Projeto de Gestão Escolar

**Daniel Barboza, Felipe Augusto Hertzer, Jonas Mario Kunzler, Moisés Rodrigues Machado, Vinicius Madeira Radiske**

Disciplina de Projeto de Software

Universidade de Santa Cruz do Sul (UNISC) – Santa Cruz do Sul, RS – Brasil

{danibarbozad, felipeaugustohertzer, jonasmkunzler, moises.93, vradiske} @gmail.com

***Resumo.*** *Este documento autoriza formalmente o início do projeto e designa o gerente para o mesmo. O projeto trata-se de um Sistema de Gestão sobre Rotinas Escolares – SGRE.*

**1. TERMO DE ABERTURA**

**1.1 Equipe do Projeto**

Nome do aluno responsável: Jonas Mario Kunzler

Integrantes: Daniel Barboza, Felipe Hertzer, Moisés Machado, Vinicius Radiske

**1.2 Partes Interessadas (Stakeholders)**

|  |  |
| --- | --- |
| **Nome** | **Função** |
| Carla F. Andrade | Diretora |
| Maria D. Castanhagi | Secretária |
| Roberta H. S. Santos | Estagiaria |
| João S. Henn | Professor |
| Eliza Samúdio | Professor |

**1.3 Escopo do Projeto**

Tendo em mente as dificuldades e complexidades de processos envolvendo uma escola particular, será desenvolvida uma solução sistematizada dos processos mais persistentes encontrados. Este projeto visa controlar as notas dos alunos e seus processos quando iniciado o período de rematrículas. Bem como organizar e controlar o fluxo de obras do acervo bibliográfico da escola.

**Sobre Notas:**

Com a implantação do sistema será gerado um controle das notas individuais por alunos, divididas pelo período escolar, junto teremos o controle de presença que influenciará na aprovação ou não do aluno ao final do período letivo. Estas notas irão ser atribuídas para as diversas matérias que o aluno ira participar, ao final do período letivo estas notas serão exportadas para o boletim, mantendo um histórico do aluno ao longo de sua vida escolar dentro da escola.

Este modulo do sistema terá como responsável o professor que ministrará a disciplina na escola, o qual lançará as notas juntamente com as faltas. Este processo será feito através de login e senha exclusiva de cada professor no sistema.

**Sobre a Biblioteca:**

Nesta etapa teremos uma bibliotecária a qual controlará pelo sistema o empréstimo e a devolução de livros, bem como o zelo pelo acervo e a inclusão de novos exemplares.

Ao aluno que desejar pesquisar algum livro na biblioteca, deve se dirigir a bibliotecária que por sua vez terá um terminal com sistema e consultará se há o livro desejado, indicando ao aluno a localização e a disponibilidade do mesmo. O aluno também pode localizar livremente a obra entre as estantes e efetuar o empréstimo.

**Sobre a matrícula:**

Este modulo será para controlar e organizar o período de rematrícula. Todos alunos já matriculados terão um período para firmar a rematrícula, caso esta não se confirme abrirá uma vaga para ingresso de um novo aluno na escola.

Alunos serão cadastrados e ajustados pela secretaria da escola, com login e senha de privilégios exclusivos, todas negociações entre aluno e escola serão mediados por este departamento.

Cada operador será responsável pelo cadastramento, inclusão, alterações de registros sobre seu domínio. Os terminais estarão interligados entre si, e compartilharão os itens dos demais setores, porem a alteração ou exclusão dos itens será permitido apenas no setor.

O sistema será composto por no mínimo três terminais e um servidor que estarão interligados por uma intranet, serão permitidos novos computadores caso exista a necessidade. O servidor não estará disponível para uso, apenas será permitido acesso a ele via rede, e o mesmo estará equipado com um No-Break de autonomia de 20 minutos, para prevenção de futuras quedas de energia, bem como o desligamento correto do sistema.

Seu desenvolvimento será na plataforma PHP e usará como banco de dados o MySQL.

Cada usuário terá um campo para efetuar seu login e senha no sistema, para fins de controles futuros de lançamentos e alterações. Será implementado um controle sobre informações disponíveis para visualização e alteração, pois nem todos terão acesso a todas as funcionalidades do sistema.

Internamente no sistema teremos duas maneiras de realizar backup dos dados e registros, de maneira manual onde é usada uma mídia removível ou automaticamente com horário pré-definido o qual o próprio sistema salva em todos os terminais via rede uma mesma cópia do backup, mantendo um histórico de uma semana, posteriormente são sobrescritos os registros mais antigos.

**1.4 Exclusões Específicas do Escopo**

- Não será utilizado em aplicativos móveis ou servidores web.

- O sistema não será projetado para controlar compras ou manutenções que venham a ser feitas.

- O sistema não contemplará o controle sobre folha de pagamento de funcionários.

- A questão contábil da instituição não será abrangida pelo sistema.

**1.5 Objetivos do Projeto**

O sistema tem por objetivo tornar mais eficiente, ágeis e seguras as operações realizadas nos processos de uma escola particular, os quais em períodos de rematrícula veem seu fluxo de pessoas aumentarem no setor de secretaria. O sistema visa ordenar este fluxo e diminuir o tempo para este ajuste.

Ao longo do tempo o sistema terá um histórico completo do aluno com suas notas e faltas, facilitando na geração de seu boletim e cálculo de notas para futura geração de seu histórico escolar ao finalizar o ensino fundamental e médio.

**1.6 Justificativas do Projeto**

O projeto será desenvolvido para aperfeiçoar o processo de rematrícula e geração das notas de cada aluno. Controlará também o acervo da biblioteca que até então era feita em um livro com empréstimos e devoluções de obras.

Pois atualmente a escola organiza suas rotinas em planilhas eletrônicas de Excel que são impressas, preenchidas a mão e arquivadas em pastas.

**1.7 PREMISSAS E RESTRIÇÕES**

**1.7.1 Premissas**

Para a implantação do sistema equipamentos deverão ser adquiridos para podermos garantir o funcionamento correto de todos os setores e funções.

Os novos funcionários deverão receber treinamento individual para ambientação com o novo sistema. Reuniões mensais devem ser agendadas para elucidação de pontos definidos pela equipe de desenvolvimento, bem como uma reunião geral ao final do marco 5 para início da etapa de implantação, que se dará de modo gradual.

**1.7.2 Restrições**

**1.7.2.1 Segurança da Informação**

Toda e qualquer pessoa envolvida com o projeto, que tenha acesso a informações privilegiadas do sistema, deverão assinar o Termo de Compromisso no qual se comprometem ao zelo das informações do sistema e das reuniões.

**1.7.2.2 Escopo**

Qualquer alteração no projeto deverá ser apresentada em reuniões de ponto de controle para que sua viabilidade seja analisada e aprovada pelo Gerente do projeto e a Diretora da escola.

**1.7.2.3 Tecnologia**

No desenvolvimento da solução, devem ser utilizadas ferramentas Open Source.

**1.8 Fatores Críticos de Sucesso do Projeto**

Requisitos funcionais e não funcionais importantes para a avaliação final do projeto.

**1.9 Prazo do Projeto**

O prazo estimado para este projeto, para o atendimento do escopo descrito nesta declaração, é de 90 dias.

Data Prevista para Início: 01/03/2016

Data Prevista para Término: 01/06/2016

Plano de Entregas e Marcos do Projeto

Entrega/Marco Descrição Data

Marco 1 Abertura do Projeto 01/03/2016

Marco 2 Levantamento dos Requisitos 15/03/2016

Marco 3 Termino do marco 2 29/03/2016

Marco 4 Documentação 02/04/2016

Marco 4 Termino do marco 4 30/04/2016

Marco 5 Desenvolvimento 14/05/2016

Marco 6 Entrega parcial p1 01/06/2016

O detalhamento dos prazos para as fases, pacotes de trabalho, atividades e tarefas do projeto encontra-se no cronograma do projeto.

**1.10 Aprovação**

Declaramos ter lido este documento e estar de acordo com seus termos, condições e conteúdo, aprovando-o em sua totalidade mediante assinatura abaixo.

**2. LISTA DE REQUISITOS**

**2.1 Requisitos não funcionais**

1. As interfaces devem ser rápidas e de fácil utilização.

2. O sistema deve suportar sistemas operacionais de plataforma Windows.

3. A equipe de desenvolvimento do sistema deverá utilizar Netbeans 8.1.0.

4. O sistema será utilizado em computadores desktops ou notebooks.

5. O sistema deve suportar acesso a base de dados local via conexão de rede.

6. Todas as transações devem ser garantidas pela ACID.

7. O sistema deve ser manipulado apenas por pessoas capacitadas.

8. O sistema deve suportar usuários e níveis de permissões.

9. O banco de dados estará criptografado para que não haja alterações indevidas.

10. O sistema estará disponível no formato 24/7.

11. O sistema deverá ser desenvolvido em PHP.

12. A comunicação do sistema será com o MySQL

**2.2 Glossário**

Rematrícula - Aluno já matriculado, que realiza a renovação da matricula.

Matricula - Novo aluno que pretende uma vaga na escola.

Boletim - Documento com informações sobre o aluno e seu desempenho escolar.

Nota - Índice de desempenho do aluno na matéria.

Retirada - empréstimo de um livro a um aluno.

Renovação - solicitação de adiamento da data de devolução de um livro emprestado.

Reserva - solicitação de reserva de um livro para um determinado aluno poder retirá-lo.

Multa - valor que um aluno deve pagar para regularizar a sua situação na biblioteca.

**2.3 C01 - Cadastro de Matricula**

1. A secretaria deve receber do aluno os seus documentos e de seus responsáveis para cadastro do mesmo no sistema;

2. Será feita uma busca por vagas na turma que o aluno pretende ingressar;

3. Se houver vagas ela inicia o cadastro e verifica os responsáveis;

4. Caso não haja vagas o aluno fica em uma lista de espera;

5. Após cadastro efetuado com sucesso, será entregue um documento comprovando sua matricula;

**2.4 C02 - Cadastro de Rematrícula**

1. Não poderá ter percentual de faltas maior que 25%.
2. Não pode possuir mais de 5 advertências.
3. Para avançar de série deve atingir em todas as matérias o índice mínimo de 7.
4. É recriado um processo de matricula onde já se possui os dados do aluno e a vaga garantida.

**2.5 C03 - Cadastro de Notas e Faltas**

1. O professor deverá logar-se no sistema.

2. Escolherá a turma e matéria desejada.

3. Deve selecionar o aluno e então registrar a nota do mesmo.

4. Caso o aluno tenha falta, deve ser registrado.

**2.6 C04 - Cadastro de Turma**

1. A secretária deverá logar-se no sistema;

2. São inseridas as informações referentes à turma, como série e matérias;

3. É associado um professor para cada matéria selecionada na turma.

**2.6 C04 - Geração do Boletim**

1. Com login e senha, será possível gerar um arquivo PDF.

2. Terá no arquivo os dados do Aluno as matérias cursadas e suas respectivas notas.

3. Caso o aluno tenha faltas serão exibidas por matéria a quantidade e sua porcentagem.

4. Caso o aluno tenha notas inferiores ao índice 7, serão exibidas em vermelho.

5. Para a retirada do mesmo será necessário um responsável.

**2.7 C05 - Cadastro de Pessoas**

1. Cada pessoa será identificada por seu RG e uma senha, terá acesso ao sistema e realizar determinadas ações dependendo do tipo de pessoa definido.

2. Se o ele tive o tipo bibliotecário poderá receber o pagamento de multas, efetuar retirada de livros, reservas de livros, renovações e receber devoluções de empréstimos de livros, terá acesso a consulta de exemplares disponíveis na biblioteca.

3. Se a pessoa for do tipo secretária ela poderá fazer inclusões de matrícula e rematrícula de alunos.

4. Se o tipo for um professor terá disponível as opções de inclusão de nota dos alunos e também das faltas.

5. Para cada aluno será gerado um número de matrícula único e intransferível. Também será gerada uma senha automática que pode ser alterada. Essa senha vai ser utilizada para quando o aluno ou responsável quiser saber as notas e ao número de faltas.

**2.8 C06 - Cadastro de autores, livros e exemplares**

1. Deve ser possível cadastrar autores e livros que serão relacionados de modo que um autor pode ter vários livros e um livro pode ter vários autores.

2. Cada livro pode ter diversos exemplares.

3. O exemplar pode ser marcado como "consulta na sede" e nesse caso não poderá ser retirado.

4. O exemplar também obrigatoriamente deverá ter um número específico na estante cadastrado, desse modo o bibliotecário poderá consultar a sua localização e indicar ao aluno, a fim deste encontrar o livro mais facilmente na biblioteca.

**2.9 C07 - Cadastro de Feriados**

1. Para o cálculo de dias úteis para devolução dos livros, os fins de semana e feriados serão desconsiderados.

2. Para isso, deverá ser feito um cadastro de feriados, podendo ser classificados por cidade, estado ou nacional.

3. Os feriados devem ter um dia e mês obrigatoriamente. Opcionalmente eles podem ser cadastrados em um ano específico.

**2.10 C08 - Retirada e devolução**

1. Cada aluno poderá fazer 10 retiradas simultâneas, sendo este o limite.

2. Cada retirada vale por cinco dias úteis (um fim de semana ou feriado [C07] não é considerado dia útil), após isso o livro deverá ser devolvido se não for renovado.

3. A retirada só é efetuada se o aluno estiver em dia com o setor financeiro da escola.

4. O aluno poderá renovar o livro via telefone ou presencialmente na biblioteca ao ser atendido, desse modo a renovação será prolongada por mais cinco dias úteis.

