

**UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CÂMPUS PATO BRANCO
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO
DE SISTEMAS**

AUGUSTO SILVERIO SOARES

**SISTEMA WEB PARA CONTROLE DE MATRÍCULAS E EMISSÃO DE
BOLETIM ESCOLAR**

TRABALHO DE CONCLUSÃO DE CURSO

PATO BRANCO

2014

AUGUSTO SILVERIO SOARES

**SISTEMA WEB PARA CONTROLE DE MATRÍCULAS E EMISSÃO DE
BOLETIM ESCOLAR**

Trabalho de Conclusão de Curso de graduação, apresentado à disciplina de Trabalho de Diplomação, do curso superior de Tecnologia em Análise e Desenvolvimento de Sistemas, da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco, como requisito parcial para obtenção do título de Tecnólogo.

Orientadora: MSc. Rúbia Eliza de Oliveira Schultz **Ascari**

PATO BRANCO

2014

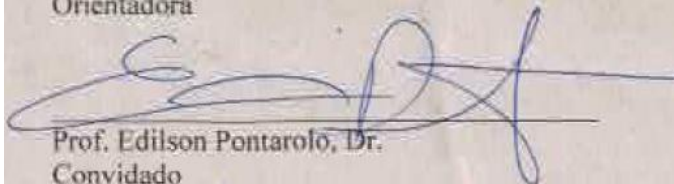
ATA Nº: 244

DEFESA PÚBLICA DO TRABALHO DE DIPLOMAÇÃO DO ALUNO AUGUSTO SILVERIO SOARES.

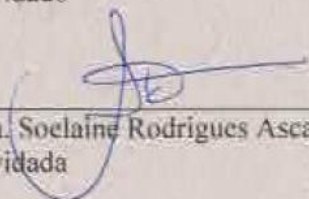
Às 16:00 hrs do dia 10 de novembro de 2014, Bloco V da UTFPR, Câmpus Pato Branco, reuniu-se a banca avaliadora composta pelos professores Rúbia E. O. Schultz Ascari (Orientadora), Edilson Pontarolo (Convidado) e Soelaine Rodrigues Ascari (Convidada), para avaliar o Trabalho de Diplomação do aluno Augusto Silverio Soares, matrícula 1120301, sob o título **Sistema Web para Controle de Matrículas e Emissão de Boletim Escolar**; como requisito final para a conclusão da disciplina Trabalho de Diplomação do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, COADS. Após a apresentação o candidato foi entrevistado pela banca examinadora, e a palavra foi aberta ao público. Em seguida, a banca reuniu-se para deliberar considerando o trabalho **APROVADO**. Às 16:30 hrs foi encerrada a sessão.



Prof.a. Rúbia E. O. Schultz Ascari, M.Sc.
Orientadora



Prof. Edilson Pontarolo, Dr.
Convidado



Prof.a. Soelaine Rodrigues Ascari, M.Sc.
Convidada

RESUMO

SOARES, Augusto Silverio. Sistema Web para controle de matrículas e emissão de boletim escolar. 2014. 46f. Trabalho de conclusão de curso - Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2014.

O funcionamento de uma escola demanda muita informação e realização de procedimentos diversos. Para que uma escola esteja apta a atuar, vários requisitos são necessários, como por exemplo, a composição de um quadro de profissionais, técnicos administrativos e professores e uma matriz curricular que esteja de acordo com a legislação vigente. Diante de tais necessidades verificou-se a oportunidade de desenvolver um sistema web simples para gerenciamento de matrículas e emissão de boletim escolar, ou seja, um sistema para auxiliar no gerenciamento de alguns processos realizados em escolas. A implementação do sistema foi realizada utilizando as linguagens de programação *código aberto* Java e ActionScript Server com MXML, que formam a tecnologia Flex. Para o banco de dados foi utilizado o PostgreSQL, também *código aberto*.

Palavras-chave: Sistema Web. Gestão Escolar. Flex. JPA. Hibernate.

ABSTRACT

SOARES, Augusto Silverio. Web system to enrollments control and issuance of report cards. 2014. 46f. Trabalho de conclusão de curso - Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, Universidade Tecnológica Federal do Paraná. Pato Branco, 2014.

The operation of a school demands a lot of information and performing various procedures. For a school to be able to operate, several requirements are needed, such as the composition of a cadre of professional, administrative and technical teachers and a curriculum that is in line with current legislation. Given these needs there was the opportunity to develop a simple web system for managing enrollment and issuance of report cards, ie, a system to help manage some processes carried out in schools. The implementation of the system was performed using the programming languages Java and open source ActionScript with MXML Server, which form the Flex technology. For the PostgreSQL database, open source was also used.

Palavras-chave: Web System. School Management. Flex. JPA, Hibernate.

LISTA DE FIGURAS

Figura 1 - Integração do framework BlaszeDS.....	23
Figura 2 - Diagrama de casos de uso do sistema desenvolvido.	36
Figura 3 - Diagrama de classes do sistema desenvolvido.	37
Figura 4 – Diagrama de Entidade e Relacionamento do sistema.....	38
Figura 5 - Tela de login do sistema desenvolvido.	39
Figura 6 - Página principal do sistema desenvolvido	40
Figura 7 - Cadastro da Unidade de Ensino.	40
Figura 8 - Tela para manutenção de Unidade de Ensino.	41
Figura 9 - Tela de Matrícula.	42
Figura 10 - Tela de Classe.	42
Figura 11 – Relatório de Boletim Escolar por Matrícula.	43

LISTA DE QUADROS

Quadro 1 - Disciplinas do Ensino Fundamental.	16
Quadro 2 - Requisito funcional Cadastrar Unidade de Ensino.	28
Quadro 3 - Requisito funcional Cadastrar Usuário	28
Quadro 4 - Requisito funcional Cadastrar Pessoas.....	29
Quadro 5 - Requisito funcional Cadastrar Período Letivo	29
Quadro 6 - Requisito funcional Cadastrar Tipo de Disciplinas	29
Quadro 7 - Requisito funcional Cadastrar Disciplinas	30
Quadro 8 - Requisito funcional Cadastrar Curso.....	30
Quadro 9 - Requisito funcional Cadastrar Série	30
Quadro 10 - Requisito funcional Cadastrar Classe	31
Quadro 11 - Requisito funcional Cadastrar Turno	31
Quadro 12 - Requisito funcional Cadastrar Aluno	31
Quadro 13 - Requisito funcional Cadastrar Professores	32
Quadro 14 - Requisito funcional Cadastrar Matrícula.....	32
Quadro 15 - Requisito funcional Cadastrar Matriz Curricular	32
Quadro 16 - Requisito funcional Lançar Faltas	33
Quadro 17 - Requisito funcional Cadastrar Notas	33
Quadro 18 - Requisito funcional Emitir Boletim por Classe.....	33
Quadro 19 – Requisito funcional Emitir Boletim por Matrícula	34
Quadro 20 - Requisito funcional Emitir relatório de alunos por classe	34
Quadro 21 - Emitir relatório de alunos por curso.....	34
Quadro 22 - Requisito funcional Cadastrar Servidor	35
Quadro 23 - Requisito funcional Fechar Período Letivo	35

LISTAGENS DE CÓDIGO

Listagem 1 - Arquivo de configuração do Hibernate.....	44
Listagem 2 - Classe HDao.....	45
Listagem 3 - Classe HPessoaDao.....	46
Listagem 4 - Classe Pessoa.....	46
Listagem 5 - Remote Object.....	47
Listagem 6 – Classe PessoaService	47
Listagem 7 - Arquivo remoting-config.xml	48

LISTA DE SIGLAS

AMF	<i>Action Message Format</i>
ACID	Acrônimo de Atomicidade, consistência, Isolamento e Durabilidade
ANSI	<i>American National Standards Institute</i>
API	<i>Application Programming Interface</i>
BSD	<i>Berkeley Software Distribution</i>
DDL	<i>Data Definition Language</i>
DER	Diagrama de Entidades e Relacionamentos
DML	<i>Data Manipulation Language</i>
EPL	<i>Eclipse Public License</i>
GNU	<i>GNU Is Not Unix</i>
HTML	<i>Hypertext Markup Language</i>
HTTP	<i>Hyper Text Transfer Protocol</i>
IBM	<i>International Business Machines</i>
IDE	<i>Integrated Development Environment</i>
IDH	Índice de Desenvolvimento Humano
IDS	Indústria de Desenvolvimento de Software
IOS	<i>Iphone Operation System</i>
ISO	<i>International Organization for Standardization</i>
JPA	<i>Application Persistence API</i>
LDBE	Lei de Diretrizes e Bases da Educação
MXML	<i>Magic Extensible Markup Language</i>
ODBC	<i>Open Database Connectivity</i>
PDF	<i>Portable Document Format</i>
PHP	<i>Personal Home Page</i>
RIA	<i>Rich Internet Application</i>
SDK	<i>Software Development Kit</i>
SEQUEL	<i>Structured English Query</i>
SGBD	Sistema Gerenciador de Banco de Dados
SQL	<i>Structured Query Language</i>
SSL	<i>Secure Sockets Layer</i>
SWES	Sistema Web Escolar Simples

TCP/IP	<i>Transmission Control Protocol – Internet Protocol</i>
UML	<i>Unified Modeling Language</i>
XML	<i>Extensible Markup Language</i>

SUMÁRIO

1 INTRODUÇÃO	12
1.1 CONSIDERAÇÕES INICIAIS	12
1.2 OBJETIVOS	13
1.2.1 Objetivo Geral	13
1.2.2 Objetivos Específicos	13
1.3 JUSTIFICATIVA	13
1.4 ESTRUTURA DO TRABALHO	14
2 REFERENCIAL TEÓRICO	15
2.1 PROCESSO COMUM DA EDUCAÇÃO BÁSICA.....	15
2.2 GESTÃO INFORMATIZADA DA EDUCAÇÃO E SEUS PROCESSOS	16
2.3 USO DE SOFTWARE NA GESTÃO EDUCACIONAL.....	17
2.4 SOFTWARES EXISTENTES PARA GESTÃO EDUCACIONAL	17
3.1 MATERIAIS	19
3.1.1 RICH INTERNET APPLICATION	19
3.1.2 PostgreSQL.....	20
3.1.3 SQL	21
3.1.4 Ferramenta Administrativa Pgadmin	21
3.1.5 Ferramenta De Diagramação CaseStudio.....	22
3.1.6 BlazeDS	22
3.1.7 Adobe Flash Builder	23
3.1.8 Flex Sdk	24
3.1.9 IDE Eclipse.....	24
3.1.10 Visual Paradigm for UML.....	24
3.1.11 Hibernate e JPA	25
3.3 MÉTODO.....	25
4 RESULTADOS	27
4.1 ESCOPO DO SISTEMA	27
4.2 MODELAGEM DO SISTEMA	27
4.2.1 Requisitos Funcionais e Não-Funcionais	28
4.2.2 Diagrama de casos de uso.....	35
4.2.3 Diagrama de classe.....	36
4.2.4 Diagrama de entidade e relacionamento.....	37
4.3 APRESENTAÇÃO DO SISTEMA	39
4.4 IMPLEMENTAÇÃO DO SISTEMA	43
5 CONCLUSÕES	49
REFERÊNCIAS	50

1 INTRODUÇÃO

Neste capítulo serão apresentadas as considerações iniciais sobre o trabalho, bem como seus objetivos, a justificativa e a estrutura do texto.

