + "JOIN Turmas_Alunos ta ON a.Id_Aluno = ta.Aluno_ID "
                + "WHERE ta.Turma_ID = ?";
            ps = con.prepareStatement(sql);
            ps.setInt(1, turmaId);
            rs = ps.executeQuery();

            while (rs.next()) {
                Aluno aluno = new Aluno();
                aluno.setId_Aluno(rs.getInt("Id_Aluno"));
                aluno.setNomeA(rs.getString("NomeA"));
                alunosEnturmados.add(aluno);
            }
        } catch (SQLException e) {
            JOptionPane.showMessageDialog(null, "Erro ao buscar alunos enturmados: " + e.getMessage());
        }
    }
}

```

Figura 31: Enturmação

```

public void cadastrarAluno(Aluno a) throws ClassNotFoundException, SQLException {
    Connection con = ConnectionFactory.getConnection();
    PreparedStatement ps = null;
    try {
        ps = con.prepareStatement("INSERT INTO Alunos (Id_Aluno, CPF, Nome, Email, Senha, Data_Nasc, Endereco, Telefone, Genero, Responsavel_ID)VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)");
        ps.setInt(1, a.getId_Aluno());
        ps.setString(2, a.getCPFAluno());
        ps.setString(3, a.getNomeA());
        ps.setString(4, a.getEmailA());
        ps.setString(5, a.getSenhaA());
        ps.setString(6, a.getDataA());
        ps.setString(7, a.getEnderecoA());
        ps.setString(8, a.getTelefoneA());
        ps.setString(9, a.getGeneroA());
        ps.setInt(10, a.getId_Responsavel());
        ps.executeUpdate();
        ps.close();
    } catch (SQLException e) {
        JOptionPane.showMessageDialog(null, "Erro ao salvar: " + e.getMessage());
    } finally {
        ConnectionFactory.closeConnection(con, ps);
    }
}

```

Figura 32: Cadastro de Alunos

```

public void cadastrarAdm(Adm adm) throws ClassNotFoundException, SQLException {
    Connection con = ConnectionFactory.getConnection();
    PreparedStatement ps = null;
    try {
        ps = con.prepareStatement("INSERT INTO Administradores (CPF, Nome, Email, Senha, Data_Nasc, Endereco, Telefone, Genero)VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)");
        ps.setString(1, adm.getCPFAdm());
        ps.setString(2, adm.getNomeAdm());
        ps.setString(3, adm.getEmailAdm());
        ps.setString(4, adm.getSenhaAdm());
        ps.setString(5, adm.getData_NascAdm());
        ps.setString(6, adm.getEnderecoAdm());
        ps.setString(7, adm.getTelefoneAdm());
        ps.setString(8, adm.getGeneroAdm());
        ps.executeUpdate();
        ps.close();
        JOptionPane.showMessageDialog(null, "Salvo com sucesso.");
    } catch (SQLException e) {
        JOptionPane.showMessageDialog(null, "Erro ao salvar: " + e.getMessage());
    } finally {
        ConnectionFactory.closeConnection(con, ps);
    }
}

```

Figura 33: Cadastro de Turmas