5. Um exemplar poderá ser renovado no máximo 10 vezes, após isso a renovação será bloqueada pelo sistema e o livro deve ser devolvido.

**2.11 C09 - Multas**

1. Caso o aluno atrase na devolução de um livro (por no mínimo um dia útil) ele estará automaticamente considerado irregular, e não poderá realizar mais nenhuma renovação, reserva ou retirada enquanto não regularizar através de pagamento.

2. O valor da multa é calculado por dia útil, sendo que a biblioteca poderá ter o seu valor de multa diário cadastrado.

3. A multa só é contada enquanto o livro não tiver sido devolvido, sendo assim, se um usuário estiver com três dias de atraso e devolver o livro, a multa não continuará contando dias, mas o usuário continuará irregular enquanto não realizar o pagamento.

4. Se o usuário perder ou rasurar um livro, deverá comunicar presencialmente o bibliotecário para combinar um pagamento para recolocação do livro. O bibliotecário deverá marcar o exemplar como extraviado.

**2.12 C10 - Reservas**

1. É possível que um aluno reserve livros na biblioteca presencialmente ou via telefone (falando com um bibliotecário).

2. O aluno só poderá reservar um livro caso esteja em situação regular (sem multas).

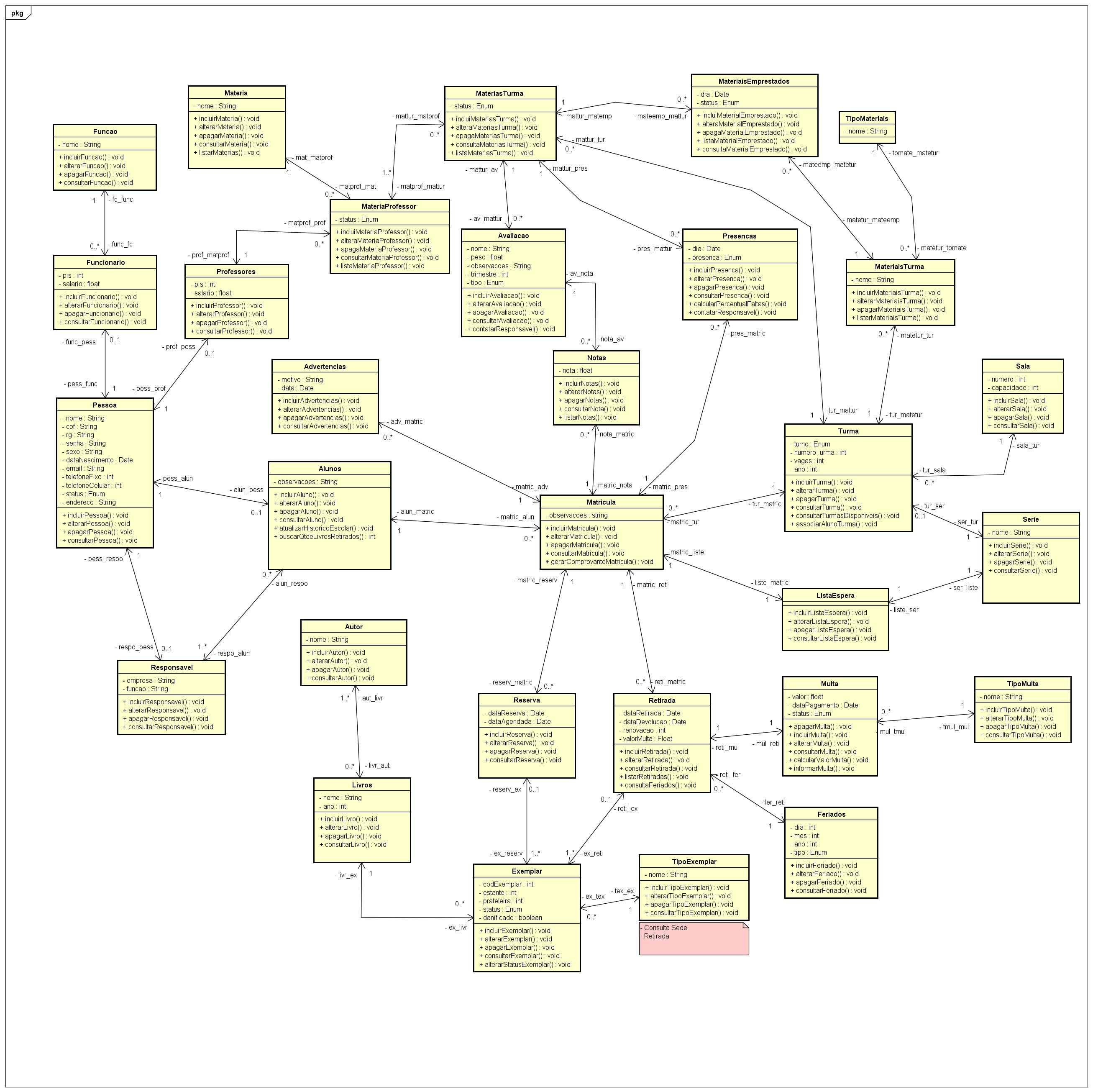
3. Se todos os exemplares de um livro estiverem retirados e um aluno fizer uma reserva, o aluno que tiver o livro retirado com a data de entrega agendada para a mais próxima da data atual não poderá renovar, a menos que um dos exemplares seja devolvido antes da data de entrega nesse meio tempo ou que o usuário cancele a reserva.

4. Caso mais de um aluno reserve um livro e todos os exemplares estejam retirados na biblioteca, eles serão colocados numa fila de reservas, sendo assim o primeiro a reservar terá acesso ao livro antes.

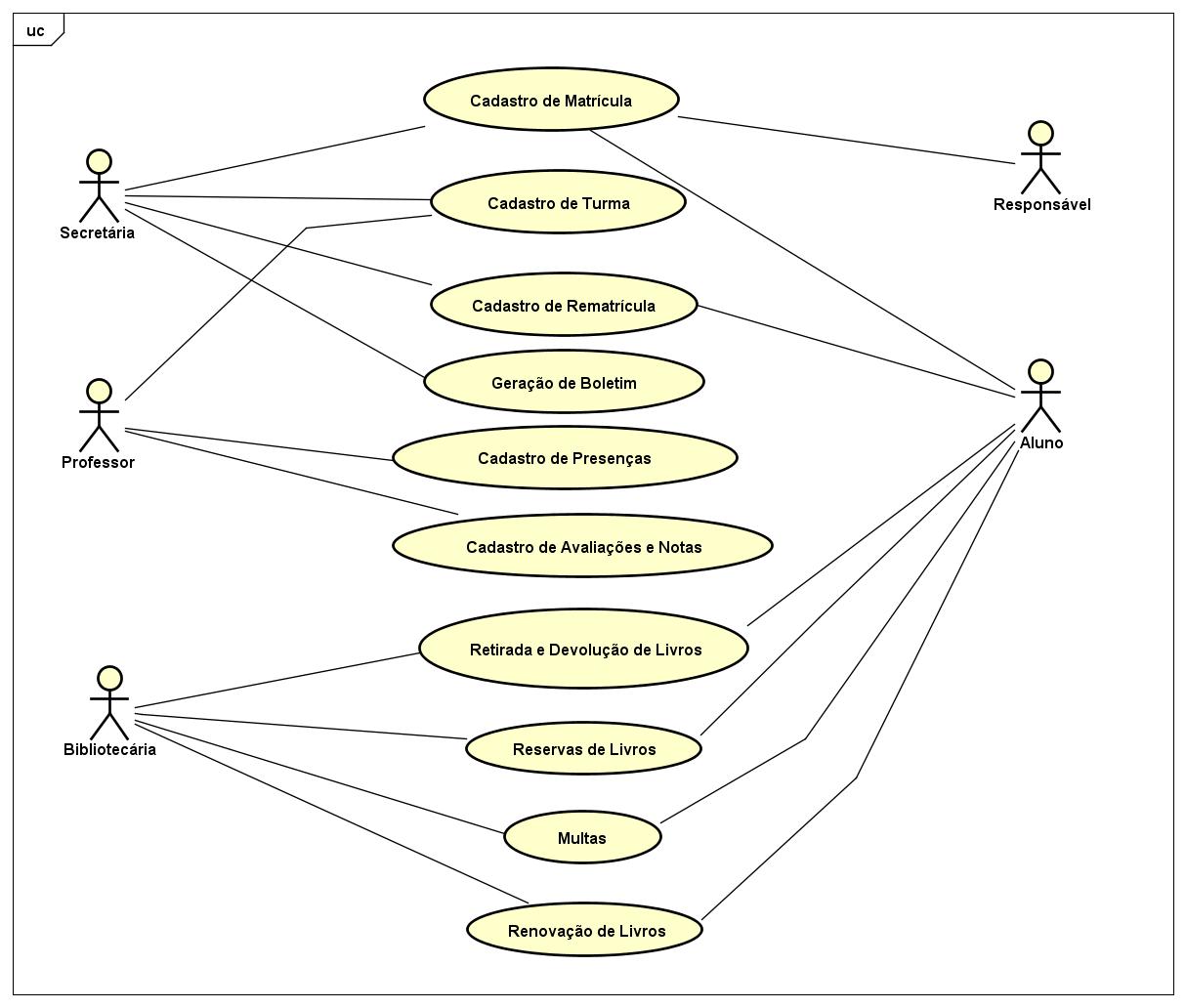
5. A reserva vale apenas por um dia útil, deste modo se passar o dia e a pessoa que reservou não retirar o livro, ele ficará livre ou passará para a próxima reserva da fila. Sendo assim, se o usuário fizer uma reserva sexta-feira e não retirar o livro até o final de expediente na segunda, sua reserva não valerá mais.

6. Cada aluno só poderá realizar 5 reservas simultâneas.

**3. DIAGRAMA DE CLASSES**



**4. DIAGRAMA DE CASOS DE USO**



**5. CASOS DE USO**

**5.1 Cadastro de Matrícula**

**5.1.1 Descrição**

Fluxo para o cadastramento de um novo aluno na escola

**5.1.2 Pré-condição**

Possuir documentos do aluno, e do responsável.

Declaração de Transferência (original)

Caso necessário histórico de Transferência (original, prazo de entrega 30 dias após a efetivação da matrícula)

Comprovante de Residência do responsável (1 xerox)

**5.1.3 Pós-condição**

Aluno apto a participar das aulas.

Apto a efetuar retiradas de livros.

**5.1.4 Requisitos relacionados**

Relacionado com C01, C02 e C05

**5.1.5 Lista de consulta**

Relação de atributos usados para exibição da lista de registros

**5.1.6 Fluxo Principal**

Cadastro Matricula

Inicia cadastro de matricula

Busca vagas turma do aluno

Se nao encontrar vaga entao

Insere aluno em lista de espera

Senao

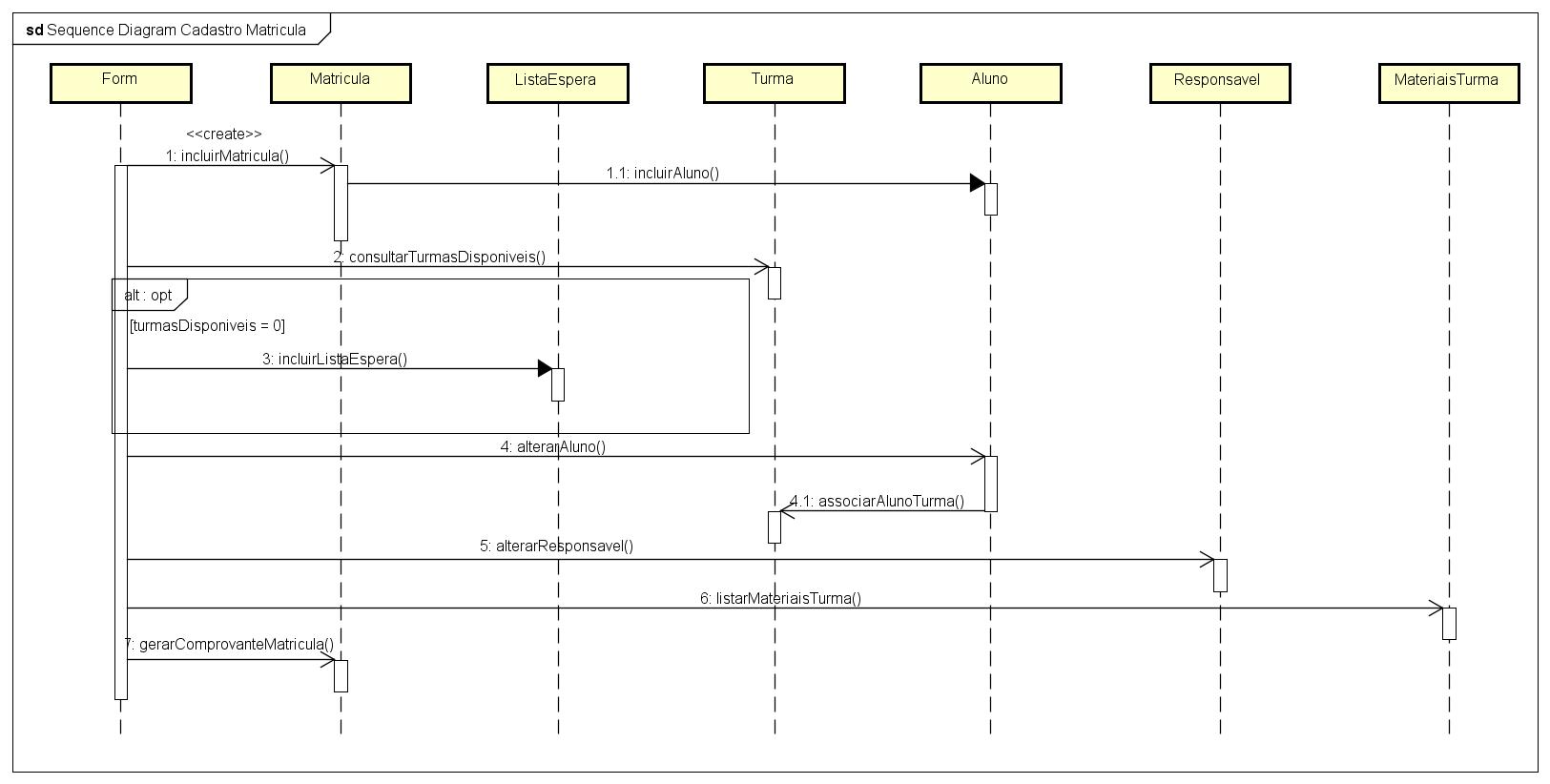
Cadastra dados do aluno

Cadastra dados do responsavel

Inclui aluno na turma

Imprime lista de materiais

Imprime comprovante matricula



**5.1.7 Especificação de métodos**

|  |  |
| --- | --- |
| Método | Descrição |
| Form#incluirMatricula() | Instancia uma nova matrícula com base nos dados preenchidos no formulário. |
| Form#consultarTurmasDisponiveis() | Busca todas turmas com vagas disponíveis. |
| Form#alterarAluno() | Atualiza valores de aluno cadastrado. |
| Turma#associarAlunoTurma() | Associa determinado aluno à turma existente. |
| Form#alterarResponsavel() | Atualiza valores de responsável cadastrado. |
| Form#listarMateriaisTurma() | Lista os materiais necessários para a turma selecionada. |
| Form#gerarComprovanteMatricula() | Gera comprovante de matrícula. |

**5.2 Cadastro de Rematrícula**

**5.2.1 Descrição**

Aluno matriculado faz sua matricula novamente para o próximo ano.

