SERVIÇO NACIONAL DE APRENDIZAGEM COMERCIAL

SENAC

CURSO DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

PROJETO INTEGRADOR III: DESENVOLVIMENTO DE SISTEMAS ORIENTADO A OBJETOS

Professor Enoque Felipe dos Santos Leal

Professor Julio Cesar Severino

Integrantes do grupo:

Suelen Gonçalves Sales

Thais Gabriel Almeida Gai

Fernando Quintanilha Namur

Pierre Leon Fernandes Silva

Guilherme Martins Barros Vital

SUMÁRIO

1 INTRODUÇÃO - VISÃO GERAL DA PROPOSTA

1.1 Contextualização e motivação

A gestão eficiente de dados é essencial para o funcionamento de uma grande universidade. Este projeto tem como objetivo desenvolver um sistema que simplifique o cadastro e a gestão de diferentes tipos de usuários da universidade SENAC, como alunos, professores, fornecedores e outros. A principal motivação é aumentar a eficiência e a precisão na gestão de dados, empregando técnicas modernas de engenharia de software.

1.2 Objetivos

- Desenvolver um sistema de gestão de dados utilizando orientação a objetos.

- Criar diagramas UML para modelar o sistema.

- Planejar o desenvolvimento do sistema considerando todo o ciclo de vida do software.

2. PLANEJAMENTO PARA O DESENVOLVIMENTO DA SOLUÇÃO PROPOSTA

2.1 Ciclo de Vida do Desenvolvimento

O ciclo de vida escolhido para o desenvolvimento da solução é o modelo incremental, que permite a entrega de partes funcionais do sistema em ciclos curtos. Isso facilita a validação contínua e a adaptação às necessidades dos usuários.

2.2 Premissas

- O sistema deve ser desenvolvido utilizando uma linguagem de programação orientada a objetos.
- A interface deve ser intuitiva e fácil de usar.
- O sistema deve ser escalável para suportar muitos usuários.

2.3 Requisitos/História do Usuário

Requisito 1: O sistema deve permitira interação de Pessoas Físicas.

Requisito 2: O sistema deve permitir a interação de Pessoas Jurídicas.

Requisito 3: O sistema deve permitir a interação de Professores.

Requisito 4: O sistema deve permitir a interação de Fornecedores.

Requisito 5: O sistema deve permitir a interação de Alunos.

Requisito 6: O sistema deve permitir a interação de Funcionários

2.4 Planejamento

Etapa 1: Levantamento de Requisitos (1 semana)

Etapa 2: Modelagem UML (2 semanas)

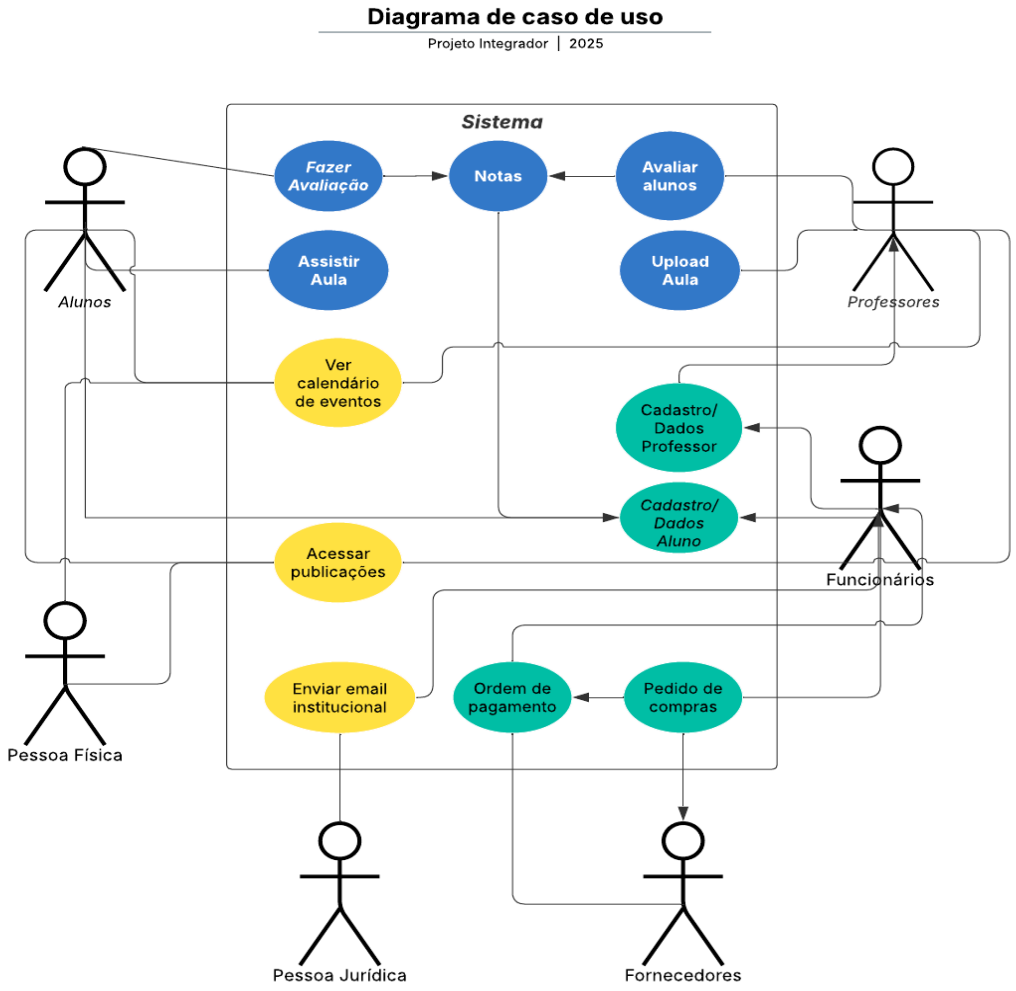
Etapa 3: Desenvolvimento do Protótipo Funcional (3 semanas)