1.1. CONSIDERAÇÕES INICIAIS

Com a constante inclusão digital a informação ficou mais acessível à população. A evolução da tecnologia é um dos principais fatores desencadeadores de tal acontecimento, *smartphones* a preços acessíveis, pontos de internet sem fio liberados nas grandes cidades, estabelecimentos comerciais e órgãos públicos facilitando, cada vez mais, o acesso dos usuários à grande rede mundial de computadores.

Essa superabundância da tecnologia pode ser um poderoso instrumento para informatização da educação e seus respectivos processos nas escolas públicas. Ainda há em nosso País, municípios com baixo IDH (Índice de Desenvolvimento Humano) (ATLAS BRASIL, 2014) e, conseqüentemente, com baixo poder aquisitivo, sendo assim inviável para estes municípios contratar um sistema de gestão educacional para as suas escolas. As tecnologias código aberto existentes são uma boa solução para proporcionar a tais municípios uma informatização na sua gestão educacional com custo baixo.

Como resultado deste trabalho, foram utilizadas tecnologias de código aberto, para desenvolver um software para auxiliar no gerenciamento de uma escola. O software poderá melhor organizar os processos realizados em uma escola básica bem como informatizar pais, professores e alunos contribuindo para o desenvolvimento da educação e da inclusão tecnológica dos envolvidos.

1.2. OBJETIVOS

O objetivo principal está relacionado a análise e desenvolvimento de um software para controle de matrículas e emissão de boletim escolar, os objetivos específicos complementam o objetivo geral.

1.2.1. Objetivo Geral

Analisar e desenvolver um sistema Web para controle de matrículas e emissão de boletins escolares.

1.2.2. Objetivos Específicos

- Oferecer ao usuário uma solução de software em padrão adequado de usabilidade e interação.
- Atender as necessidades de escolas no processo de matrículas, fechamento de período letivo e suas particularidades.
- Facilitar o processo de emissão de boletim escolar.
- Disponibilizar uma alternativa de baixo custo a municípios de baixo IDH (Índice de Desenvolvimento Humano).

1.3. JUSTIFICATIVA

A justificativa para realização desse projeto é embasada na necessidade crescente das instituições de ensino público informatizar e melhor organizar seus processos e informações. Ainda hoje em algumas escolas do Brasil, o registro de notas e frequências dos alunos é realizado em cadernos de anotações e todo o processo de cálculo de médias e frequências é realizado manualmente.

As tecnologias necessárias para desenvolvimento do sistema proposto são todas de código aberto não gerando custo ao município, exceto o de

infraestrutura e equipamentos. O objetivo desta ferramenta é auxiliar municípios de baixo IDH a informatizar a gestão escolar, tendo em vista que, segundo a Lei 9394 de 1996 (BRASIL, 2014), cabe ao município oferecer a educação básica.

Para que esse projeto seja desenvolvido, serão utilizados os conhecimentos adquiridos no curso de Tecnologia em Análise e Desenvolvimento de Sistemas da UTFPR, Câmpus Pato Branco, principalmente relacionados ao emprego da linguagem de programação Java trabalhando em conjunto com Flex (FLEX, 2014).

1.4 ESTRUTURA DO TRABALHO

O texto deste trabalho está organizado em capítulos. O primeiro capítulo apresenta a ideia do sistema, incluindo as considerações iniciais, os objetivos e a justificativa.

O capítulo 2 contém o referencial teórico que serviu como base para a proposta conceitual do sistema desenvolvido. O referencial teórico se concentra em processos desenvolvidos em escolas de ensino básico, softwares existentes para uso em escolas e como as tecnologias Web possibilitam a informatização da administração do ensino.

No capítulo 3 estão os materiais e o método utilizados. Os materiais se referem ao que é necessário para modelar e implementar o sistema, incluindo as tecnologias, as ferramentas e os ambientes de desenvolvimento utilizados. O método se refere aos procedimentos utilizados no ciclo de vida do sistema, abrangendo da definição dos requisitos à implementação do sistema.

O capítulo 4 contém o sistema desenvolvido, com a modelagem produzida e as telas de codificação do sistema. Neste capítulo também está incluído o uso das tecnologias empregadas para desenvolver o sistema.

No capítulo 5 está a conclusão com as considerações finais. Por fim estão as referências bibliográficas.

2 REFERENCIAL TEÓRICO

Este capítulo apresenta o referencial teórico que serviu como base para a proposta conceitual do sistema desenvolvido. O referencial teórico se concentra em processos desenvolvidos em escolas de ensino básico e softwares existentes para uso em escolas.

2.1 PROCESSO COMUM DA EDUCAÇÃO BÁSICA

A educação básica no Brasil, segundo a Lei de Diretrizes e Bases da Educação de 1996, Título V dos Níveis e das Modalidades de Educação e Ensino, Capítulo 1, Artigo 21, Parágrafo I, está dividida em educação infantil, ensino fundamental e ensino médio.

A lei define que o ensino básico, nas modalidades fundamental e médio seja organizado conforme regras comuns específicas. Para o desenvolvimento deste sistema, foram consideradas as principais regras, referentes a: carga horária mínima anual de oitocentas horas distribuídas por um mínimo de duzentos dias de efetivo trabalho escolar excluído o tempo destinado para exames finais; controle de frequência, a cargo da escola, que exige o mínimo de setenta e cinco por cento de frequência do total de horas letivas. A grade curricular do ensino fundamental e médio deverá conter as disciplinas que atendem a Lei de Diretrizes e Bases da Educação (LDBE) podendo ser complementada por uma disciplina classificada como diversificada, ficando a escolha desta disciplina a critério da unidade de ensino.

Para tanto se faz necessário que a grade curricular contenha obrigatoriamente as disciplinas de base nacional comum conforme a LDBE (BRASIL, 2014), abrangendo desta maneira o estudo da língua portuguesa e da matemática, o conhecimento do mundo físico e natural, o conhecimento da realidade social e política, especialmente do Brasil, o ensino da arte, especialmente em suas expressões regionais e a educação física, o ensino religioso é de matrícula facultativa. A esta grade curricular, obrigatoriamente, deve ser adicionada uma parte diversificada a partir da quinta série, com o ensino de pelo menos uma língua estrangeira moderna, cuja escolha ficará a cargo da comunidade escolar, dentro das possibilidades da instituição. Sendo

assim, o Quadro 1 apresenta as disciplinas do ensino fundamental consideradas neste trabalho:

Disciplina	Categoria
Português	Base Nacional Comum
Matemática	Base Nacional Comum
Historia	Base Nacional Comum
Geografia	Base Nacional Comum
Ciências	Base Nacional Comum
Educação Física	Base Nacional Comum
Artes	Base Nacional Comum

Quadro 1 - Disciplinas do Ensino Fundamental.

2.2 GESTÃO INFORMATIZADA DA EDUCAÇÃO E SEUS PROCESSOS

Por traz das salas de aula existe todo um processo administrativo para gerir o bom funcionamento de uma instituição de ensino, seja ela pública ou privada. O objetivo deste trabalho é justamente informatizar e agilizar os principais processos que são realizados pelos agentes educacionais nas instituições.

Procedimentos como a definição de grade curricular, emissão de boletins e controle de notas e frequências são alguns dos principais processos administrativos de uma instituição de ensino, e para defini-los, geralmente se mobiliza grande parte dos agentes educacionais da instituição de ensino. Muitas vezes o controle de notas, frequências e anotações são feitos manualmente nos populares “Diário de Classe” e alimentadas manualmente no decorrer do período letivo, estando tais informações sujeitas a serem prejudicadas devido a degradação do tempo e manuseio frequente do diário, o que pode futuramente, no fechamento do período resultar em redução da confiabilidade das informações e gerar retrabalho.

Informatizar tais procedimentos garantem a integridade e longevidade da informação, diminuem tempo de processos e procedimentos, contribuem possibilitando a integração com sistemas do governo. Por exemplo, o Educacenso, é um sistema on-line que visa coletar, organizar, transmitir e disseminar os dados censitários, possibilitando até a interação entre pais e escolas, onde, em um ambiente Web os pais dos alunos podem acompanhar o

desempenho de seus filhos propiciando assim a inclusão digital fora dos muros das instituições (EDUCACENSO, 2014).

2.3 USO DE SOFTWARE NA GESTÃO EDUCACIONAL

No Brasil, o início da informática na educação deu seus primeiros passos na década de 80, por meio do projeto Educom que tinha por objetivo principal desenvolver pesquisas interdisciplinares sobre a aplicação da informática no processo de ensino-aprendizagem (TAVARES, 2007).

A Universidade Federal do Rio de Janeiro foi a instituição pioneira no uso do computador em atividades acadêmicas no núcleo de cálculo científico (PROFUNCIONARIO, 2007).

Ainda na década de 80 também a Internet ganhou sua forma mais robusta, criando assim possibilidades que se consolidaram nos dias de hoje. Diante de tal evolução vários tipos de soluções informatizadas foram criadas e difundidas, empresas de sistemas de diversos segmentos expandiram-se rapidamente. Soluções para gestão pública municipal ganharam um grande espaço no mercado, tais softwares auxiliam nas tomadas de decisões e permitem a centralização de todos os dados relacionados às áreas administrativa, acadêmica e pedagógica dos municípios que os detêm.

2.4 SOFTWARES EXISTENTES PARA GESTÃO EDUCACIONAL

Dentre os softwares livres se destaca o i-Educar do governo federal em parceria com a empresa Portabilis (SOFTWARE PUBLICO, 2014), que foi desenvolvido originalmente pela Prefeitura de Itajaí-SC em linguagem PHP (*Personal Home Page*) e disponibilizado como Software Livre no Portal do Software Público Brasileiro (<http://www.softwarepublico.gov.br/>), sendo atualmente mantido pela Comunidade i-Educar. O software pode ser obtido gratuitamente, sendo necessário apenas se cadastrar no site do portal de software público.