**5.2.2 Pré-condição**

Já estar matriculado.

Ter sido aprovado na série do ano passado.

Ter um bom comportamento dentro da instituição.

**5.2.3 Pós-condição**

Apto a assistir as aulas.

Apto a retirar livros na biblioteca.

**5.2.4 Requisitos relacionados**

Relacionado com C01, C02 e C05

**5.2.5 Lista de consulta**

Relação de atributos usados para exibição da lista de registros

**5.2.6 Fluxo Principal**

Cadastro de Rematricula

Busca matricula

Se nota acima de 7 E falta menor que 25%

Busca vaga proxima turma

Inclui aluno em turma

Imprime lista de materiais

Senao

Busca vaga mesma turma

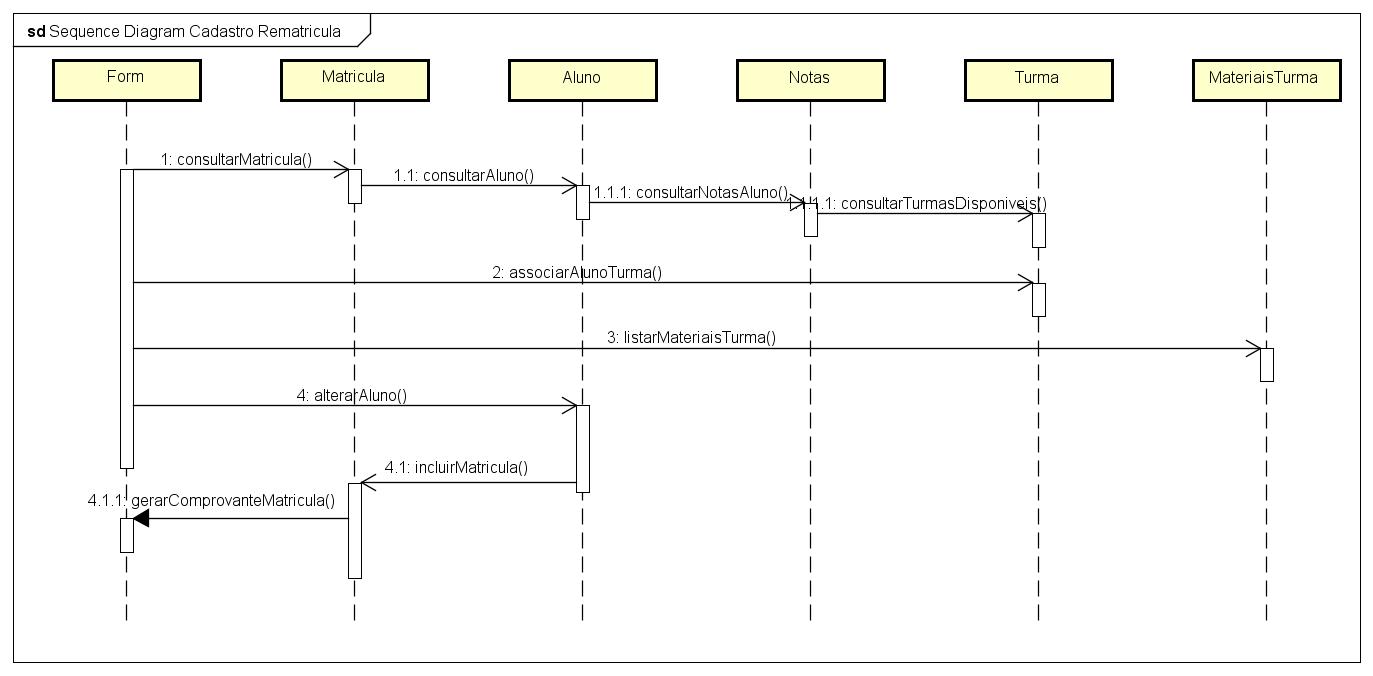
Se nao encontrar vaga entao

Insere aluno em lista de espera

Imprime comprovante matricula

Senao

Recusa rematricula



**5.2.7 Especificação de Métodos**

|  |  |
| --- | --- |
| Método | Descrição |
| Form#consultarMatricula() | Consulta dados de matrícula informada pelo usuário. |
| Aluno#consultarAluno() | Consulta dados de aluno por matrícula informada pelo usuário. |
| Notas#consultarNotasAluno() | Consulta notas de aluno por matrícula informada pelo usuário. |
| Turma#buscarVagasSerie() | Retorna todas turmas com vagas disponíveis por série. |
| Form#associarAlunoTurma() | Associa aluno à turma selecionada. |
| Form#listarMateriaisTurma() | Lista os materiais necessários para a turma selecionada. |
| Form#consultarAluno() | Consulta dados de aluno por matrícula informada pelo usuário. |
| Matricula#gerarComprovanteMatricula() | Gera comprovante de matrícula para alunos. |

**5.3 Geração de Boletim**

**5.3.1 Descrição**

Será gerado documento do trimestre, tendo notas e faltas do aluno.

**5.3.2 Pré-condição**

Ter matricula para geração.

Ter no sistema notas e as faltas lançadas pelos docentes.

**5.3.3 Pós-condição**

Entrega do mesmo aos responsáveis

Guardar histórico por alunos no banco de dados.

**5.3.4 Requisitos relacionados**

Relacionado com C03, C04 e C05

**5.3.5 Lista de consulta**

Relação de atributos usados para exibição da lista de registros

**5.3.6 Fluxo Principal**

Geração de Boletim

Busca matricula

Verifica disciplinas

Verifica notas e faltas

Se nota acima de 7 E falta menor que 25%

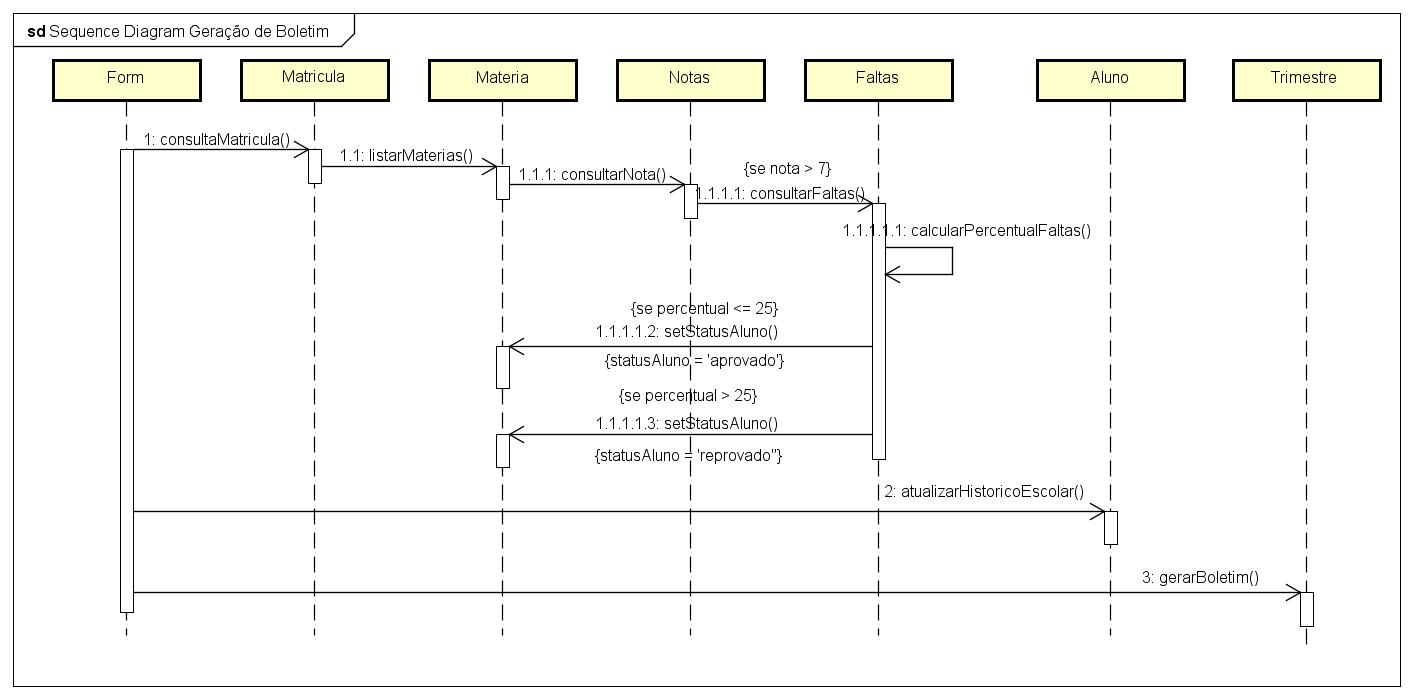
disciplina recebe aprovado

Senao

disciplina recebe reprovado

Atualiza historico escolar aluno

Imprime boletim



**5.3.7 Especificação de Métodos**

|  |  |
| --- | --- |
| Método | Descrição |
| Form#consultarMatricula() | Consulta dados de matrícula informada pelo usuário. |
| Materia#listarMaterias() | Lista todas matérias relacionadas à matrícula informada pelo usuário. |
| Notas#consultarNota() | Consulta notas de determinada matéria para matrícula informada pelo usuário. |
| Faltas#consultarFaltas() | Consultar faltas de matrícula informada pelo usuário para determinada matéria. |
| Faltas#calcularPercentualFaltas() | Calcula percentual de faltas de aluno em relação às aulas já ministradas. |
| Materia#setStatusAluno | Define o valor do campo Status de aluno. Valores possíveis são: “Aprovado” ou “Reprovado”. |
| Form#atualizarHistoricoEscolar | Atualiza histórico escolar com novos dados de notas e faltas de matérias cursadas pelo aluno. |
| Form#gerarBoletim | Gera boletim de aluno selecionado por usuário. |

**5.4 Retirada e Devolução**

**5.4.1 Descrição**

Fluxo para retirado e devolução de livros na biblioteca.

**5.4.2 Pré-condição**

Retirada

- Estar matriculado.

- Livro deve estar na biblioteca e ter exemplar disponível.

- Não ultrapassar o limite de retiradas.

Devolução

- Estar matriculado.

- Ter retirado um livro.

**5.4.3 Pós-condição**

Retirada

- Exemplar será atualizado para indisponível no sistema.

- Lança registro na matricula do aluno com numero do exemplar.

Devolução

- Exemplar será atualizado para disponível no sistema.

- Eventual extravio ou atraso na entrega, multa deverá ser cobrada.