```

public void cadastrarProf(Professor P) throws ClassNotFoundException, SQLException {
    Connection con = ConnectionFactory.getConnection();
    PreparedStatement ps = null;
    try {
        ps = con.prepareStatement("INSERT INTO Professores (Id_Professor, CPF, Nome, Email, Senha, Data_Nasc, Endereco, Telefone, Genero)VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)");
        ps.setInt(1, P.getId_Professor());
        ps.setString(2, P.getCPFP());
        ps.setString(3, P.getNomeP());
        ps.setString(4, P.getEmailP());
        ps.setString(5, P.getSenhaP());
        ps.setString(6, P.getData_NascP());
        ps.setString(7, P.getEnderecoP());
        ps.setString(8, P.getTelefoneP());
        ps.setString(9, P.getGeneroP());
        int rowsAffected = ps.executeUpdate();
        if (rowsAffected > 0) {
            JOptionPane.showMessageDialog(null, "Aluno criado com sucesso.");
        } else {
            JOptionPane.showMessageDialog(null, "Erro ao inserir aluno.");
        }
    } catch (SQLException e) {
        JOptionPane.showMessageDialog(null, "Erro no banco de dados: " + e.getMessage());
    } finally {
        ConnectionFactory.closeConnection(con, ps);
    }
}

```

Figura 34: Cadastro de Professores

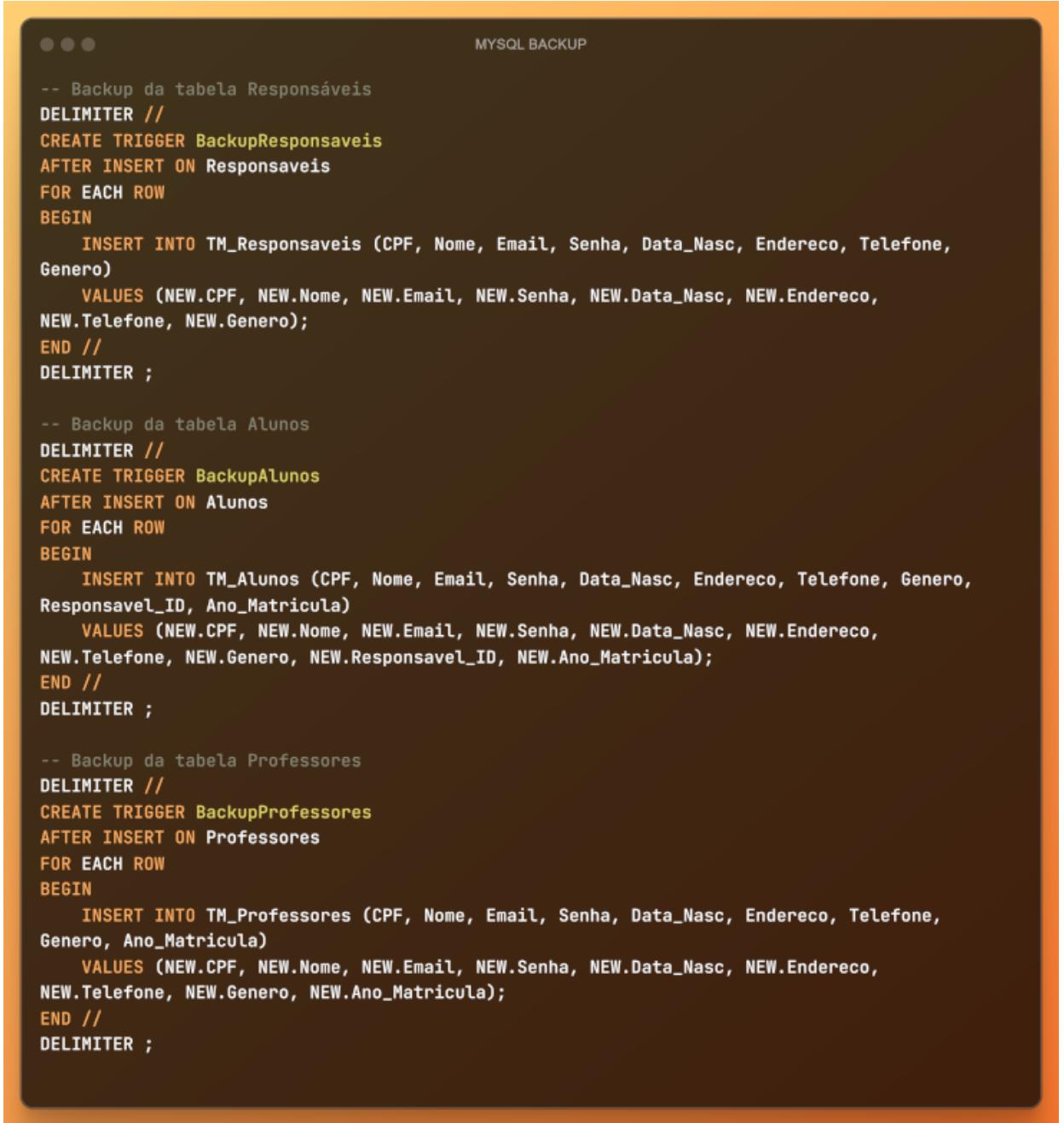
```
public List<Turma> consultarPorTurma(int classe) throws ClassNotFoundException, SQLException {
    Connection con = ConnectionFactory.getConnection();
    PreparedStatement ps = null;
    ResultSet rs = null;
    List<Turma> turmas = new ArrayList<>();
    try {
        String sql = "SELECT a.Nome, a.Email, t.Id_Turma, t.Classe, t.Ano, t.Etapa "
            + "FROM Alunos AS a "
            + "JOIN Turmas_Alunos AS at ON a.Id_Aluno = at.Aluno_ID "
            + "JOIN Turmas AS t ON t.Id_Turma = at.Turma_ID "
            + "WHERE t.Classe = ?";
        ps = con.prepareStatement(sql);
        ps.setInt(1, classe);
        rs = ps.executeQuery();