Como exemplo, uma empresa privada que é referência no cenário nacional em soluções para gestão municipal, é a IDS (Indústria de Desenvolvimento de Software) Software e Assessoria, pois possui o software IDS Educação, desenvolvido em tecnologia Java/Flex totalmente Web. Dentre as funcionalidades do sistema se destacam o portal do aluno, onde pais podem acompanhar o desempenho dos alunos na escola e a estrutura do software que pode ser facilmente adaptável às necessidades de cada município, tornando-se uma ferramenta poderosa associada aos indicadores que a empresa fornece.

3 MATERIAIS E MÉTODO

Este capítulo apresenta as ferramentas, as tecnologias e o método utilizados para a modelagem e a implementação do sistema. Os materiais se referem às ferramentas e às tecnologias utilizadas, como linguagem de programação, banco de dados, interface de desenvolvimento e editores para a modelagem. O método contém as principais atividades realizadas para o desenvolvimento do trabalho.

3.1 MATERIAIS

Para o desenvolvimento do sistema foram utilizadas como IDE as ferramentas Adobe Flash Builder e Eclipse IDE (*Integrated Development Environment*), como frameworks o BlazeDS e Hibernate, e para banco de dados foi escolhido o PostgreSQL. As ferramentas Draw.io, Case Studio e Visual Paradigm for UML (*Unified Modeling Language*) foram utilizadas para a criação de diagramas de classes e diagramas de caso de uso e o pgAdmin como ferramenta administrativa para o banco de dados PostgreSQL. Para prover maior usabilidade ao sistema desenvolvido, foi empregado o modelo RIA (Rich Internet Application).

3.1.1 RICH INTERNET APPLICATION

Na década de 90 foi criado um novo modelo de aplicações voltadas para a Internet. O modelo é baseado em *Hypertext Markup Language* (HTML), segundo Allaire (2002). Nesse modelo, o usuário, por meio da página no navegador Web faz requisições aos servidores, onde estas informações são processadas, validadas e devolvidas ao usuário com todas as informações atualizadas. O termo RIA foi introduzido em 2002 pela Macromedia, este módulo busca disponibilizar as funcionalidades e facilidades de uso de uma aplicação *desktop* na Web juntamente com a interatividade proporcionada pela tecnologia que permite desenvolver interfaces de fácil utilização.

Segundo Guanais (2010) RIA é um novo tipo de aplicação *Web* com o objetivo de incrementar e melhorar as opções e capacidades das aplicações *Web* tradicionais. Dentre as vantagens da utilização da RIA destaca-se a riqueza da interface oferecida ao usuário, possibilitando uma interatividade similar ao ambiente desktop, e somente o processamento necessário será enviado ao servidor possibilitando assim requisição de múltiplos usuários e garantindo o desempenho satisfatório da aplicação (DUHL, 2003).

3.1.2 PostgreSQL

PostgreSQL é um poderoso sistema gerenciador de banco de dados objeto-relacional de código aberto (POSTGRESQL, 2014). Pode ser considerada como uma ferramenta madura, pois tem mais de 15 anos de desenvolvimento ativo e uma arquitetura que comprovadamente ganhou forte reputação de confiabilidade, integridade de dados e conformidade a padrões.

O PostgreSQL roda em todos os grandes sistemas operacionais, incluindo GNU (*GNU Is not Unix*), Linux, Unix e Microsoft Windows. É totalmente compatível com ACID (Acrônimo de Atomicidade, consistência, Isolamento e Durabilidade), tem suporte completo a chaves estrangeiras, junções (JOINS), visões, gatilhos e procedimentos armazenados (em múltiplas linguagens). Inclui a maior parte dos tipos de dados do ISO SQL:1999, incluindo INTEGER, NUMERIC, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL, e TIMESTAMP. Suporta também o armazenamento de objetos binários, incluindo figuras, sons ou vídeos. Possui interfaces nativas de programação para C/C++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC (*Open Database Conectivity*), entre outros, e uma excepcional documentação (POSTGRESQL, 2014).

Segundo Biazus (2003), pela riqueza de recursos e conformidade com os padrões, o PostgreSQL é um SGBD (Sistema Gerenciador de Banco de Dados) muito adequado para o estudo universitário do modelo relacional, além de ser uma ótima opção para empresas implementarem soluções de alta confiabilidade sem altos custos de licenciamento.

“É um programa distribuído sob a licença BSD (*Berkeley Software Distribution*), o que torna o seu código fonte disponível e o seu uso livre para aplicações comerciais ou não. O PostgreSQL foi

implementado em diversos ambientes de produção no mundo”(BIAZUS, 2003, p.1).

3.1.3 SQL

Originalmente chamado SEQUEL (*Structured English QUery*) e inspirado na álgebra relacional, o SQL (*Structured Query Language*, ou Linguagem de Consulta Estruturada), originou-se no início dos anos 70 nos laboratórios da IBM (*International Business Machines*), dentro de um projeto chamado System R.

Embora o SQL tenha sido originalmente criado pela IBM, rapidamente surgiram vários "dialetos" desenvolvidos por outros produtores. Essa expansão levou à necessidade de ser criado e adaptado um padrão para a linguagem. Esta tarefa foi realizada pela *American National Standards Institute* (ANSI) em 1986 e ISO (*International Organization for Standardization*) em 1987.

Os recursos da linguagem SQL podem ser divididos, principalmente, em duas partes:

- a) Linguagem de definição de dados (DDL - *Data Definition Language*): comandos para a definição de esquemas de relações, exclusão de relações, modificação nos esquemas de relações e criação de domínios.
- b) Linguagem interativa de manipulação de dados (DML - *Data Manipulation Language*): abrange uma linguagem de consulta baseada tanto na álgebra relacional quanto no cálculo relacional de tuplas. Engloba também comandos para inserção, exclusão e modificação de tuplas no banco de dados.

3.1.4 Ferramenta Administrativa Pgadmin

O software pgAdmin é uma ferramenta de banco de dados de código aberto bastante popular e avançada para desenvolvimento e administração em PostgreSQL.

O pgAdmin pode ser usado em Linux, FreeBSD, Solaris, Mac OSX e Windows.

“pgAdmin é projetado para atender às necessidades de todos os usuários, desde escrever simples consultas SQL até o desenvolvimento de bancos de dados complexos. A interface gráfica suporta todas as funcionalidades do PostgreSQL e facilita sua administração. O aplicativo também inclui um editor com destaque de sintaxe SQL / batch / shell, agente de agendamento de trabalho, o apoio à Slony-I, mecanismo de replicação e muito mais. A conexão com o servidor pode ser feita usando TCP / IP (*Transmission Control Protocol – Internet Protocol*) ou soquetes do domínio Unix (em * nix), e pode ser criptografada em SSL (*Secure Sockets Layer*) para um maior nível de segurança, drivers adicionais não são necessários para se comunicar com o servidor de banco de dados (PGADMIN, 2013, p.1).

Software livre (sob licença PostgreSQL), desenvolvido por uma comunidade de especialistas do PostgreSQL em todo o mundo, o pgAdmin está disponível em mais de uma dúzia de idiomas.

3.1.5 Ferramenta De Diagramação CaseStudio

O CaseStudio é uma ferramenta voltada para o desenvolvimento de diagramas de entidade e relacionamento e diagramas de fluxo de dados, permitindo também gerar scripts para criação de tabelas e banco de dados.

O CaseStudio tem suporte aos principais bancos de dados como, SQL Server, PostgreSQL, Firebird, entre outros. Com uma interface simples e amigável o CaseStudio facilita e auxilia muito o usuário a modelar e manter a integridade de sua base de dados.

3.1.6 BlazeDS

BlazeDS é um framework de código aberto mantido pela Adobe, que possibilita chamadas de serviços remotos e trabalha com base no protocolo HTTP (*Hyper Text Transfer Protocol*), utilizando como base o formato de dados binário AMF (*Action Message Format*) para serializar e deserializar dados, o que torna a comunicação com o servidor muito otimizada e é amplamente utilizada na tecnologia Flex. Abaixo a Figura 1 demonstra o a integração do framework com Flex e Java.



Figura 1 - Integração do framework BlaszeDS

3.1.7 Adobe Flash Builder

Adobe Flash Builder é um ambiente de desenvolvimento integrado construído sobre a plataforma Eclipse, originalmente desenvolvido pela Macromedia. Antes da versão 4, este produto era conhecido como Flex Builder. A mudança de nome foi com a intenção de demonstrar sua interação com outros produtos da Adobe.

Essa IDE tem por objetivo otimizar o desenvolvimento de aplicações RIA e aplicativos desktop multi-plataforma, especialmente para a plataforma Adobe Flash.

Adobe Flash Builder 4 está disponível em três edições: Standard, Premium e Educations. O pacote está disponível gratuitamente para uso não comercial na condição de teste grátis por 30 dias e para estudantes. A partir da versão 4.6 a interface visual do Adobe Flash Builder foi removida da IDE, a versão atual é a 4.7 e neste trabalho será utilizada a versão 4.5 da ferramenta juntamente com o Flex SDK (*Software Development Kit*) 4.5.

3.1.8 Flex Sdk

Lançado em 2004 e Inicialmente conhecido como Macromedia Flex, em 2005 foi adquirido pela Adobe e passou a ser chamado de Adobe Flex. Para uso desse produto era necessário licença para utilização, em 2007 seu código fonte foi aberto, em 2011 foi doado a Apache pela Adobe e é mantida como projeto de alto nível.

Flex é um poderoso framework/compilador que permite o desenvolvimento de aplicações tanto para dispositivos móveis, contemplando iOS (*Iphone Operation System*), Android e BlackBerry, quanto para aplicações desktop e principalmente Web. Possibilita desenvolver interfaces intuitivas e possui um grande número de componentes e uma ativa comunidade de desenvolvedores. É liberado sob a Licença Apache, atualmente está na versão 4.12 (Março/2014).

Flex usa MXML para criação e manutenção de layout da interface e outros aspectos estáticos não-visuais, ActionScript é usado para tratar de aspectos dinâmicos e regras de negócio, tendo como alvo aplicações baseadas principalmente em Flash.

3.1.9 IDE Eclipse

Eclipse é uma IDE para desenvolvimento java mantida pela Eclipse Foundation, foi desenvolvida em java e tem suporte a várias outras linguagens, por exemplo, PHP, PYTHON, C++, entre outras (PALMEIRA, 2014). A primeira versão do produto foi desenvolvida pela IBM, a qual investiu 40 milhões de dólares no projeto (CNET, 2014) e posteriormente foi doado como software livre para a comunidade. O Eclipse conta com inúmeros plug-ins para desenvolvimento e é mantido sob a licença EPL (*Eclipse Public License*), atualmente está na versão Luna 4.4.

3.1.10 Visual Paradigm for UML

Visual Paradigm for UML é uma ferramenta CASE com várias opções de modelagem. A ferramenta possui um ambiente de trabalho que facilita a visualização e a manipulação do projeto de modelagem.