**5.4.4 Requisitos relacionados**

Relacionado com C08, C09 e C10

**5.4.5 Lista de consulta**

Relação de atributos usados para exibição da lista de registros

**5.4.6 Fluxo Principal**

Retirada e Devolução

Busca matricula

Se retirada livro

localiza livro

verifica disponibilidade livro

verifica se exemplar disponivel

Verifica qtd livros retirados

Se qtd livros retirados alunosuperior a 7

recusa retirada

Senao

autoriza retirada

registra exemplar e aluno em tabela retirada

atualiza status do exemplar

Senao

verifica retiradas do aluno

verifica data de retirada e devolucao

verifica danos no livro

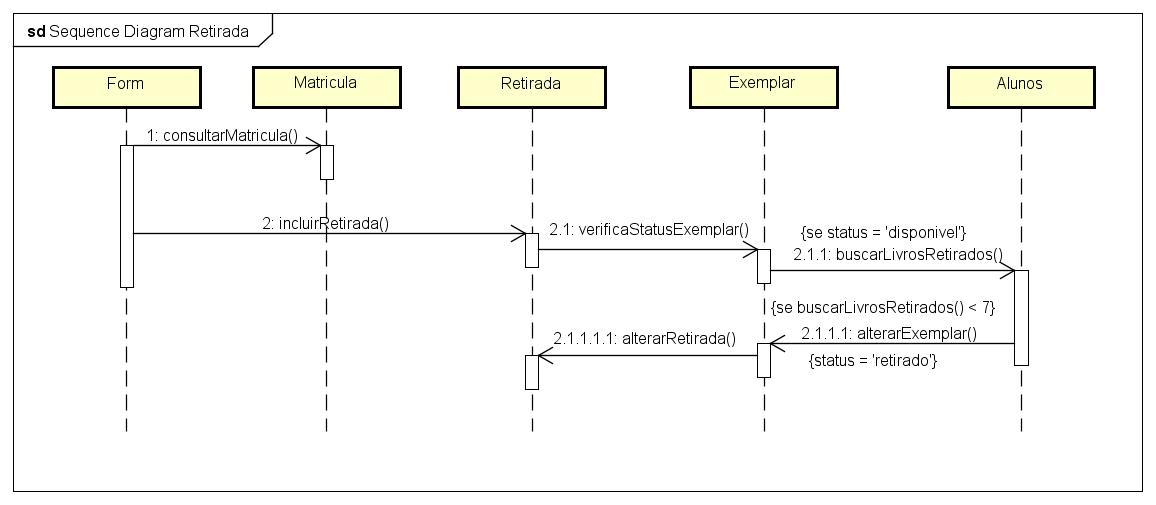
Se ha danos ou data maior que devolucao

lança multa financeiro

informa aluno sobre multa

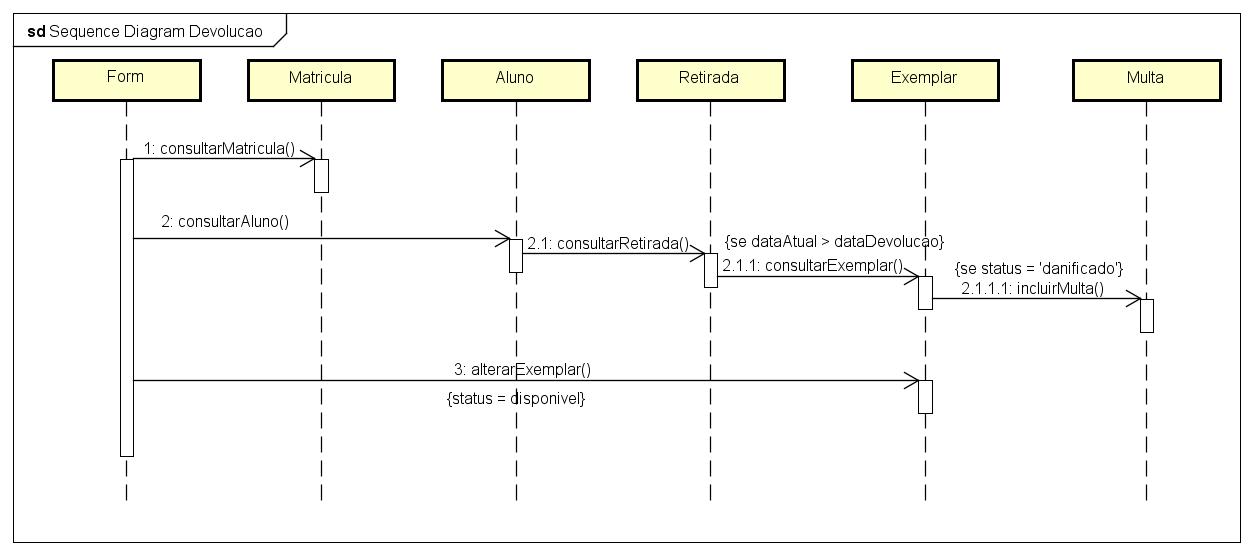
verifica exemplar a ser devolvido

atualiza status do exemplar



5.4.7 Especificação de métodos

|  |  |
| --- | --- |
| Método | Descrição |
| Form#consultarMatricula() | Consulta dados de matrícula informada pelo usuário. |
| Form#incluirRetirada() | Instancia nova retirada com base nos dados informados pelo usuário no formulário. |
| Exemplar#verificaStatusExemplar() | Verifica status do exemplar selecionado. Valores possíveis: “disponível”, “reservado”, “retirado”, “extraviado”, “atrasado” ou “danificado”. |
| Alunos#buscarLivrosRetirados() | Retorna quantidade de livros retirados por determinado aluno. |
| Exemplar#alterarStatus() | Altera status de exemplar. |
| Retirada#alterarRetirada() | Atualiza dados de retirada. |



5.4.8 Especificação de métodos

|  |  |
| --- | --- |
| Método | Descrição |
| Form#consultarMatricula() | Consulta dados de matrícula informada pelo usuário. |
| Form#consultarAluno() | Consulta dados de aluno selecionado pelo usuário. |
| Aluno#consultarRetirada() | Consulta dados de determinada retirada de aluno selecionado pelo usuário. |
| Exemplar#consultarExemplar() | Consulta dados de exemplar. |
| Multa#incluirMulta() | Inclui nova multa à determinado exemplar caso seu status seja “danificado”, “extraviado” ou “atrasado”. |
| Form#alterarExemplar() | Altera dados de determinado exemplar. |

**5.5 Reservas**

**5.5.1 Descrição**

Fluxo para reserva de uma obra pelo aluno na biblioteca.

**5.5.2 Pré-condição**

Estar matriculado.

Livro deve estar na biblioteca e ter exemplar disponível.

Não ultrapassar limite de retiradas.

Estar em dia com as mensalidades.

**5.5.3 Pós-condição**

Exemplar é marcado como reservado.

Lança reserva na matricula do aluno.

**5.5.4 Requisitos relacionados**

Relacionado com C08, C09 e C10

**5.5.5 Lista de consulta**

Relação de atributos usados para exibição da lista de registros

**5.5.6 Fluxo Principal**

Reservas

Busca matricula

localiza livro

verifica disponibilidade livro

verifica se exemplar disponivel

Verifica qtd livros retirados

Se qtd livros retirados aluno superior a 7

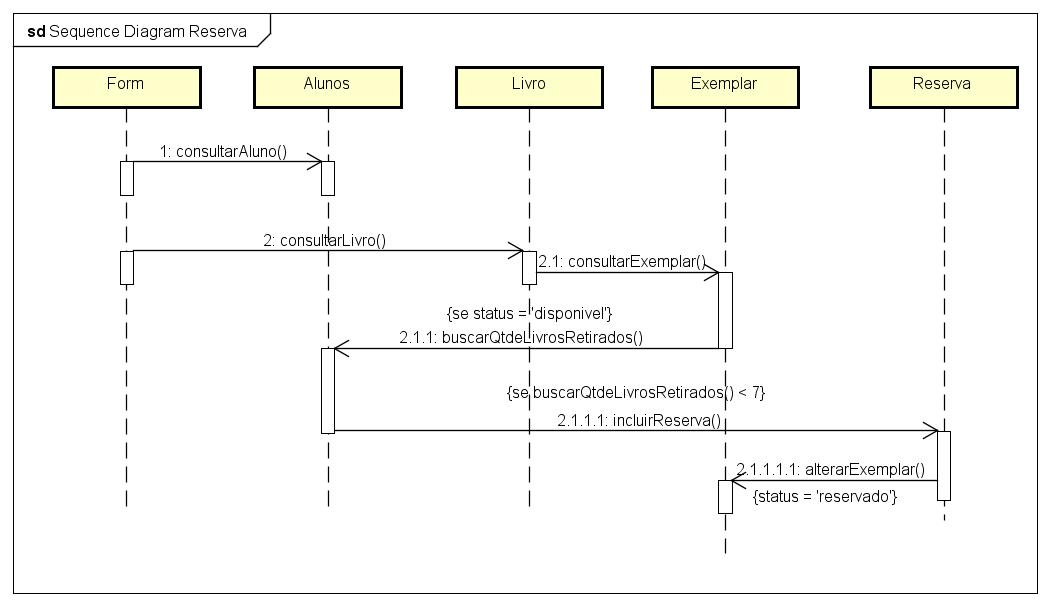
recusa reserva

Senao

autoriza reserva

registra reserva

atualiza status do exemplar



**5.5.7 Especificação de métodos**

|  |  |
| --- | --- |
| Método | Descrição |
| Form#consultarAluno() | Consulta dados de aluno selecionado pelo usuário. |
| Exemplar#consultarExemplar() | Consulta dados de exemplar selecionado por usuário. |
| Alunos#buscarQtdeLivrosRetirados() | Retorna quantidade de livros retirados pelo aluno. |
| Reserva#incluirReserva() | Instancia nova reserva com base nos dados preenchidos no formulário. |
| Exemplar#alterarExemplar() | Atualiza status do exemplar para “reservado”. |

**5.6 Multas**

**5.6.1 Descrição**

Fluxo para geração de multa sobre exemplares extraviados ou entregues fora da data.

**5.6.2 Pré-condição**

Estar matriculado.

Deve ter entregado um exemplar fora do prazo, ou extraviado o mesmo.

**5.6.3 Pós-condição**

Lançado multa no cadastro financeiro da matricula respectiva.

Se for atraso atualiza status do exemplar para disponível.

Se for extravio, retira exemplar do sistema.

**5.6.4 Requisitos relacionados**

Relacionado com C08, C09 e C10.

**5.6.5 Lista de consulta**

Relação de atributos usados para exibição da lista de registros

**5.6.6 Fluxo Principal**

Multas

Busca matricula

verifica tipo de multa

Se tipo atraso

verifica dias e multiplica por valor atraso

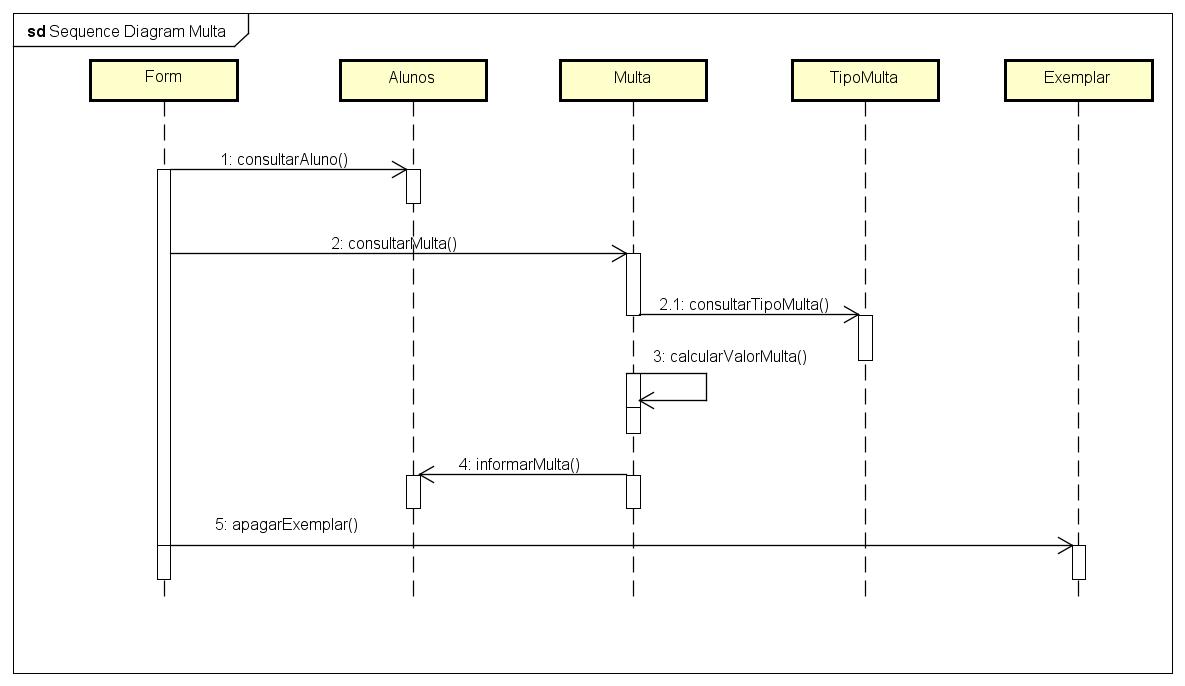
Se tipo dano ou perda

verifica preço de compra do livro

retira exemplar da biblioteca

informa ao aluno multa

lança multa financeiro



**5.6.7 Especificação de métodos**

|  |  |
| --- | --- |
| Método | Descrição |
| Form#consultarAluno() | Consulta dados de aluno selecionado pelo usuário. |
| Form#consultarMulta() | Consultar dados de multa informada pelo usuário. |
| TipoMulta#consultarTipoMulta() | Consulta tipo de determinada multa. |
| Multa#calcularValorMulta() | Verifica se o livro está atrasado, danificado ou extraviado, e calcula o valor da multa de acordo com o status do livro. |
| Multa#informarMulta() | Envia e-mail com valor de multa para aluno. |
| Form#apagarExemplar() | Caso o livro tenha sido extraviado ou danificado, apaga exemplar da base de dados. |

**5.7 Cadastro Notas e Faltas**

**5.7.1 Descrição**

Dados relacionados com ano, turma e disciplina do aluno.

**5.7.2 Pré-condição**

Possuir login e senha.

Estar lecionando em uma turma com alunos matriculados.

Possuir matricula onde serão guardados os registros.

**5.7.3 Pós-condição**

Sistema estará apto a gerar o boletim do trimestre.

**5.7.4 Requisitos relacionados**

Relacionado com C03, C04 e C05.