        Map<Integer, Turma> mapaTurmas = new HashMap<>();

        while (rs.next()) {
            int id = rs.getInt("Id_Turma");
            int classeTurma = rs.getInt("Classe");
            int ano = rs.getInt("Ano");
            String etapa = rs.getString("Etapa");
            Turma t = mapaTurmas.get(classeTurma);
            if (t == null) {
                t = new Turma();
                t.setId_Turma(id);
                t.setClasse(classeTurma);
                t.setAno(ano);
                t.setEtapa(etapa);
                mapaTurmas.put(classeTurma, t);
            }
            Aluno aluno = new Aluno();
            aluno.setNomeA(rs.getString("Nome"));
            aluno.setEmailA(rs.getString("Email"));
            t.addAluno(aluno);
        }
        turmas.addAll(mapaTurmas.values());
    } catch (SQLException e) {
        JOptionPane.showMessageDialog(null, "Erro ao consultar: " + e.getMessage());
    } finally {
        ConnectionFactory.closeConnection(con, ps, rs);
    }
    return turmas;
}
```

Figura 35: Listagem de Turmas

13.4. MYSQL



```
MySQL BACKUP

-- Backup da tabela Responsáveis
DELIMITER //
CREATE TRIGGER BackupResponsaveis
AFTER INSERT ON Responsaveis
FOR EACH ROW
BEGIN
    INSERT INTO TM_Responsaveis (CPF, Nome, Email, Senha, Data_Nasc, Endereco, Telefone,
Genero)
    VALUES (NEW.CPF, NEW.Nome, NEW.Email, NEW.Senha, NEW.Data_Nasc, NEW.Endereco,
NEW.Telefone, NEW.Genero);
END //
DELIMITER ;

-- Backup da tabela Alunos
DELIMITER //
CREATE TRIGGER BackupAlunos
AFTER INSERT ON Alunos
FOR EACH ROW
BEGIN
    INSERT INTO TM_Alunos (CPF, Nome, Email, Senha, Data_Nasc, Endereco, Telefone, Genero,
Responsavel_ID, Ano_Matricula)
    VALUES (NEW.CPF, NEW.Nome, NEW.Email, NEW.Senha, NEW.Data_Nasc, NEW.Endereco,
NEW.Telefone, NEW.Genero, NEW.Responsavel_ID, NEW.Ano_Matricula);
END //
DELIMITER ;

-- Backup da tabela Professores
DELIMITER //
CREATE TRIGGER BackupProfessores
AFTER INSERT ON Professores
FOR EACH ROW
BEGIN
    INSERT INTO TM_Professores (CPF, Nome, Email, Senha, Data_Nasc, Endereco, Telefone,
Genero, Ano_Matricula)
    VALUES (NEW.CPF, NEW.Nome, NEW.Email, NEW.Senha, NEW.Data_Nasc, NEW.Endereco,
NEW.Telefone, NEW.Genero, NEW.Ano_Matricula);
END //
DELIMITER ;
```

Figura 36: Backup das Tabelas

14. CONCLUSÃO

Ao final do desenvolvimento do EduSyst, foi possível consolidar um sistema funcional, voltado para apoiar a gestão escolar em escolas públicas, em especial as de ensino médio no estado do Rio de Janeiro. Nossa objetivo principal foi criar uma ferramenta que tornasse mais prática e organizada a comunicação e o acompanhamento de alunos, parentes e equipe escolar, e acreditamos ter dado um primeiro passo nesse sentido. Conscientes de que muitos aprimoramentos ainda são possíveis, principalmente no que diz respeito à segurança e à escalabilidade do sistema, concluímos esta etapa com a visão de que o EduSyst servirá como um alicerce, permitindo que melhorias futuras possam ser feitas e que mais escolas possam, aos poucos, se beneficiar de uma gestão digital mais estruturada e eficiente.

O desenvolvimento, embora desafiador, proporcionou um aprendizado intenso em programação, design de banco de dados e metodologias de desenvolvimento em equipe, consolidando um conhecimento essencial para a nossa formação. Este trabalho é um ponto de partida, e ficamos satisfeitos com o que conseguimos realizar até aqui. Esperamos que a plataforma contribua para o ambiente escolar e inspire outras iniciativas voltadas para a realidade das escolas públicas.