A ferramenta Visual Paradigm, além de modelar os diversos tipos de diagramas ainda gera automaticamente outros diagramas por meio da análise dos casos, por exemplo: um Diagrama de Entidades e Relacionamentos (DER) a partir de um diagrama de classe e depois gera o código SQL a partir do DER. Ainda é capaz de gerar código PHP, Java, C, C++, C# entre outros (VISUALPARADIGM, 2012). O Visual Paradigm for UML foi utilizado nesse projeto para criação dos diagramas de classes e diagrama de casos de uso.

3.1.11 Hibernate e JPA

Hibernate é um framework escrito na linguagem java e disponível para .NET com o nome NHibernate. Com o Hibernate é possível mapear os atributos de objetos de uma aplicação e relacioná-los com uma base de dados relacional mediante o uso de arquivos XML (*Extensible Markup Language*) ou anotações.

JPA ou Java Persistence API é uma API (*Application Programming Interface*) em linguagem java para persistência de dados, esta API facilita o uso do Hibernate através de anotações nos objetos, poupando grande parte da configuração e mapeamento destes objetos nos arquivos de configuração do Hibernate (MEDEIROS, 2014).

3.3 MÉTODO

As etapas desenvolvidas para modelagem e implementação do sistema para controle de matrículas e emissão de boletim escolar seguiram o modelo sequencial linear proposto por Pressman (2011). As etapas definidas foram:

- a) Levantamento dos requisitos

Esse sistema surge da experiência do acadêmico com sistemas deste ramo de atuação. O estágio do acadêmico foi realizado em uma empresa

focada no desenvolvimento de softwares diversos, entre eles sistemas para gestão escolar. Assim, a definição dos requisitos foi realizada com base no conhecimento do acadêmico e observação dos clientes atendidos pela empresa.

b) Análise e projeto

Os requisitos definidos foram modelados sob a forma de casos de uso, diagrama de classes e diagrama de entidades e relacionamentos do banco de dados.

c) Implementação

A implementação foi realizada utilizando o ambiente IDE Eclipse para linguagem java, o framework Adobe Flash Builder para utilizar a tecnologia Flex. BlazeDS para comunicação entre ambos e pgAdmin como administrador de base de dados para PostgreSQL.

d) Testes

Os testes foram realizados pelo próprio acadêmico e tiveram o objetivo de verificar erros de código e a interação com o sistema.

4 RESULTADOS

Este capítulo apresenta o que foi obtido como resultado do trabalho, ou seja, o sistema modelado e desenvolvido.

4.1 ESCOPO DO SISTEMA

O sistema desenvolvido foi nomeado como SWES, as quatro letras no nome significam Sistema Web Escolar Simples. O SWES atende aos procedimentos básicos realizados em uma unidade de ensino funcionando em uma plataforma de acesso Web.

O SWES possui um controle de acessos, onde o usuário informa seu login e senha. Este acesso de usuário está dividido em três categorias: professores, servidores e administradores. Cada tipo de usuário tem acesso às suas áreas de competência.

O sistema possui os cadastros das informações necessárias para o exercício de período letivo, sendo eles, cadastro de pessoas, unidade escolar, período letivo, matrícula, turma, série, curso, lançamento de notas e matriz curricular. Alguns procedimentos são automatizados como o fechamento do período letivo e geração de boletins.

O SWES permite a emissão de relatórios como boletins, alunos por classe, cursos e demais relatórios necessários.

4.2 MODELAGEM DO SISTEMA

A seguir são apresentados os requisitos definidos para o sistema, os diagramas de casos de uso, de classes e de entidades e relacionamentos gerados.

4.2.1 Requisitos Funcionais e Não-Funcionais

Requisitos funcionais descrevem explicitamente as funcionalidades e serviços do sistema, requisitos não funcionais definem propriedades e restrições no sistema (SOMMERVILE, 2003).

Os principais requisitos funcionais definidos para o sistema são apresentados nos Quadros de 2 a 22, destacando os requisitos não-funcionais associados.

O Quadro 2 apresenta o requisito funcional Cadastrar Unidade de Ensino e os requisitos não funcionais relacionados.

F1 Cadastrar Unidade de Ensino		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de cadastrar a unidade de ensino e todas suas informações necessárias, sendo possível editar, criar e excluir cadastros.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF1.1 Restrição de Usuário	Somente usuário do tipo Administrador poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 2 - Requisito funcional Cadastrar Unidade de Ensino.

O Quadro 3 apresenta o requisito funcional Cadastrar Usuário e os requisitos não funcionais relacionados.

F2 Cadastrar Usuário		Oculto ()		
Descrição: O sistema deve oferecer ao usuário administrador a possibilidade de cadastrar os usuários que acessam o sistema, sendo possível editar, criar e excluir cadastros.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF2.1 Restrição de Usuário	Somente usuário do tipo Administrador poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 3 - Requisito funcional Cadastrar Usuário

O Quadro 4 apresenta o requisito funcional Cadastrar Pessoas e os requisitos não funcionais relacionados.

F3 Cadastrar Pessoas		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de cadastrar os pessoas no sistema, sendo possível editar, criar e excluir cadastros.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF3.1 Restrição de Usuário	Somente usuário do tipo Administrador e Servidor poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 4 - Requisito funcional Cadastrar Pessoas

O Quadro 5 apresenta o requisito funcional Cadastrar Período Letivo e os requisitos não funcionais relacionados.

F4 Cadastrar Período Letivo		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de cadastrar períodos letivos no sistema, sendo possível editar e criar cadastros.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF4.1 Restrição de Usuário	Somente usuário do tipo Administrador e Servidor poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 5 - Requisito funcional Cadastrar Período Letivo

O Quadro 6 apresenta o requisito funcional Cadastrar Tipo de Disciplinas e os requisitos não funcionais relacionados.

F5 Cadastrar Tipo de Disciplinas		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de cadastrar o tipo das disciplinas no sistema, sendo possível editar, excluir e criar cadastros.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF5.1 Restrição de Usuário	Somente usuário do tipo Administrador e Servidor poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 6 - Requisito funcional Cadastrar Tipo de Disciplinas

O Quadro 7 apresenta o requisito funcional Cadastrar Disciplinas e os requisitos não funcionais relacionados.

F6 Cadastrar Disciplinas		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de cadastrar disciplinas no sistema, sendo possível editar e excluir.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF6.1 Restrição de Usuário	Somente usuário do tipo Administrador e Servidor poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 7 - Requisito funcional Cadastrar Disciplinas

O Quadro 8 apresenta o requisito funcional Cadastrar Curso e os requisitos não funcionais relacionados.

F7 Cadastrar Curso		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de cadastrar os cursos no sistema, sendo possível editar, criar e excluir cadastros.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF7.1 Restrição de Usuário	Somente usuário do tipo Administrador e Servidor poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 8 - Requisito funcional Cadastrar Curso

O Quadro 9 apresenta o requisito funcional Cadastrar Série e os requisitos não funcionais relacionados.

F8 Cadastrar Série		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de cadastrar as series no sistema, sendo possível editar, criar e excluir cadastros.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF8.1 Restrição de Usuário	Somente usuário do tipo Administrador e Servidor poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 9 - Requisito funcional Cadastrar Série

O Quadro 10 apresenta o requisito funcional Cadastrar Classe e os requisitos não funcionais relacionados.

F9 Cadastrar Classe		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de cadastrar as classes no sistema, sendo possível editar, criar e excluir cadastros.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF9.1 Restrição de Usuário	Somente usuário do tipo Administrador e Servidor poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 10 - Requisito funcional Cadastrar Classe

O Quadro 11 apresenta o requisito funcional Cadastrar Turno e os requisitos não funcionais relacionados.

F10 Cadastrar Turno		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de cadastrar os turnos no sistema, sendo possível editar, criar e excluir cadastros.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF10.1 Restrição de Usuário	Somente usuário do tipo Administrador e Servidor poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 11 - Requisito funcional Cadastrar Turno

O Quadro 12 apresenta o requisito funcional Cadastrar Alunos e os requisitos não funcionais relacionados.

F11 Cadastrar Alunos		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de cadastrar os alunos no sistema, sendo possível editar, criar e excluir cadastros.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF11.1 Restrição de Usuário	Somente usuário do tipo Administrador e Servidor poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 12 - Requisito funcional Cadastrar Aluno

O Quadro 13 apresenta o requisito funcional Cadastrar Professores e os requisitos não funcionais relacionados.

F12 Cadastrar Professores		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de cadastrar os professores no sistema, sendo possível editar, criar e excluir cadastros.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF12.1 Restrição de Usuário	Somente usuário do tipo Administrador e Servidor poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 13 - Requisito funcional Cadastrar Professores

O Quadro 14 apresenta o requisito funcional Cadastrar Matrícula e os requisitos não funcionais relacionados.

F13 Cadastrar Matrícula		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de cadastrar a matrícula do aluno no sistema, sendo possível editar, criar e excluir cadastros.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF13.1 Restrição de Usuário	Somente usuário do tipo Administrador e Servidor poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 14 - Requisito funcional Cadastrar Matrícula

O Quadro 15 apresenta o requisito funcional Cadastrar Matriz Curricular e os requisitos não funcionais relacionados.

F14 Cadastrar Matriz Curricular		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de cadastrar a matriz curricular para cada curso e série sendo possível editar, criar e excluir cadastros.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF14.1 Restrição de Usuário	Somente usuário do tipo Administrador e Servidor poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 15 - Requisito funcional Cadastrar Matriz Curricular

O Quadro 16 apresenta o requisito funcional Lançar Faltas e os requisitos não funcionais relacionados.

F15 Lançar Faltas		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de efetuar o lançamento de faltas por classe, possibilitando a inclusão, exclusão e alteração.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF15.1 Restrição de Usuário	Todos os tipos de usuário.	Segurança	(x)	(x)

Quadro 16 - Requisito funcional Lançar Faltas

O Quadro 17 apresenta o requisito funcional Cadastro de Notas e os requisitos não funcionais relacionados.

F16 Cadastro de Notas		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de cadastrar as notas do aluno em suas respectivas disciplinas matriculadas, possibilitando editar e/ou excluir.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF16.1 Restrição de Usuário	Todos os tipos de usuário.	Segurança	(x)	(x)

Quadro 17 - Requisito funcional Cadastrar Notas

O Quadro 18 apresenta o requisito funcional Emitir Boletim por Classe e os requisitos não funcionais relacionados.

F17 Emitir Boletim por Classe		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de imprimir os boletins por classe				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF17.1 Restrição de Usuário	Somente usuário do tipo Administrador e Servidor poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 18 - Requisito funcional Emitir Boletim por Classe

O Quadro 19 apresenta o requisito funcional Emitir Boletim Matrícula e os requisitos não funcionais relacionados.