**5.7.5 Lista de consulta**

Relação de atributos usados para exibição da lista de registros

**5.7.6 Fluxo Principal**

Cadastro de Notas e Faltas

Inicia login no sistema

Se Login for professor

Seleciona materia

Seleciona turma

Seleciona aluno

Insere qnt trabalhos

Insere qnt provas

Verifica qtd de faltas

Se maior que 20%

avisa responsavel qtd falta

Insere faltas

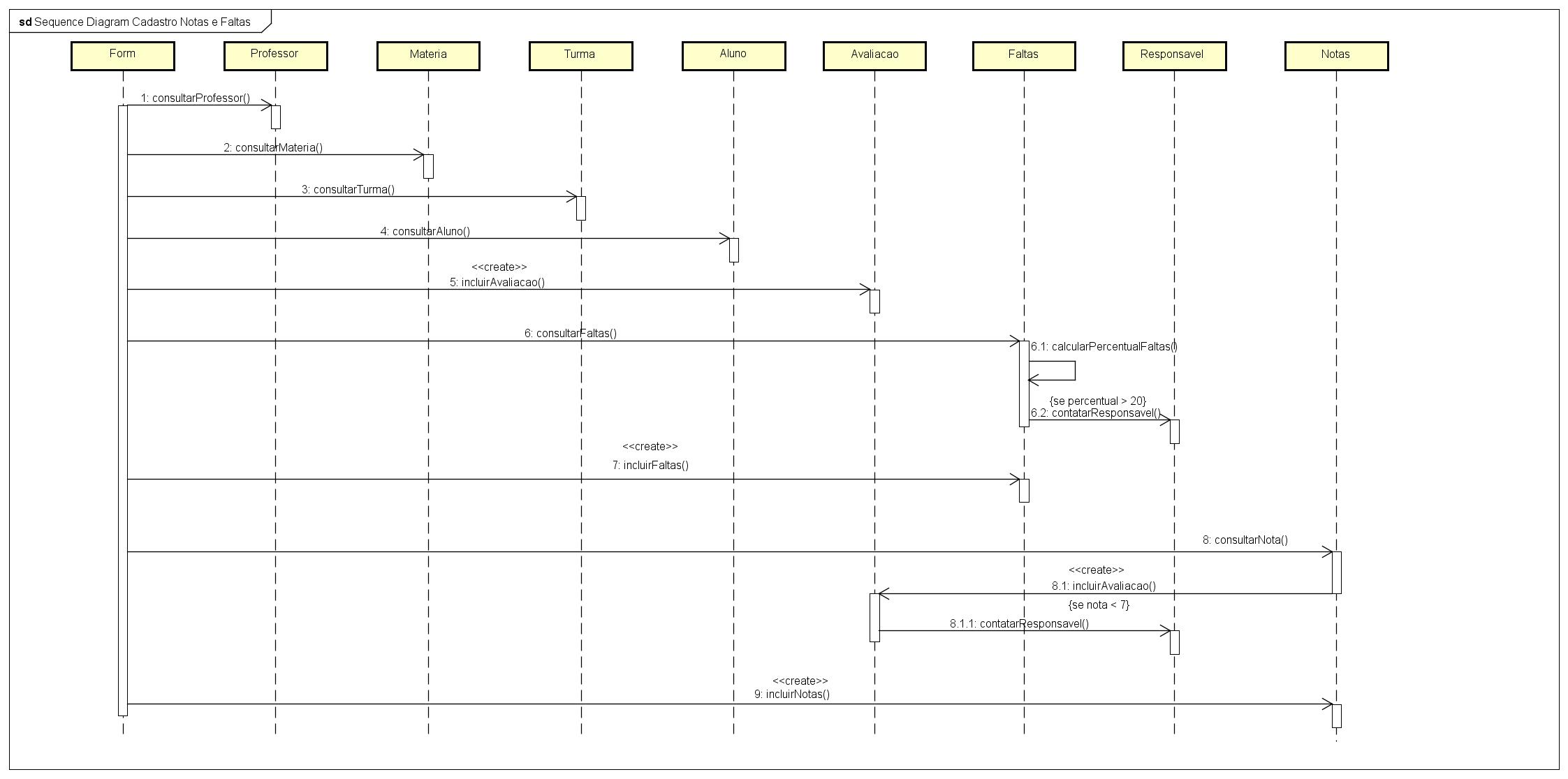
Verifica nota

Se notas menor que 7

aluno recebe prova extra de exame

avisa responsavel

Insere notas



**5.7.7 Especificação de Métodos**

|  |  |
| --- | --- |
| Método | Descrição |
| Form#consultarProfessor() | Consulta dados de professor informado pelo usuário. |
| Form#consultarMateria() | Consulta dados de matéria informada pelo usuário. |
| Form#consultarTurma() | Consulta dados de turma selecionada pelo usuário. |
| Form#consultarAluno() | Consulta dados de aluno selecionado pelo usuário. |
| Form#incluirAvaliacao() | Instancia uma nova avaliação com base nos dados preenchidos no formulário. |
| Form#consultarFaltas() | Consulta faltas de determinado aluno selecionado pelo usuário. |
| Faltas#calcularPercentualFaltas() | Calcula percentual de faltas de aluno em relação às aulas já ministradas. |
| Responsável#contatarResponsável() | Envia e-mail para responsável de determinado aluno. |
| Form#incluirFaltas() | Instancia nova quantidade de faltas com base nos dados preenchidos no formulário. |
| Form#consultarNota() | Consulta nota de aluno selecionado pelo usuário. |
| Avaliacao#incluirAvaliacao() | Instancia uma nova avaliação caso o aluno não tenha atingido a média necessária da matéria ou avaliação. |
| Responsavel#contatarResponsavel() | Envia e-mail para responsável de determinado aluno. |
| Form#incluirNotas | Instancia uma nova nota para determinada avaliação selecionada pelo usuário. |

**5.8 Cadastro de Turma**

**5.8.1 Descrição**

Fluxo para cadastro de nova turma e associação com série.

**5.8.2 Pré-condição**

Somente secretárias podem ter permissão para realizar o cadastro;

Professor deve estar disponível para ser associado à determinada matéria.

**5.8.3 Pós-condição**

Turma estará apta a receber alunos de acordo com quantidade de vagas.

**5.8.4 Requisitos relacionados**

Relacionado com C01 e C02.

**5.8.5 Lista de consulta**

Relação de atributos usados para exibição da lista de registros.

**5.8.6 Fluxo Principal**

Cadastro de Turma

Inicia login no sistema

Se Login for secretária

Seleciona turno

Informa numéro de turma

Informa quantidade de vagas

Informa ano

Insere qnt provas

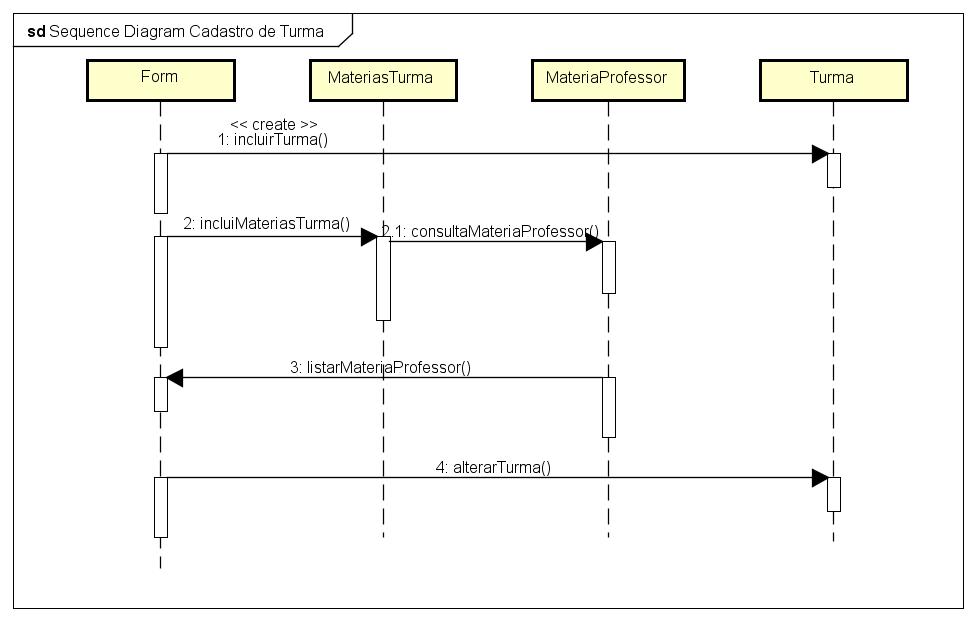
Insere matérias

Verifica status professor para turno informado

Se status ‘disponível’

associa professor com matéria

Insere nova turma no sistema



**5.8.7 Especificação de Métodos**

|  |  |
| --- | --- |
| Método | Descrição |
| Form#incluirTurma() | Cria nova turma. |
| Form#incluirMateriasTurma() | Cria nova matéria e associa à turma. |
| MateriaProfessor#consultaMateriaProfessor() | Consulta associação de matérias com professores disponíveis. |
| MateriaProfessor#listarMateriaProfessor() | Lista todas associações de matérias com professores disponíveis. |
| Form#alterarTurma() | Atualiza turma com novos dados de matérias e professores. |

**6. DESIGN PATTENS**

Design Patterns é uma solução geral reutilizável para um problema que ocorre com frequência em um contexto. Solução que já fora testada e implementada comprovando sua eficiência. Um padrão de projeto não é um código fonte propriamente, mas sim um template de como identificar a informação necessária para entender o problema, e a partir deste gerar a solução em termos de relacionais entre as classes e os objetos necessários.

O MVC ou Model-view-controller é uma arquitetura ou padrão que lhe permite dividir as funcionalidades de seu sistema/site em camadas, separa a representação da informação da interação do usuário. Essa métrica tem como foco dividir um grande problema em vários problemas menores e de menor complexidade. Dessa forma, qualquer tipo de alterações em uma das camadas não interfere nas demais, facilitando a atualização de layouts, alteração nas regras de negócio e adição de novos recursos. Em caso de grandes projetos, o MVC facilita muito a divisão de tarefas entre a equipe.

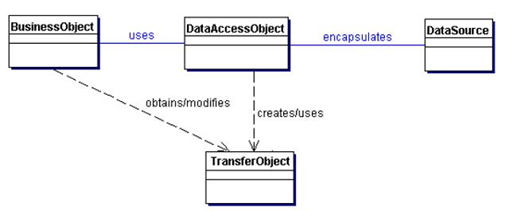
A camada do modelo (model) consiste nos dados da aplicação, regras de negócios, lógica e funções.

Na camada de uma visão (view) pode ser qualquer saída de representação dos dados, como uma tabela ou um diagrama. É possível ter várias visões do mesmo dado, como um gráfico de barras para gerenciamento e uma visão tabular para contadores.

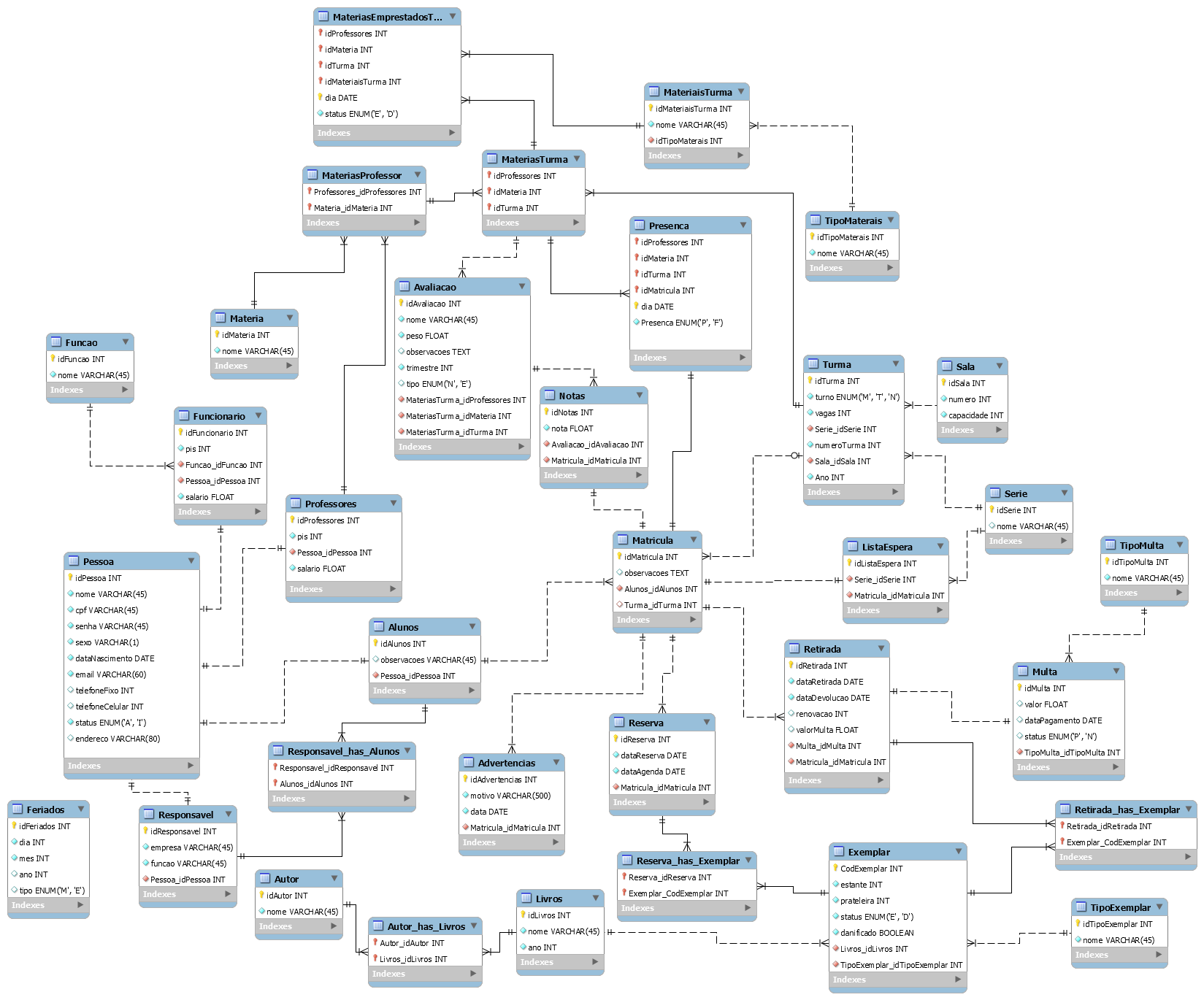
O controlador (controller) faz a mediação da entrada, convertendo-a em comandos para o modelo ou visão. As ideias centrais por trás do MVC são a reusabilidade de código e separação de conceitos.