F18 Emitir Boletim Matrícula		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de imprimir os boletins por matrícula.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF18.1 Restrição de Usuário	Somente usuário do tipo Administrador e Servidor poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 19 – Requisito funcional Emitir Boletim por Matrícula

O Quadro 20 apresenta o requisito funcional Emitir Relatório de Alunos por Classe e os requisitos não funcionais relacionados.

F19 Emitir Relatório de Alunos por Classe		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de emitir um relatório de alunos por classe.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF19.1 Restrição de Usuário	Somente usuário do tipo Administrador e Servidor poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 20 - Requisito funcional Emitir relatório de alunos por classe

O Quadro 21 apresenta o requisito funcional Emitir Relatório de Alunos por Curso e os requisitos não funcionais relacionados.

F20 Emitir relatório de alunos por curso		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de emitir um relatório de alunos por curso.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF20.1 Restrição de Usuário	Somente usuário do tipo Administrador e Servidor poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 21 - Emitir relatório de alunos por curso

O Quadro 22 apresenta o requisito funcional Cadastrar Servidor e os requisitos não funcionais relacionados.

F21 Cadastrar Servidor		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade cadastrar os servidores da unidade de ensino, sendo possível editar e excluir.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF21.1 Restrição de Usuário	Somente usuário do tipo Administrador e Servidor poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 22 - Requisito funcional Cadastrar Servidor

O Quadro 23 apresenta o requisito funcional Fechar Período Letivo e os requisitos não funcionais relacionados.

F22 Fechar Período Letivo		Oculto ()		
Descrição: O sistema deve oferecer ao usuário a possibilidade de encerrar o ano letivo.				
Requisitos Não-Funcionais				
Nome	Restrição	Categoria	Desejável	Permanente
NF22.1 Restrição de Usuário	Somente usuário do tipo Administrador e Servidor poderá ter acesso ao cadastro.	Segurança	(x)	(x)

Quadro 23 - Requisito funcional Fechar Período Letivo

4.2.2 Diagrama de casos de uso

Os principais requisitos funcionais, já identificados anteriormente, foram representados sob a forma de um diagrama de casos de uso, apresentado na Figura 2 abaixo. Este diagrama apresenta as principais funcionalidades do sistema e os atores definidos.

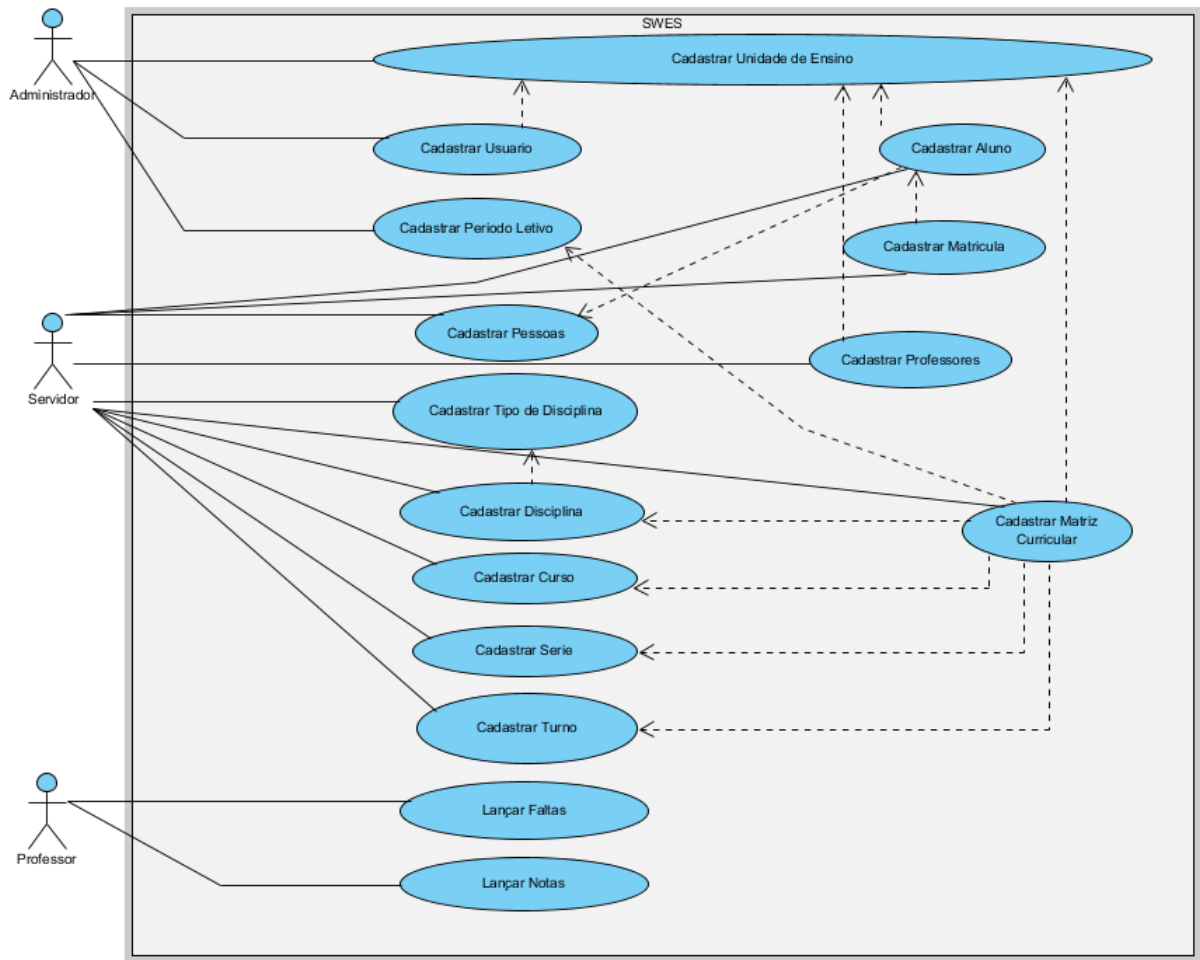


Figura 2 - Diagrama de casos de uso do sistema desenvolvido.

4.2.3 Diagrama de classe

A Figura 3 apresenta parte do diagrama de classes que compõem o sistema implementado neste projeto. Este diagrama é composto por algumas classes referentes aos dados de matrícula, pessoa, unidade de ensino e matriz curricular.

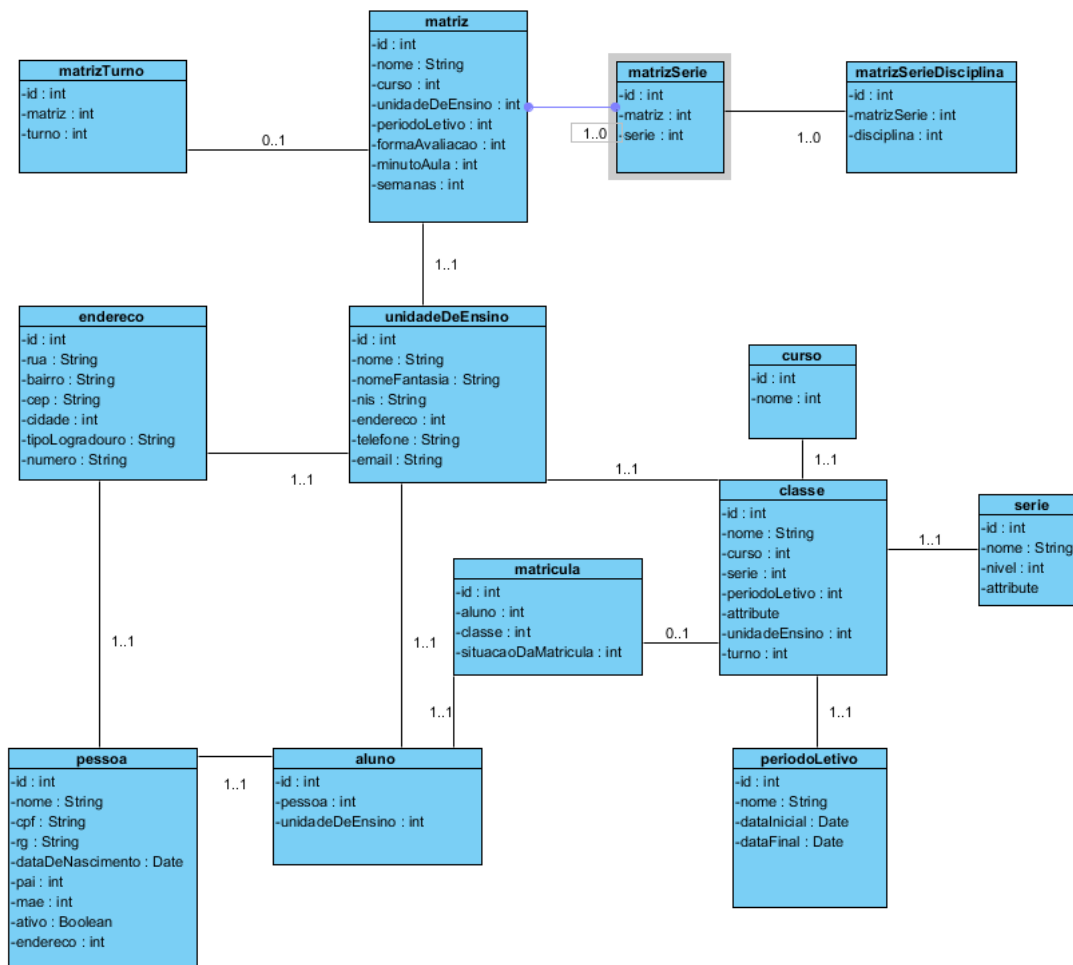


Figura 3 - Diagrama de classes do sistema desenvolvido.

4.2.4 Diagrama de entidade e relacionamento

A Figura 4 apresenta o diagrama de entidades e relacionamentos gerado, apresentando algumas das tabelas necessárias para a execução da matrícula em uma unidade de ensino. Este diagrama também não foi incluído em sua totalidade em função de seu tamanho ser um pouco grande, o que poderia tornar ilegível algumas informações.

[1,1]

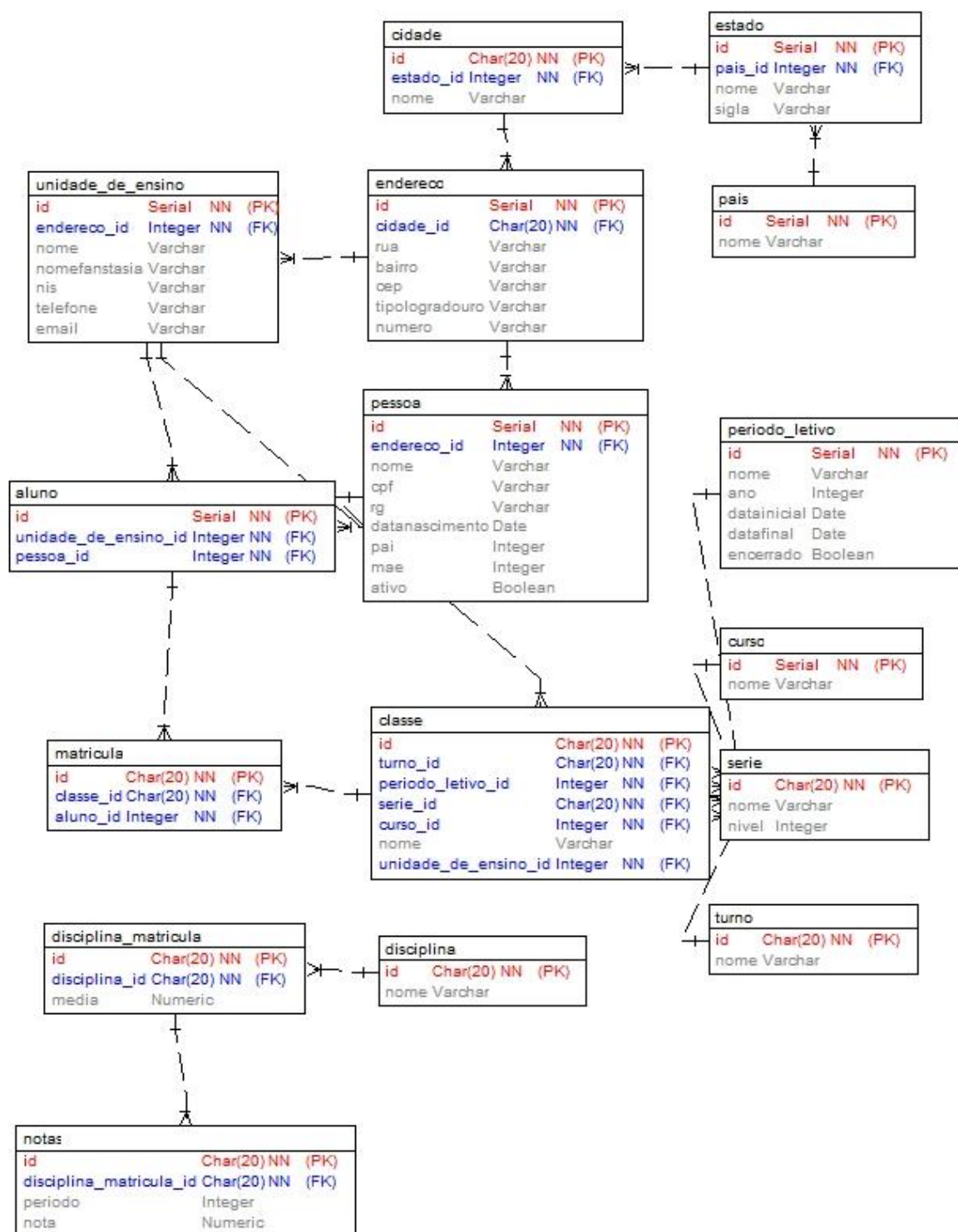


Figura 4 – Diagrama de Entidade e Relacionamento do sistema.

4.3 APRESENTAÇÃO DO SISTEMA

O SWES, desenvolvido como resultado deste trabalho de conclusão de curso visa oferecer uma maneira de auxiliar as escolas na informatização de processos e na organização e execução dos procedimentos administrativos.

Na Figura 5 tem-se a tela de login que é apresentada para o usuário quando o sistema é iniciado. A partir dessa tela o usuário poderá logar no sistema.

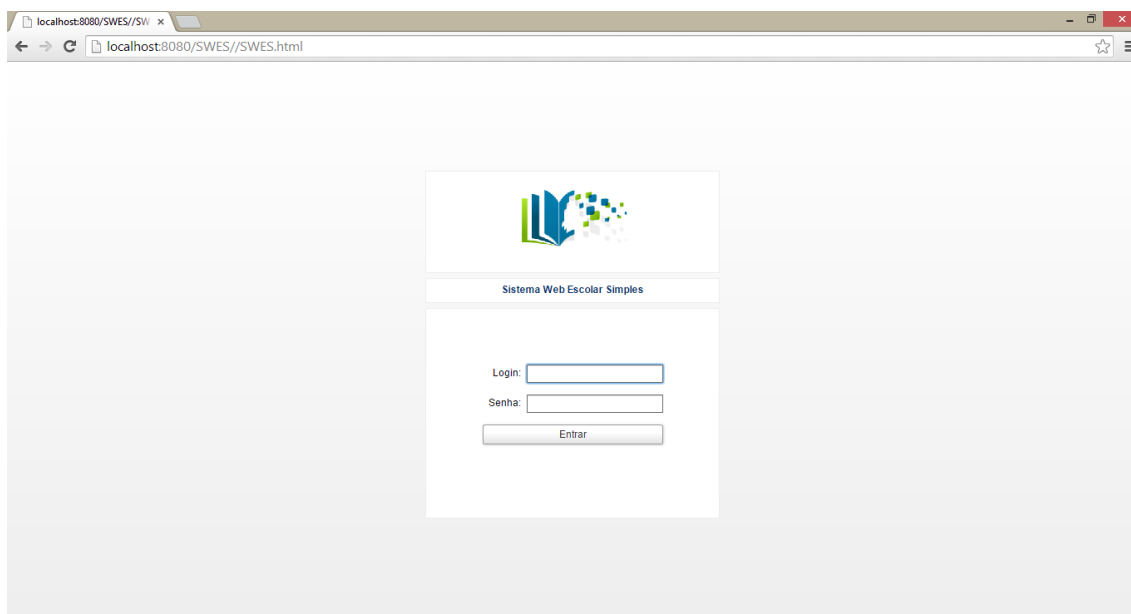


Figura 5 - Tela de login do sistema desenvolvido.

A Figura 6 mostra a página principal do sistema. A partir dessa tela o usuário logado tem acesso a todos os módulos do sistema por meio do menu localizado à esquerda da página. As telas de usuário e de unidade de ensino são acessadas por meio de links disponíveis no canto superior direito ao clicar no nome da unidade de ensino e/ou no usuário que está logado no sistema.

O menu disponível permite acesso instantâneo às principais funcionalidades do sistema. O logotipo da empresa fica localizado no centro da página.

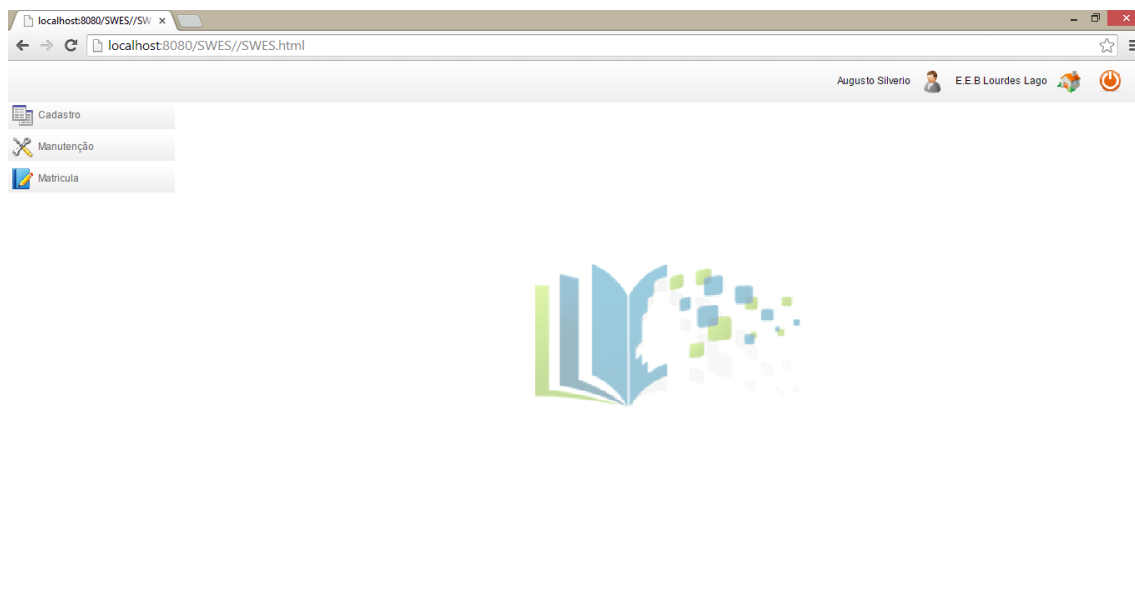
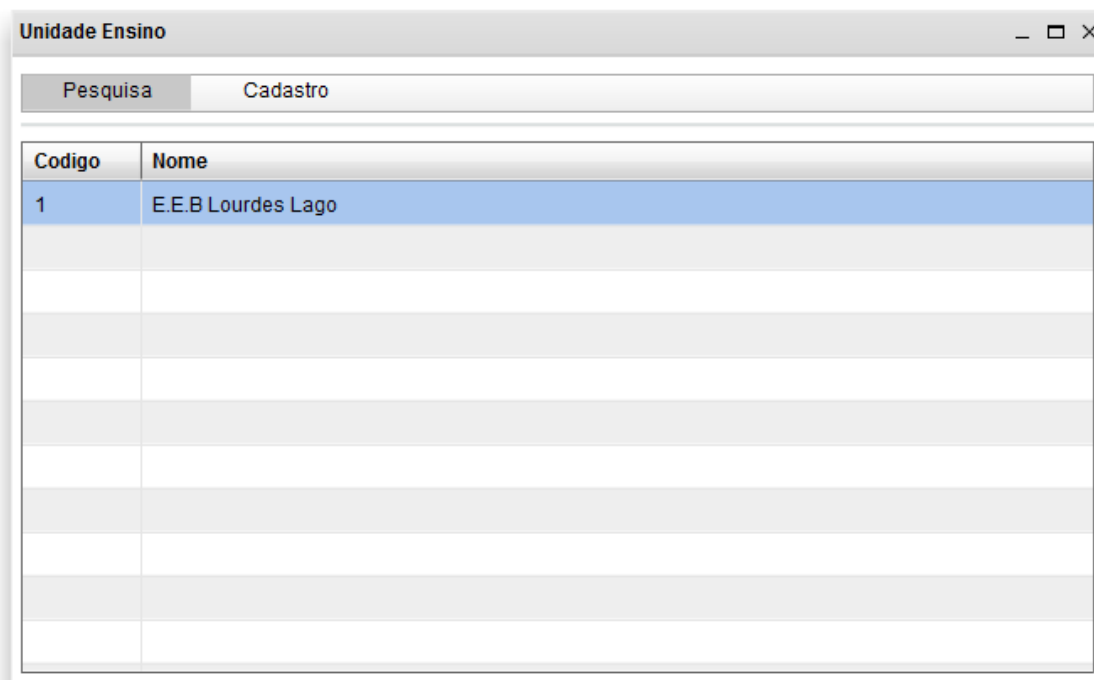