**6.1 DAO**

Objeto de acesso a dados (ou simplesmente DAO, acrônimo de Data Access Object), é um padrão para persistência de dados que permite separar regras de negócio das regras de acesso a banco de dados. Numa aplicação que utilize a arquitetura MVC, todas as funcionalidades de bancos de dados, tais como obter as conexões, mapear objetos para tipos de dados SQL ou executar comandos SQL, devem ser feitas por classes DAO.



**7. MODELO BANCO DE DADOS**



**8. ESPECIFICAÇÕES DE REGRAS DE NEGÓCIO**

**8.1 Cadastrar Pessoa**

|  |  |
| --- | --- |
| Nome | Cadastrar Pessoa |
| Parâmetros | responsavel, funcionario, professor, aluno |
| Fluxo Principal | Se pessoa.rg não é vazio  E pessoa.cpf não é vazio  E pessoa.nome não é vazio  E pessoa.dataNascimento não é vazio  E pessoa.unidade não é vazio  E pessoa.email não é vazio  E pessoa.cpf não esta sendo usado Então  incluirPessoa()  Seleciona TipoPessoa  Se TipoPessoa = “funcionario” entao  Se funcionario.pis nao vazio  E funcionario.salario nao vazio entao  incluirFuncionario()  Se TipoPessoa = “professor” entao  Se professor.pis nao vazio  E professor.salario nao vazio entao  incluirProfessor()  Se TipoPessoa = “responsavel” entao  Se responsavel.funcao nao vazio  E responsavel.empresa nao vazio entao  incluirResponsavel()  Se TipoPessoa = “aluno” entao  Se aluno.observacao nao vazio entao  incluirAluno()  Senão  Retorna Falso |

**8.2 Cadastrar Responsável**

|  |  |
| --- | --- |
| Nome | Cadastrar Responsável |
| Parâmetros | resposavel, pessoa |
| Fluxo Principal | Se numero de responsavel do pessoa for igual a Configuracoes.quantidadeMaximaResponsavelPorpessoa  OU responsavel.funcao for vazio  OU responsavel.empresa for nulo {  Retorna Falso  Senão {  responsavel.matricula = Gerar matricula (8.6 – Gerar matricula)  enviar a matricula para pessoa.email  incluirResponsavel()  Retornar Verdadeiro  }  } |

**8.3 Cadastrar Autor**

|  |  |
| --- | --- |
| Nome | Cadastrar Autor |
| Parâmetros | autor |
| Fluxo Principal | Se autor.nome não vazio Então  Cadastrar autor  Retorna Verdadeiro  Senão  Retorna Falso |

**8.4 Verificar CPF existente**

|  |  |
| --- | --- |
| Nome | Verificar CPF existente |
| Parâmetros | cpf |
| Fluxo Principal | pessoa = Buscar de pessoa onde pessoa.cpf = cpf  Se pessoa.existente  Retorna pessoa  Senão  Retorna nulo |

**8.5 Buscar Feriados**

|  |  |
| --- | --- |
| Nome | Buscar Feriados |
| Parâmetros |  |
| Fluxo Principal | Retorna feriados onde:  (feriado.dia = data.mes  E feriado.mes = data.mes  E (feriado.ano é nulo OU feriado.ano = data.ano))  OU ((feriado.unidade é nulo)  E((feriado.cidade = unidade.cidade)  OU(feriado.cidade é nulo  E feriado.estado = unidade.cidade.estado)  )  ) |

**8.6 Atualização de atrasos**

|  |  |
| --- | --- |
| Nome | Atualização de atrasos |
| Parâmetros |  |
| Fluxo Principal | T\_atual = data e hora atual  Retiradas = Buscar de retiradas  Onde R.dataDevolucaoEfetiva é nulo  E R.dataDevolucaoEsperada < Dt\_atual  Para cada Ret em Retiradas  Dt\_Esperada = R.devolucao\_esperada  ValorMulta = Buscar precoMulta de unidades onde unidade = R.exemplar.unidade  Para Dt = de DTE até DTA {  F = buscar feriados(DT, R.unidade)  Se F não retornou resultados {  Diaria\_Multa = Busca de DiariaMulta onde  Retirada = Ret e data = DT  Se Diaria\_multa não existente {  Criar Diaria\_Multa,  definir retirada = Ret,  valor = ValorMulta,  data = DT  }  }  }  Atualizar Ret  definir status = Atrasado  Se Ret.Departamento definido {  Atualizar Ret.responsavel, definir status = irregular  Atualizar Ret.pessoa, definir status = Irregular  Se não {  Atualizar Ret.pessoa, definir status = Irregular  }  } |

**8.7 Cadastra Livros**

|  |  |
| --- | --- |
| Nome | Cadastra Livros |
| Parâmetros | livros |
| Fluxo Principal | Se livro.nome não é vazio  E livro.autor não é nulo Então  cadastrar livro  Retorna Verdadeiro  Senão  Retorna Falso |

**8.8 Cadastra Editora**

|  |  |
| --- | --- |
| Nome | Cadastra editora |
| Parâmetros | editora |
| Fluxo Principal | Se editora.nome não é vazio Então  cadastrar editora  Retorna Verdadeiro  Senão  Retorna Falso |

**8.9 Cadastra Edição**

|  |  |
| --- | --- |
| Nome | Cadastra edicao |
| Parâmetros | edicao |
| Fluxo Principal | Se edicao.isbn não esta sendo usada por algum livro  E edicao.ano não for vazio Então  cadastrar editora  Retorna Verdadeiro  Senão  Retorna Falso |

**8.10 Cadastra Exemplares**

|  |  |
| --- | --- |
| Nome | Cadastra Exemplares |
| Parâmetros | lista de exemplares |
| Fluxo Principal | Para cada exemplar da lista de exemplares {  Se exemplar.edicao não for nulo Entao {  Cadastrar exemplar  }  }  Retorna Verdadeiro |

**8.11 Cadastrar Feriado**

|  |  |
| --- | --- |
| Nome | Cadastra Feriado |
| Parâmetros | feriado |
| Fluxo Principal | Se feriado.dia > 0 E feriado.dia < 32  E feriado.mes > 0 E feriado.mes < 13  E feriado.nome não é nulo Então    Para cada retirada de livros {  Se retirada.status for igual a “não entregue” {  E retirada.dataDevolucaoPrevista for maior que a data do feriado  E retirada.dataDevolucaoEfetiva for nulo Então  Se retirada possui uma diária de multa para a data do feriado Então {  Excluir a diária e multa  }  }  }  cadastrar Feriado  Retornar Verdadeiro |

**8.12 Retirada**

|  |  |
| --- | --- |
| Nome | Retirada |
| Parâmetros | retirada |
| Fluxo Principal | Se retirada.pessoa não é nulo  Se retirada.pessoa.status não é irregular  Se numero da reserva do retirada.exemplar é maior que o numero de exemplares disponíveis Então  Se retirada.pessoa tem reserva para retirada.exemplar  E a data da reserva é maior ou igual a data atual Então  retirada.exemplar.status = “retirado”  atualizar retirada.exemplar  cadastra retirada  Retornar Verdadeiro  Senão  Retorna Falso  Senão  Retirada.exemplar.status = “retirado”  atualiza retirada.exemplar  cadastrar retirada  Retornar Verdadeiro  Senão  Retornar Falso  Senão  Se retirada.responsavel não é nulo  Se retirada.responsavel.status não é irregular  Se numero de reserva da retirada.exemplar é maior que o numero de exemplar disponível Então  Se o retirada.responsavel tem reserva para retirada.exemplar  E a data da reserva é maior ou igual a data atual Então  Retirada.exemplar.status = “retirado”  atualiza retirado.exemplar  Cadastra retirada  Retorna Verdadeiro  Senão  Retorna Falso  Senão  retirada.exemplar.status = “retirado” |

**8.13 Pagamento Multa**

|  |  |
| --- | --- |
| Nome | Pagamento Multa |
| Parâmetros | retirada, tipo, observacao |
| Fluxo Principal | Se retirada.status é igual a “entregue com multa” Então {  Se tipo é igual “anular” Então  retirada.observacao = observação  retirada.status = “entregue com multaAnulada”  atualizar retirada  Retornar Verdadeiro  Senão  valorMulta = 0  Para cada diária de retirada.diariasMulta {  Se diária.anulado é falso Então {  valorMulta = ValorMulta + diária.valor  }  }  Retirada.valorPago = valorMulta  Retirada.status = “entregue com multa paga”  }  Senão  Se retirada.status é igual a “extraviado com pagamentos pendentes” então  Se tipo é igual “anular” então {  Retirada.observacao = observação  Retirada.statis = “extraviado com multa anulada”  Atualizar retirada  Retornar Verdadeiro  Fim Se  Senão  Retirada.valorPago  Retirada.exemplar.precoSugerido  Retirada.status = “extraviado com multa paga”  Fim Senão  Senão  Retorno Falso  Fim Se |

**8.14 Cadastro de Rematrícula**

|  |  |
| --- | --- |
| Nome | Cadastro de Rematrícula |
| Parâmetros | Matricula, nota, presenca |
| Fluxo Principal | Se status = “ A ”  Se notas >= 7 E faltas < 25% dos dias letivos então  buscaVagaProximaSerie  Senão  buscaVagaMesmaSerie  Se buscaVagaMesmaSerie = “não encontrado” então  matricula = “recusada”  Incluir ListaEspera(idMatricula)  Se buscaVagaProximaSerie = “encontrado”  turma = matricula  imprimeListaDeMateriais  enviaComprovanteMatriculaEmail  imprimeComprovanteMatricula  Senão  Retorna Falso  Fim Se |

**8.15 Cadastra Matrícula**

|  |  |
| --- | --- |
| Nome | Cadastra Matrícula |
| Parâmetros | pessoa, turma, serie |
| Fluxo Principal | consultaTurmasDisponiveis(idSerie)  Se consultaTurmasDisponiveis(idSerie) = “não encontrado”  Incluir ListaEspera(idMatricula)  Senão  turma = consultaTurmaDisponivel(idSerie)  IncluirMatricula(idAluno, idTurma)  IncluirAluno(idPessoa)  IncluirResposavel(idPessoa)  imprimeMateriaisTurma  enviaComprovanteMatriculaEmail  imprimeComprovanteMatricula  Fim Se |

**8.16 Cadastrar Turma**

|  |  |
| --- | --- |
| Nome | Cadastrar Turma |
| Parâmetros | professor, materia, serie, turma |
| Fluxo Principal | incluirTurma(turma)  incluirMateriasTurma(materia)  Se materia = consultarMateriaProfessor(idMateria)  E professor.Status = “disponivel”  E serie = turma.Serie  alteraMateriasTurma(idTurma)  Senão  recusa turma  Fim Se |

**8.17 Cadastra Materia**

|  |  |
| --- | --- |
| Nome | Cadastra Materia |
| Parâmetros | Funcao, Funcionario |
| Fluxo Principal | Se funcao.nome = “Secretaria” entao  Se materia.nova != consultarMateria(“nomeDaMateria”)  materia.nome = nomeDaMateria  Senão  retorna Materia já existente  Senão  retorna Falso |

**8.18 Cadastra Professor**

|  |  |
| --- | --- |
| Nome | Cadastra Professor |
| Parâmetros | Pessoa |
| Fluxo Principal | Se professor.novo != consultarProfessor(“nomeNovoProfessor”)  idProfessor = idPessoa  prefessor.pis = numero  professor.salario = remuneracao  incluirProfessor();  Senão  retorna Falso |

**8.19 Cadastra Avaliação**

|  |  |
| --- | --- |
| Nome | Cadastra Avaliação |
| Parâmetros | MateriasTurma, Turma, Materia, MateriaProfessor |
| Fluxo Principal | Se idAvalicao = consultarTurma(idAvaliacao) entao  Se funcao.nome = “professor” entao  avaliacao.nome = nome  avaliacao.peso = peso  avaliacao.trimestre = 1 ou 2 ou 3  materiasTurma = idAvaliacao  materiasPorfessor = idAvaliacao  Senão  retorna você não privilégios para esta operação  Senão  retorna Falso |

**8.20 Cadastra Notas**

|  |  |
| --- | --- |
| Nome | Cadastra Notas |
| Parâmetros | Avaliacao, Professor |
| Fluxo Principal | Se usuario = consultarProfessor() entao  incluirNotas(idAvaliacao)  Senão  retorna você não tem permissao para inserir notas |