Figura 6 - Página principal do sistema desenvolvido

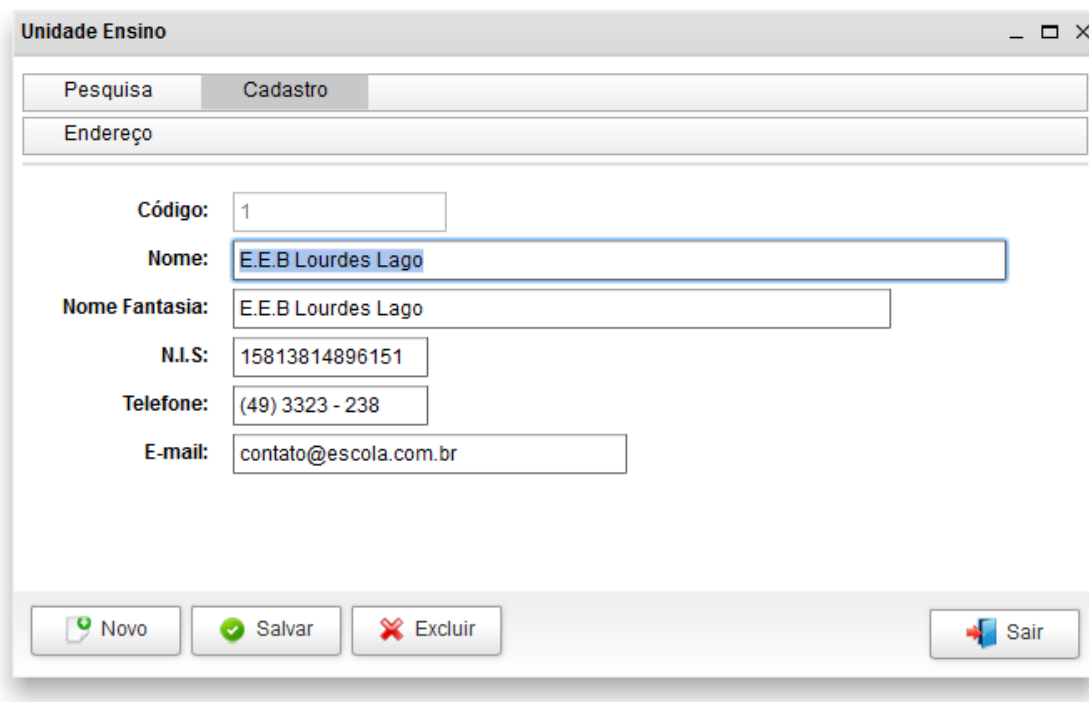
A Figura 7 apresenta o cadastro da Unidade de Ensino, esta tela possui uma tela de lista das unidades cadastradas.



Codigo	Nome
1	E.E.B Lourdes Lago

Figura 7 - Cadastro da Unidade de Ensino.

Com um duplo clique é possível gerenciar as informações do registro selecionado, conforme mostra a Figura 8.



Unidade Ensino

Pesquisa Cadastro

Endereço

Código: 1

Nome: E.E.B Lourdes Lago

Nome Fantasia: E.E.B Lourdes Lago

N.I.S: 15813814896151

Telefone: (49) 3323 - 238

E-mail: contato@escola.com.br

Novo Salvar Excluir Sair

Figura 8 - Tela para manutenção de Unidade de Ensino.

Os demais cadastros do sistema seguem o padrão apresentado nas Figuras 7 e 8.

Algumas telas possuem a opção de emitir relatório conforme mostram as Figuras 9 e 10.

A Figura 9 apresenta a tela de matrícula, onde há um botão pelo qual é possível emitir o boletim do aluno.

Matricula

Pesquisa Cadastro

Código: 6

Aluno: Robson Josué

Classe: 3 - Classe 6B

Situação: 2 - Aprovado

Novo Salvar Excluir Sair

Emitir Boletim

Figura 9 - Tela de Matrícula.

A Figura 10 apresenta a Tela de Classes e contém a mesma funcionalidade da Tela de Matrícula, mas por meio dela será emitido o boletim de todos os alunos da classe.

Classe

Pesquisa Cadastro

Código: 8

Nome: Classe 7B

Curso: 3 - Ensino Fundamental 6º/9º Ano

Serie: 6 - 6º Ano

Periodo Letivo: 2014

Unidade de Ensino: E.E.B Lourdes Lago

Turno: 2 - Matutino

Novo Salvar Excluir Sair

Emitir Boletim

Figura 10 - Tela de Classe.

A Figura 11 apresenta o boletim gerado a partir da tela de matricula no formato PDF (*Portable Document Format*).

E.E.B Lourdes Lago
contato@escola.com.br
(49) 3323 - 238
CHAPECO - SC

Boletim Escolar por Matricula

Aluno:	5 - Abdul Kalif				
Classe:	3 - Classe 6B				
Curso:	3 - Ensino Fundamental 6º/9º Ano				
Serie:	6 - 6º Ano	Turno:			2 - Matutino
Faltas:	29	Frequencia:			97,10%
Situação:	2 - Aprovado				
Disciplina	1º	2º	3º	4º	Média Final
Artes	10,00	8,50	9,80	5,00	8,33
Ciências	0	0	0	0	0
Educação Física	0	0	0	0	0
Geografia	0	0	0	0	0
História	0	0	0	0	0
Inglês	0	0	0	0	0
Matematica	0	0	0	0	0
Português	0	0	0	0	0

Figura 11 – Relatório de Boletim Escolar por Matrícula.

4.4 IMPLEMENTAÇÃO DO SISTEMA

Nesta seção são apresentados exemplos da codificação do sistema, com o objetivo de mostrar o uso das tecnologias adotadas para esta finalidade.

A Listagem 1 apresenta o arquivo responsável pela configuração do Hibernate. O arquivo persistence.xml tem por finalidade informar ao Hibernate em qual banco de dados ele deve conectar, seu usuário e senha para acesso, além de ler as anotações feitas com JPA para identificar as classes e suas respectivas tabelas para persistência na base de dados.

```

1 |<?xml version="1.0" encoding="UTF-8"?>
2 |<persistence version="1.0"
3 |   xmlns="http://java.sun.com/xml/ns/persistence" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4 |   xsi:schemaLocation="http://java.sun.com/xml/ns/persistence http://java.sun.com/xml/ns/persistence/persistence_1_0.xsd">
5 |
6 |   <persistence-unit name="provider" transaction-type="RESOURCE_LOCAL">
7 |     <provider>org.hibernate.ejb.HibernatePersistence</provider>
8 |     <properties>
9 |       <property name="hibernate.archive.autodetection" value="class,hbm" />
10 |      <property name="hibernate.connection.driver_class" value="org.postgresql.Driver" />
11 |      <property name="hibernate.connection.username" value="postgres" />
12 |      <property name="hibernate.connection.password" value="root" />
13 |      <property name="hibernate.connection.url" value="jdbc:postgresql://localhost/educacao" />
14 |      <property name="hibernate.cache.provider_class" value="org.hibernate.cache.NoCacheProvider" />
15 |      <property name="hibernate.cache.use_second_level_cache" value="false"/>
16 |      <property name="hibernate.cache.use_query_cache" value="false"/>
17 |      <property name="eclipselink.query-results-cache" value="false"/>
18 |
19 |      <property name="hibernate.show_sql" value="true" />
20 |      <property name="hibernate.format_sql" value="true" />
21 |      <property name="use_sql_comments" value="false" />
22 |      <property name="hibernate.hbm2ddl.auto" value="none" />
23 |      <property name="hibernate.dialect" value="org.hibernate.dialect.PostgreSQLDialect"/>
24 |    </properties>
25 |  </persistence-unit>
26 |
27 |</persistence>

```

Listagem 1 - Arquivo de configuração do Hibernate

A Listagem 2 apresenta a classe HDao, responsável por ler o arquivo de configuração persistence.xml, ler as anotações em JPA nas classes do sistema e abrir a conexão com a base de dados.

A classe HDao contém os métodos comuns para persistência já implementados, salvar, excluir, atualizar e listar, e pode ser utilizado por qualquer classe que contenha as anotações do JPA.

```

1 package br.com.swes.persistence.jpa;
2
3 import java.io.Serializable;
4
16 public class HDao<T> implements Dao<T> {
17
18     private static EntityManager entityManager;
19     protected final Class<T> oClass;
20
21     @SuppressWarnings("unchecked")
22     public HDao() {
23         super();
24         this.oClass = (Class<T>) ((ParameterizedType) getClass().getGenericSuperclass()).getActualTypeArguments()[0];
25     }
26
27     protected EntityManager getEntityManager() {
28         try{
29             if (entityManager == null) {
30                 EntityManagerFactory emf = Persistence.createEntityManagerFactory("provider");
31                 emf.getCache().evictAll();
32                 entityManager = emf.createEntityManager();
33             }
34         }
35         catch(Exception e){
36             e.printStackTrace();
37         }
38         return entityManager;
39     }
40
41     public boolean exclui(T obj) {
42         if (obj != null) {
43             getEntityManager().clear();
44             getEntityManager().getTransaction().begin();
45             obj = getEntityManager().merge(obj);
46             getEntityManager().remove(obj);
47             getEntityManager().getTransaction().commit();
48             return true;
49         }
50         return false;
51     }
52
53     public List<T> lista() {
54         return lista(null);
55     }
56
57     @SuppressWarnings("unchecked")
58     public List<T> lista(String ordem) {
59         String command;
60         getEntityManager().clear();
61         if (ordem != null && !ordem.isEmpty()) {
62             command = "SELECT obj FROM " + oClass.getSimpleName() + " obj order by obj." + ordem;
63         } else {
64             command = "SELECT obj FROM " + oClass.getSimpleName() + " obj";
65         }
66         return getEntityManager().createQuery(command).getResultList();
67     }
68
69     public T carregaPorId(Serializable id) {
70         return (T) getEntityManager().find(oClass, id);
71     }
72
73     public T salva(T obj) {
74         // getEntityManager().clear();
75         getEntityManager().getTransaction().begin();
76         getEntityManager().persist(obj);
77         getEntityManager().getTransaction().commit();
78         return obj;
79     }
80
81     public T atualiza(T obj) {
82         getEntityManager().clear();
83         getEntityManager().getTransaction().begin();
84         getEntityManager().merge(obj);
85         getEntityManager().getTransaction().commit();
86         return obj;
87     }
88
89     @SuppressWarnings("unchecked")
90     public T carrega(String query, Map<String, Object> parametros) {
91         getEntityManager().clear();
92         Query q = getEntityManager().createQuery(query);
93         for (String chave : parametros.keySet()) {
94             q.setParameter(chave, parametros.get(chave));
95         }
96         try {
97             return (T) q.getSingleResult();
98         } catch (NoResultException nre) {
99             return null;
100         }
101     }
102 }

```

Listagem 2 - Classe HDao.

A Listagem 3 apresenta o código responsável por fazer uma pesquisa em banco de dados por nome de uma pessoa e retornar os dados desta pessoa conforme o filtro informado.

```

1 package br.com.swes.persistence.jpa;
2
3 import java.util.HashMap;
4
5
6
7
8
9
10 public class HPessoaDao extends HDao<Pessoa> implements PessoaDao{
11
12     @Override
13     public List<Pessoa> pesquisaPorNome(String nome) {
14         String query = "SELECT obj FROM Pessoa obj WHERE upper(obj.nome) LIKE :filtro";
15         Map<String, Object> parametros = new HashMap<String, Object>();
16         parametros.put("filtro", "%" + nome.toUpperCase() + "%");
17         return pesquisa(query, parametros);
18     }
19
20 }
21

```

Listagem 3 - Classe HPessoaDao.

A Listagem 4 apresenta a Classe Pessoa com as anotações do JPA.

```

1 package br.com.swes.model;
2
3 import java.io.Serializable;
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18 @SuppressWarnings("serial")
19 @Entity
20 @Table(name = "gepessoa")
21 public class Pessoa implements Serializable{
22
23     @Id
24     @SequenceGenerator(name = "gepessoa_pescodigo_seq" , sequenceName = "gepessoa_pescodigo_seq" ,allocationSize = 1)
25     @GeneratedValue (strategy = GenerationType.SEQUENCE ,generator = "gepessoa_pescodigo_seq")
26     @Column(name="pescodigo", updatable=false)
27     private Integer id;
28     @Column(name="pesnome")
29     private String nome;
30     @Column(name="pescpf")
31     private String cpf;
32     @Column(name="pesrg")
33     private String rg;
34     @Column(name="pesdatnas")
35     private Date dataNascimento;
36
37     @OneToOne(fetch=FetchType.EAGER)
38     @JoinColumn(name="pespai")
39     private Pessoa pai;
40     @OneToOne(fetch=FetchType.EAGER)
41     @JoinColumn(name="pesmae")
42     private Pessoa mae;
43     @OneToOne(fetch=FetchType.EAGER , cascade = CascadeType.ALL)
44     @JoinColumn(name="endereco_id")
45     private Endereco endereco;
46
47     public Pessoa() {
48         super();
49         // TODO Auto-generated constructor stub
50     }
51
52 }
53

```

Listagem 4 - Classe Pessoa

A Listagem 5 mostra o código em MXML na aplicação em Flex responsável por mapear a classe que contém os métodos, esta classe de serviço está no Java.

```

245 <s:RemoteObject id="pessoaRO" destination="PessoaService" fault="{onFault(event)}">
246   <s:method name="lista" result="{this.onResultListaPessoa(event)}" fault="{onFault(event)}" />
247   <s:method name="salva" result="{this.onResultSalvaPessoa(event)}" fault="{onFault(event)}" />
248   <s:method name="exclui" result="{this.onResultExcluiPessoa(event)}" fault="{onFault(event)}" />
249 </s:RemoteObject>

```

Listagem 5 - Remote Object

Na linha 245 da Listagem 5 está declarado o Remote Object pessoaRO, logo abaixo na linha 246 está declarado o método “lista” que existe na classe. Para cada método informado existe uma função, à qual será atribuído o retorno do lado do Java, na linha 246 é possível ver a função onResultListaPessoa(event) que recebe o retorno do método lista.

A listagem 6 apresenta a classe PessoaService no java que é o destino do Remote Object pessoaRO.

```

1 package br.com.swes.service;
2
3 import java.util.ArrayList;
4
5 public class PessoaService {
6
7     public List<Pessoa> pesquisaPorNome( String filtro){
8         try{
9             List<Pessoa> retorno = DaoFactory.getInstance().getPessoaDao().pesquisaPorNome(filtro);
10            return retorno;
11        } catch (Exception e) {
12            System.out.println(e.getMessage());
13            e.printStackTrace();
14            return null;
15        }
16    }
17
18    public ArrayList<Pessoa> lista() {
19        try {
20            ArrayList<Pessoa> retorno = (ArrayList<Pessoa>) DaoFactory.getInstance().getPessoaDao().lista();
21            return retorno;
22        } catch (Exception e) {
23            System.out.println(e.getMessage());
24            e.printStackTrace();
25            return null;
26        }
27    }
28
29 }
30

```

Listagem 6 – Classe PessoaService

Na linha 9 da Listagem 6 é possível ver o nome da classe PessoaService que é o destino informado na aplicação Flex e logo abaixo na linha 21 está declarado o método “lista” que é o método informado na aplicação Flex.

Esta conexão entre a aplicação desenvolvida em Flex e código em Java acontece por meio do framework BlazeDS e é configurado no arquivo remoting-

config.xml conforme mostra a Listagem 7 abaixo. O BlazeDS lê este e outros arquivos para que a comunicação entre o Flex e o Java ocorra com eficiência e segurança.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <service id="remoting-service"
3     class="flex.messaging.services.RemotingService">
4
5     <adapters>
6         <adapter-definition id="java-object" class="flex.messaging.services.remoting.adapters.JavaAdapter" default="true"/>
7     </adapters>
8
9     <default-channels>
10        <channel ref="my-amf"/>
11    </default-channels>
12
13    <destination id="UsuarioService">
14        <properties>
15            <source>br.com.swes.service.UsuarioService</source>
16        </properties>
17    </destination>
18    <destination id="PessoaService">
19        <properties>
20            <source>br.com.swes.service.PessoaService</source>
21        </properties>
22    </destination>
```

Listagem 7 - Arquivo remoting-config.xml

5 CONCLUSÕES

O objetivo desse trabalho de conclusão de curso foi demonstrar que com o uso das tecnologias Java Flex, PostgreSQL, Adobe Flash Builder, BlazeDS e demais, apresentadas no Capítulo 3 foi possível implementar as funcionalidades pretendidas para o sistema Web para controle de matrículas e emissão de boletim escolar.

Os recursos que a linguagem Java oferece como orientação a objetos, herança e polimorfismo, trouxeram agilidade para a produção de código. O ambiente de desenvolvimento apresenta recursos que facilitam e agilizam o trabalho do desenvolvedor, por exemplo, a criação de métodos automáticos por meio de teclas de atalho do eclipse e no Adobe Flash Builder.

A maioria das regras de negócios em aplicações Java/Flex são implementadas em Java e procedimentos que requerem maiores processamentos são realizados no banco de dados através de funções e/ou procedures. Devido a linguagem Java ser amplamente difundida e a tecnologia Flex ser de fácil aprendizado e rica em componentes, as manutenções futuras no projeto serão fáceis de realizar. A funcionalidade essencial do sistema foi implementada por completo, atendendo, assim, aos objetivos propostos neste trabalho.

REFERÊNCIAS

ALLAIRE, Jeremy. **Macromedia Flash MX: next-generation rich client. 2002.**

Disponível em <http://www.adobe.com/devnet/flash/whitepapers/richclient.pdf>.

Acesso em: 05 nov. 2014.

ATLAS BRASIL – **Atlas do Desenvolvimento Humano.** Disponível em

<http://www.atlasbrasil.org.br/2013/pt/ranking>, acesso em 17 de nov. de 2014.

BIAZUS, Diego de Oliveira. **PostgreSQL Introdução e Historico.** Disponível

em https://wiki.postgresql.org/wiki/Introducao_e_historico, Acesso em 05 nov.

2014.

BRASIL. **Lei nº 9.394, de 20 de dezembro de 1996.** Estabelece as diretrizes e bases da educação nacional. Disponível em:

<http://www.planalto.gov.br/ccivil_03/Leis/L9394.htm>. Acesso em: 04 abr. 2014.

CNET. **IBM Makes \$40 Million Open-Source Offer.** Disponível em

http://news.cnet.com/IBM-makes-40-million-open-source-offer/2100-1001_3-275388.html. Acesso em 22 jul 2014.

DUHL, Joshua. **Rich internet applications.** White Paper. In: IDC Opinion. Disponível em:

<http://www.adobe.com/platform/whitepapers/idc_impact_of_rias.pdf>. Acesso em: 20 jun 2014.

FLEX. **About Apache Flex.** Disponível em <<http://flex.apache.org/about-what-is.html>>. Acesso em 21 jul 2014.

GUANAIS, Kaio Araújo. **Aplicações Ricas de Internet, 2010.** Disponível em: <<http://www3.iesam-pa.edu.br/ojs/index.php/sistemas/article/viewFile/544/412>>.

Acesso em 21 jun. 2014.

INEP. **Educa Censo.** Disponível em <http://portal.inep.gov.br/Web/educacenso/educacenso>>. Acesso em 03 jun 2014.

MEDEIROS, Igor. **Artigo de Introdução a JPA Java Persistence.** DevMedia. Disponível em <http://www.devmedia.com.br/introducao-a-jpa-java-persistence-api/28173>>. Acesso em 27 out 2014.

PALMEIRA, Thiago Vinícius Varallo. **Conhecendo o Eclipse uma apresentação detalhada da IDE.** Disponível em <http://www.devmedia.com.br/conhecendo-o-eclipse-uma-apresentacao-detalhada-da-ide/25589>. Acesso em 21 jul 2014.

PGADMIN. **Introduction.** Disponível em <http://www.pgadmin.org/>, Acesso em 05 nov. 2014.

POSTGRESQL. **About PostgreSQL.** Disponível em <http://www.postgresql.org/about/>. Acesso em 05 nov. 2014.

PRESSMAN, R. **Engenharia de Software.** MCGRAW HILL - ARTMED, 7. ed., 2011.

PROFUNCIONARIO. **Informática Aplicada a educação.** Disponível em http://portal.mec.gov.br/seb/arquivos/pdf/profunc/infor_aplic_educ.pdf. Acesso em 20 jun 2014.

SOFTWARE PUBLICO. **Portal do Software Público Brasileiro.** Disponível em <http://www.softwarepublico.gov.br/>. Acesso em 20 jun 2014.

SOMMERVILLE, Ian. **Engenharia de software.** São Paulo. 6. ed. Pearson Education Companion, 2003.

TAVARES, Neide Rodriguez Barea. **História da informática educacional no Brasil observada a partir de três projetos públicos.** Disponível em <http://www.lapeq.fe.usp.br/textos/tics/ticspdf/neide.pdf>. Acesso em 24 jul 2014.

VISUAL PARADIGM. **Visual Paradigm**. Disponível em <http://www.visual-paradigm.com/> . Acesso em 05 nov. 2014.