**8.21 Cadastra Presença**

|  |  |
| --- | --- |
| Nome | Cadastra Presença |
| Parâmetros | Matricula, Turma |
| Fluxo Principal | Se funcao.nome = “Secretaria”  OU funcao.nome = “Professor entao  Seleciona matricula do aluno  dia = diaCorrente  presenca = P ou F  Se calculaPercentualFaltas(IdMatricula) >= 20% entao  contatarResponsavel()  Fim Se  Senão  retorna você não tem permissao para inserir presenças |

**8.22 Cadastra Materiais**

|  |  |
| --- | --- |
| Nome | Cadastra Materiais |
| Parâmetros | Funcao, Funcionario |
| Fluxo Principal | Se funcao.nome = “Secretaria” entao  Se materiais.nova != consultarMateriais(“nomeDoMaterial”)  materiais.nome = nomeDoMaterial  Senão  retorna Material já existente  Senão  retorna Falso |

**8.23 Cadastra Materiais Emprestados**

|  |  |
| --- | --- |
| Nome | Cadastra Materiais Emprestados |
| Parâmetros | Professor, Materia, Turma |
| Fluxo Principal | Se consultarMaterialEmprestado(idMaterial) = status “D” entao  professor = consultarProfessor(IdProfessor)  materia = consultarMateria(IdMateria)  turma = consultarTurma(IdTurma)  data = dia  status = “E”  incluirMaterialEmprestado()  Senão  retorna material em uso |

**8.24 Cadastra Advertencias**

|  |  |
| --- | --- |
| Nome | Cadastra Advertencias |
| Parâmetros | Aluno, Matricula |
| Fluxo Principal | Se IdAluno = IdMatricula entao  motivo = descrição do ocorrido  data = dataCorrente  incluirAdvertencias()  Senão  retorna nao é Aluno ou Aluno ainda não castrado |

**8.25 Cadastra Serie**

|  |  |
| --- | --- |
| Nome | Cadastra Serie |
| Parâmetros | Turma |
| Fluxo Principal | Se NovoNomeSerie != consultaSerie(NovoNomeSerie) entao  nome = NovoNomeSerie  incluirSerie()  Senão  retorna Serie já existente |

**8.26 Cadastrar Login**

|  |  |
| --- | --- |
| Nome | Cadastrar Login |
| Parâmetros | pessoa |
| Fluxo Principal | Se numero informado na tela Login = IdPessoa  E senha informada na tela se Login = pessoa.senha  atorizarLogin(IdPessoa)  Senao  retorna login errado ou inexistente |

**8.27 Cadastrar Senha**

|  |  |
| --- | --- |
| Nome | Cadastrar Senha |
| Parâmetros | pessoa |
| Fluxo Principal | Se pessoa.senha = a nulo entao  Retorna texto aleátorio de Tamanho digito com caracteres de “a” a “z”, “A” a “Z”, “0” a “9” e caracteres especiais ($, @, \*,  &, ^, #). Deve possuir ao menos um caractere de cada tipo (minúsculo, maiúsculo, número e caracter especial).  enviaEmailSenha(pessoa.email)  Senão  retorna já existe senha cadastrada |

**9. ESPECIFICAÇÕES DE INTERFACES**

**Analista:** Moisés Rodrigues Machado

As interfaces apresentadas tem como objetivo facilitar o uso da aplicação. Elas também tem a preocupação com a comunicação de dados e com a segurança do banco de dados. A entrada no sistema é feito através de login e por senha. Para cada usuário do sistema é atribuído um tipo, com isso é possível identificar se o usuário é um professor, um funcionário ou aluno. Com base nisso, as permissões de acessos são baseadas nos tipos de cada usuário. Os menus são por permissões, só são exibidos menus que determinado usuário precisa ou lhe é permitido visualizar.

* 1. **Nome da Interface:** Adição ou Edição de Dados.
     1. **Descrição da Interface:** Esta é uma interface genérica para adição ou edição de dado.
     2. **Complexidade:** Baixa
     3. **Botões de ação**

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
| **Label** | **Ação** | **Pós-ação** | **Classe/método** |
| Salvar | Salva os dados  informados | Retorna para página de listagem | incluir() |

* + 1. **Campos de Registro**
       1. **Cadastro de Pessoas**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Nome | nome | varchar (45) | textfield | null | sim | - |
| Cpf | cpf | varchar (45) | textfield | null | sim | - |
| Senha | senha | varchar (45) | password | null | sim | - |
| Sexo | sexo | varchar (1) | checkbox | null | não | - |
| Data Nascimento | dataNascimento | date | date picker | null | não | - |
| Email | email | varchar (45) | textfield | null | sim | - |
| Telefone Fixo | telefoneFixo | int | textfield | null | não | - |
| Telefone Celular | telefoneCelular | int | textfield | null | não | - |
| Status | status | enum | drop list | null | sim | Drop list com uma lista se esta A =. ativo ou I => inativo |
| Endereço | endereco | varchar (80) | textfield | null | não | - |

* + - 1. **Cadastro de Funcionários**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Pis | pis | int | textfield | null | sim | - |
| Função | Funcao\_idFuncao | int | drop list | null | sim | Drop list com uma lista de funções |
| Pessoa | Pessoa\_idPessoa | int | drop list | null | sim | Drop list com uma lista de pessoas |
| Salário | salario | float | textfield | null | sim | - |

* + - 1. **Cadastro de Professores**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Pis | pis | int | textfield | null | sim | - |
| Pessoa | Pessoa\_idPessoa | int | drop list | null | sim | Drop list com uma lista de pessoas |
| Salário | salario | float | textfield | null | sim | - |

* + - 1. **Cadastro de Função**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Nome | nome | varchar (45) | textfield | null | sim | - |

* + - 1. **Cadastro de Responsável**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Empresa | pis | varchar (45) | textfield | null | sim | - |
| Função | funcao | varchar (45) | textfield | null | sim | - |
| Pessoa | Pessoa\_idPessoa | int | drop list | null | sim | Drop list com uma lista de pessoas |

* + - 1. **Cadastro de Alunos**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Observações | observacoes | varchar (45) | textfield | null | não | - |
| Pessoa | Pessoa\_idPessoa | int | drop list | null | sim | Drop list com uma lista de pessoas |

* + - 1. **Cadastro de Reservas**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Data Reserva | dataReserva | date | date picker | null | sim | - |
| Data Agenda | dataAgenda | date | date picker | null | sim | - |
| Matricula | Matricula\_idMatricula | int | drop list | null | sim | Drop list com uma lista de alunos |

* + - 1. **Cadastro de Salas**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Número | numero | int | textfield | null | sim | - |
| Capacidade | capacidade | int | textfield | null | sim | - |

* + - 1. **Cadastro de Livros**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Nome | nome | varchar (45) | textfield | null | sim | - |
| Ano | ano | int | textfield | null | sim | - |

* + - 1. **Cadastro de Exemplar**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Estande | estande | int | textfield | null | sim | - |
| Prateleira | prateleira | int | textfield | null | sim | - |
| Status | status | enum | drop list | D | sim | Drop list com uma lista se esta D =. disponível ou E => emprestado |
| Danificado | danificado | boolean | checkbox | FALSE | sim | - |
| Livro | Livros\_idLivros | int | drop list | null | sim | Drop list com uma lista de livros |
| Tipo | TipoExemplar\_idTipoExemplar | int | drop list | null | sim | Drop list com uma lista de tipos de exemplares |

* + - 1. **Cadastro de Tipo de Exemplares**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Nome | nome | varchar (45) | textfield | null | sim | - |

* + - 1. **Cadastro de Autor**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Nome | nome | varchar (45) | textfield | null | sim | - |

* + - 1. **Cadastro de Tipo de Multas**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Nome | nome | varchar (45) | textfield | null | sim | - |

* + - 1. **Cadastro de Séries**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Nome | nome | varchar (45) | textfield | null | sim | - |

* + - 1. **Cadastro de Turma**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Turno | turno | enum | drop list | null | sim | Drop list com uma lista de turno (M => manha, T => tarde, N => noite) |
| Série | Serie\_idSerie | int | drop list | null | sim | Drop list com uma lista de séries |
| Turma | numeroTurma | int | textfield | null | sim | - |
| Vagas | vagas | int | textfield | null | sim | - |
| Sala | Sala\_idSala | int | drop list | null | sim | Drop list com uma lista de salas |
| Ano | ano | int | textfield | null | sim | - |

* + - 1. **Cadastro de Tipo de Materiais**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Nome | nome | varchar (45) | textfield | null | sim | - |

* + - 1. **Cadastro de Materiais**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Nome | nome | varchar (45) | textfield | null | sim | - |
| Tipo | TipoMateriais\_idTipoMateriais | int | drop list | null | sim | Drop list com uma lista de tipos de matérias |

* + - 1. **Cadastro de Pessoas Cadastro de Advertência**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Motivo | motivo | text | textfield | null | sim | - |
| Data | data | int | drop list | null | sim | Drop list com uma lista de funções |
| Aluno | Matricula\_idMatricula | int | drop list | null | sim | Drop list com uma lista de alunos |

* + - 1. **Cadastro de Matérias-Professores**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Matéria | Materia\_idMateria | int | drop list | null | sim | Drop list com uma lista de matérias |
| Professor | Professores\_idProfessores | int | drop list | null | sim | Drop list com uma lista de professores |

* + - 1. **Cadastro de Matérias-Turmas**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Professor | idProfessor | int | drop list | null | sim | Drop list com uma lista de professores |
| Matéria | idMateria | int | drop list | null | sim | Drop list com uma lista de matérias |
| Turma | idTurma | int | drop list | null | sim | Drop list com uma lista de turmas |

* + - 1. **Cadastro de Feriados**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Dia | dia | int | text field | null | sim | - |
| Mês | mes | int | text field | null | sim | - |
| Ano | ano | int | text field | null | não | - |
| Tipo | tipo | enum | checkbox | null | não | Checkbox com 2 campos: Estadual (que deve ter valor de “E”) e Municipal (que deve ter valor de “M”) |

* + - 1. **Cadastro de Matéria**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Nome | nome | varchar (45) | textfield | null | sim | - |

* + - 1. **Cadastro de Avaliação**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Nome | nome | varchar (45) | textfield | null | sim | - |
| Peso | peso | float | textfield | null | sim | - |
| Tipo | tipo | enum | textfield | null | não | Drop list com uma lista se esta N =. normal ou E => exame |
| Trimestre | trimestre | int | textfield | null | sim | - |
| Professor | MateriasTurmaa\_idProfessores | int | drop list | null | sim | Drop list com uma lista com os professores |
| Matéria | MateriasTurmaa\_idMateria | int | drop list | null | sim | Drop list com uma lista com as matérias |
| Turma | MateriasTurmaa\_idTurma | int | drop list | null | sim | Drop list com uma lista com as turmas |

* + - 1. **Cadastro de Matricula**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Observações | observacoes | varchar (45) | textfield | null | não | - |
| Aluno | Aluno\_idAlunos | int | drop list | null | sim | Drop list com uma lista de alunos |
| Turma | Turma\_idTurma | int | drop list | null | sim | Drop list com uma lista de turma |

* + - 1. **Cadastro de Materiais Emprestados**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Dia | dia | date | date picker | null | sim | - |
| Status | status | enum | textfield | D | sim | Drop list com uma lista se esta E =. emprestado ou D => Disponivel |
| Materiais Turma | MateriaisTurma\_idMateriaisTurma | int | drop list | null | sim | Drop list com uma lista com as matérias |
| Professor | MateriasTurmaa\_idProfessores | int | drop list | null | sim | Drop list com uma lista com os professores |
| Matéria | MateriasTurmaa\_idMateria | int | drop list | null | sim | Drop list com uma lista com as matérias |
| Turma | MateriasTurmaa\_idTurma | int | drop list | null | sim | Drop list com uma lista com as turmas |

* 1. **Nome da Interface:** Listagem de Dados.
     1. **Descrição da Interface:** Esta é uma interface genérica para consulta e listagem de dados. Cada consulta atualizará a lista que se encontra abaixo. Para a consulta será escolhido 3 campos para fazer as buscas.
     2. **Complexidade:** Baixa
     3. **Botões de ação**

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
| **Label** | **Ação** | **Pós-ação** | **Classe/método** |
| Editar | Realiza edição de dado | Abre página de edição de dados | alterar() |
| Deletar | Exclui dado | Atualiza pagina de listagem de dados | apagar() |
| Novo | Cadastra um dado novo | Abre página de cadastro de dados | incluir() |
| Consultar | Fazer busca de acordo com os filtros indicados | Atualiza lista de dados | consultar() |

* + 1. **Campos de Registro**
       1. **Listagem de dados**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| TipoCampo | filtro | varchar (45) | drop list | null | não | Drop list com uma lista de possiveis consultas: Contem; Igual a; Diferente de. |

* 1. **Nome da Interface:** Login de usuários.
     1. **Descrição da Interface:** Esta é a interface de login.
     2. **Complexidade:** Baixa
     3. **Botões de ação**

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
| **Label** | **Ação** | **Pós-ação** | **Classe/método** |
| Login | Realiza ação de login | Abri página inicial | login() |

* + 1. **Campos de Registro**
       1. **Login**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Usuário | usuario | varchar (45) | textfield | null | sim | - |
| Senha | senha | varchar (200) | textfield | null | sim | - |

* 1. **Nome da Interface:** Cadastro de notas
     1. **Descrição da Interface:** Esta é a interface de cadastro de notas de alunos, ao abrir a tela, esta deve vir com uma lista de alunos que realizou a avaliação.
     2. **Complexidade:** Baixa
     3. **Botões de ação**

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
| **Label** | **Ação** | **Pós-ação** | **Classe/método** |
| Salvar | Salva os dados  informados | Retorna para página de listagem | incluir() |

* + 1. **Campos de Registro**
       1. **Notas**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Nota | nota | float | textfield | null | sim | - |

* 1. **Nome da Interface:** Cadastro de presenças
     1. **Descrição da Interface:** Esta é a interface de cadastro de notas de alunos, ao abrir a tela, esta deve vir com uma lista de alunos de uma determinada turma e com todos os alunos com presença.
     2. **Complexidade:** Baixa
     3. **Botões de ação**

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
| **Label** | **Ação** | **Pós-ação** | **Classe/método** |
| Salvar | Salva os dados  informados | Retorna para página de listagem | incluir() |

* + 1. **Campos de Registro**
       1. **Notas**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Atributo** | **Tipo Dado** | **Tipo Campo** | **Valor Inicial** | **Obrigatório** | **Comentários** |
| Presença | presenca | enum | checkbox | null | sim | Checkbox com 2 campos: Presença (que deve ter valor “P”) e Ausente ( que deve ter valor “F”) |
| Data | dia | date | date picker | dia atual | sim | Deve vir com o dia atual preenchido |

* 1. **Nome da Interface:** Tela de inicio
     1. **Descrição da Interface:** Esta é a interface de início, logo após ser feito o login, o sistema deve levar o usuário para esta tela. A tela deve mostrar uma lista com os dados da tabela pessoa.
     2. **Complexidade:** Baixa
     3. **Botões de ação**

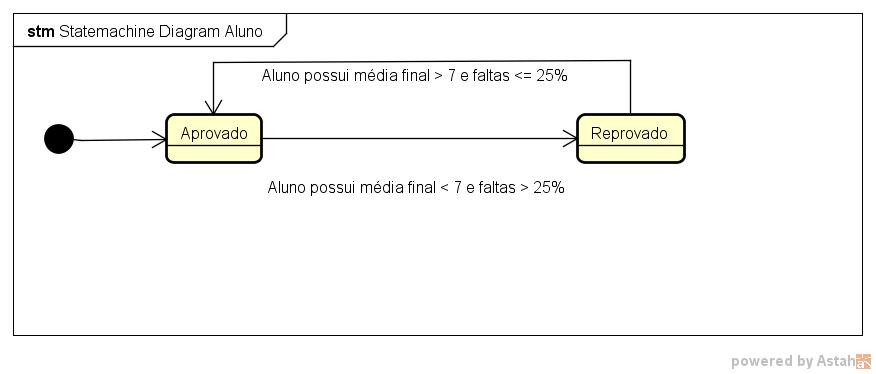
|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
| **Label** | **Ação** | **Pós-ação** | **Classe/método** |
| Editar | Leva o usuário para a tela de edição de pessoa | - | alterar() |

* 1. **Nome da Interface:** Menu
     1. **Descrição da Interface:** Esta é a interface do menu. Na primeira aba o menu deve listar todos as possíveis telas de listagens. Na aba “usuário”, deve listar as opções de início e logout.
     2. **Complexidade:** Baixa
     3. **Botões de ação**

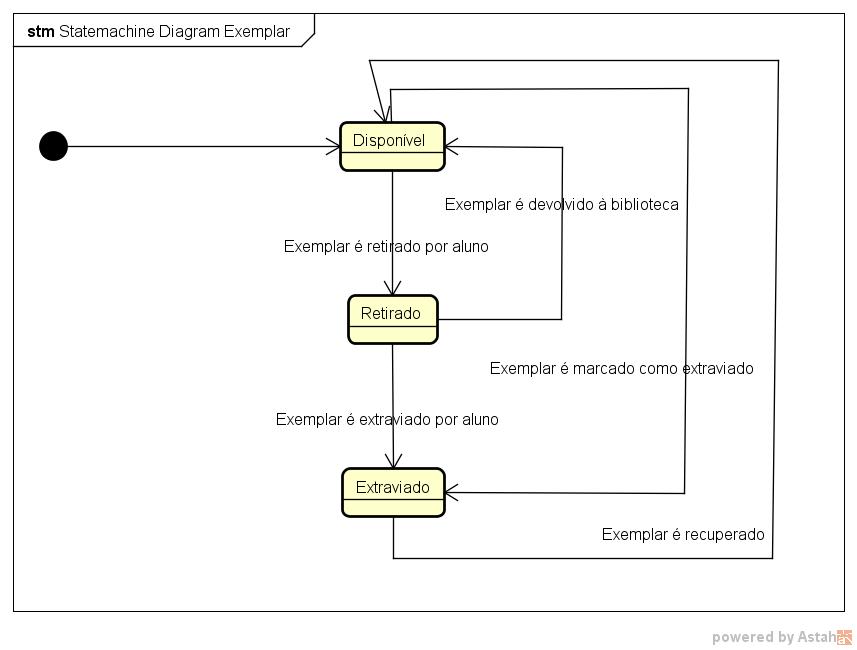
|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
| **Label** | **Ação** | **Pós-ação** | **Classe/método** |
| Menu | Abre sub-menus que levam para telas de listagem | - | - |
| Item (n) | Leva para a pagina de listagem que informa o label | Abre pagina de listagem indicada no label | consultar() |
| Inicio | Leva para tela de inicio | Abre pagina de início | - |
| Logout | Sai do sistema | Leva para tela de login | - |

**10. DIAGRAMA DE ESTADOS**

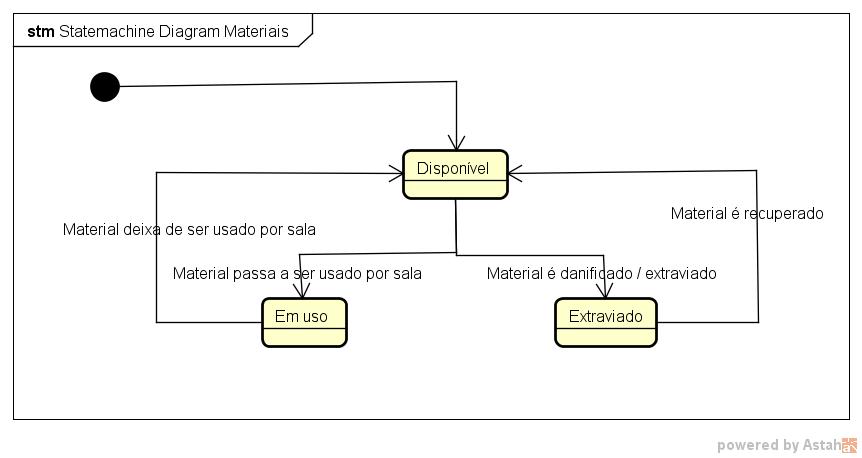
**10.1 Aluno**



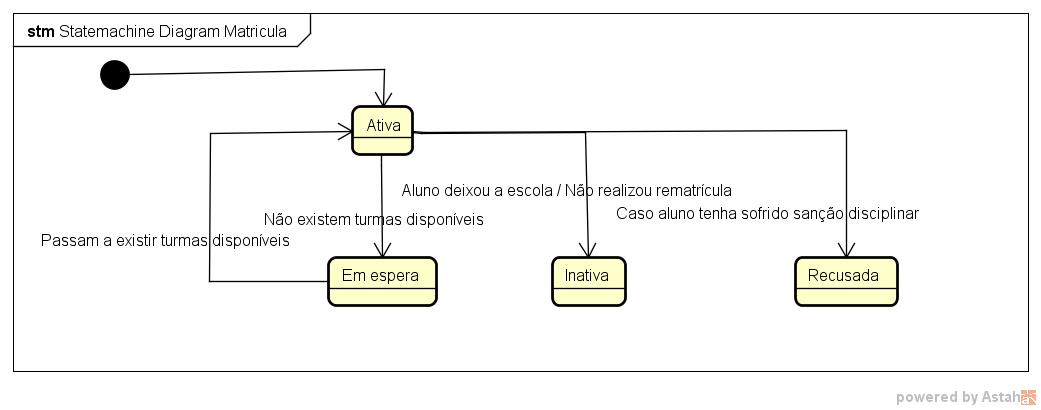
**10.2 Exemplar**



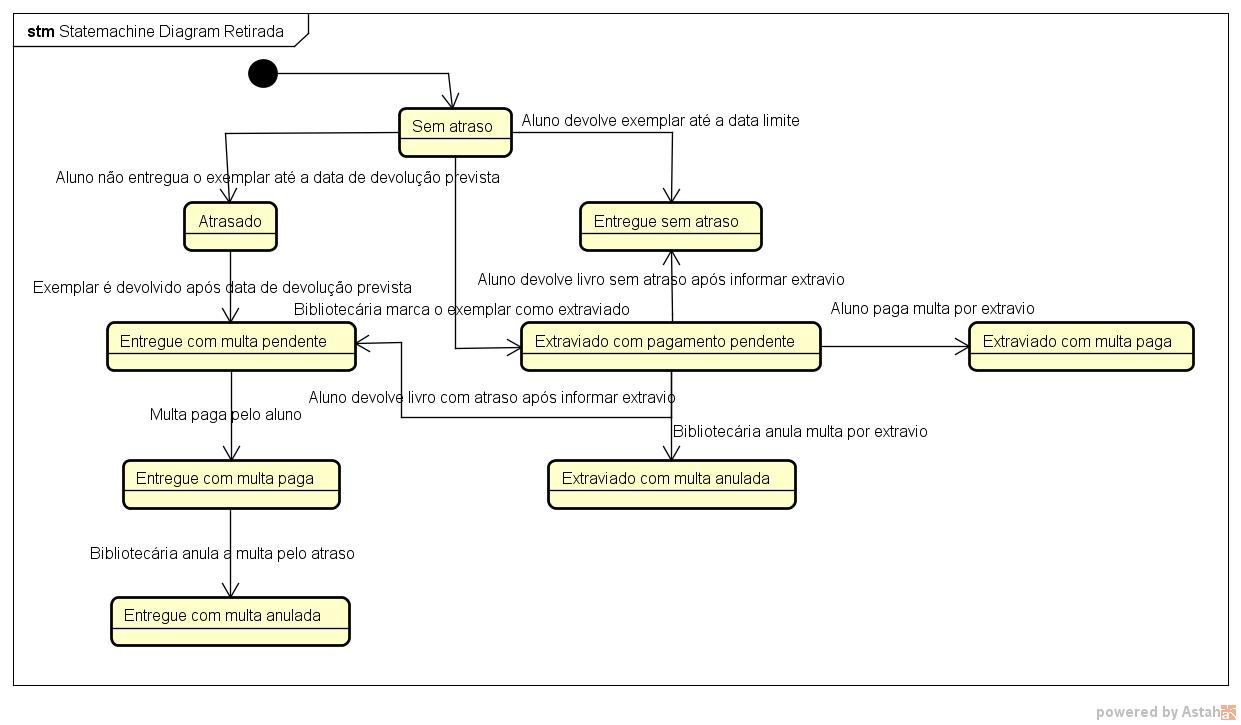
**10.3 Materiais**



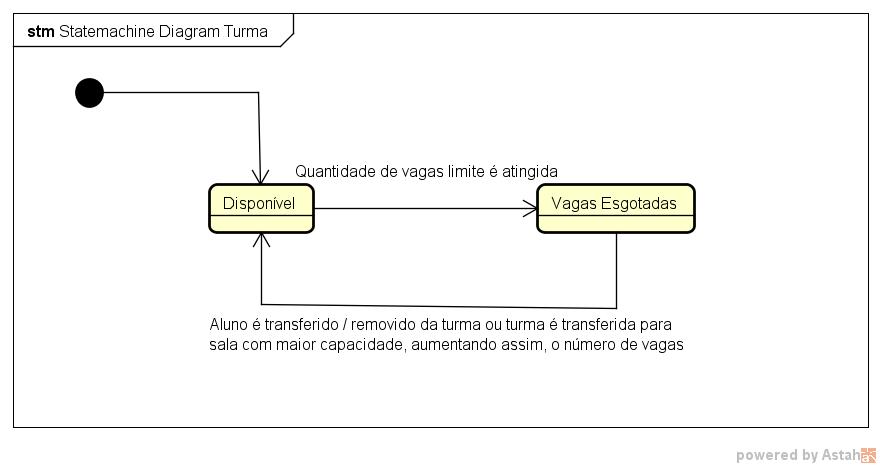
**10.4 Matricula**



**10.5 Retirada**



**10.6 Turma**



**11. TECNOLOGIAS UTILIZADAS**

**11.1 Linguagem e framework**

O sistema será desenvolvido utilizando o PHP versão 5.6.9. Recomenda-se a utilização de Windows para rodar o servidor da aplicação devido a grande compatibilidade com várias extensões do PHP. As extensões necessárias para rodar este projeto são:

Core, date, ereg, libxml, openssl, pcre, zlib, bcmath, bz2, calendar, ctype, curl, dba, dom, hash, fileinfo, filter, ftp, gd, gettext, SPL, iconv, session, json, idap, mbstring, mcrypt, standart, PDO, pdo\_pgsql, Phar, Posix, Reflection, imap, shmop, simpleXML, soap, sockets, exit, sybase\_ct, sysvsem, sysvshm, tokenizer, wddx, xml, xmlreader, xmlwriter, xsl, zip, apache2handler, mhash

Para estruturar o layout do sistema será feito o uso do framework Bootstrap que está na versão 3.3.6, o qual disponibiliza múltiplos componentes e convenções de classes para manter o site responsivo.

**11.2 Sistema de Gerenciamento do Banco de Dados**

O sistema de gerenciamento do banco de dados utilizado será o Mysql, atualmente na versão 5.7.Este sistema possui ótima estabilidade, com suporte a transações que respeitem os princípios ACID (atomicidade, consistência, isolamento e durabilidade).