Etapa 4: Validação de Usabilidade (1 semana)

Etapa 5: Implementação Final (4 semanas)

3. MODELO DE PERSISTÊNCIA DE DADOS ORIENTADOS A OBJETOS

3.1 Diagrama de Casos de Uso



3.2 Descrição dos Cenários de Casos de Uso

Caso de Uso 1: Cadastrar Pessoa Física

Ator Principal: Administrador

Pré-condição: O usuário deve estar autenticado com permissões de cadastro, e a tela de cadastro de Pessoa Física deve estar acessível.

Cenário Principal:

1. O administrador acessa a funcionalidade "Cadastrar Pessoa Física".

2. Preenche os campos obrigatórios: nome completo, CPF, data de nascimento, endereço e telefone.

3. Clica em “Salvar”.

4. O sistema valida os dados fornecidos.

5. O sistema armazena os dados no banco e exibe uma mensagem de sucesso.

Cenário Alternativo 1: O CPF informado é inválido.

→ O sistema exibe mensagem de erro e impede o cadastro.

Cenário Alternativo 2: Campos obrigatórios não preenchidos.

→ O sistema exibe alerta solicitando o preenchimento.

Cenário Alternativo 3: O usuário cancela a operação.

→ Nenhuma alteração é realizada no sistema.

Pós-condição: Pessoa Física cadastrada com sucesso ou nenhuma alteração realizada, em caso de erro ou cancelamento.

Caso de Uso 2: Cadastrar Pessoa Jurídica

Ator Principal: Administrador

Pré-condição: O usuário deve estar autenticado e com acesso à funcionalidade de cadastro de Pessoa Jurídica.

Cenário Principal:

1. O administrador acessa a funcionalidade "Cadastrar Pessoa Jurídica".
2. Preenche os campos obrigatórios: razão social, CNPJ, inscrição estadual, endereço e telefone.
3. Clica em “Salvar”.
4. O sistema valida o CNPJ e os demais campos obrigatórios.
5. Os dados são armazenados e uma mensagem de sucesso é exibida.

Cenário Alternativo 1: CNPJ duplicado ou inválido.

→ O sistema impede o cadastro e informa o erro.

Cenário Alternativo 2: Campos obrigatórios não preenchidos.

→ O sistema solicita o preenchimento antes de permitir o salvamento.

Cenário Alternativo 3: Cancelamento da operação pelo usuário.

→ Nenhuma modificação ocorre no sistema.

Pós-condição: Pessoa Jurídica cadastrada ou operação cancelada sem alterações.

Caso de Uso 3: Cadastrar Professor

Ator Principal: Administrador

Pré-condição: O administrador deve estar autenticado, e a tela de cadastro de Professor deve estar acessível.

Cenário Principal:

1. O administrador acessa a funcionalidade “Cadastrar Professor”.

2. Preenche os campos obrigatórios: nome completo, CPF, formação acadêmica, área de atuação, e-mail e telefone.

3. Clica em “Salvar”.

4. O sistema valida os dados e os armazena.

5. O sistema exibe a mensagem de cadastro realizado com sucesso.

Cenário Alternativo 1: CPF inválido ou já cadastrado.

→ O sistema bloqueia a operação e apresenta uma mensagem de erro.

Cenário Alternativo 2: Campos obrigatórios ausentes.

→ O sistema impede o cadastro e solicita correção.

Cenário Alternativo 3: O administrador cancela a operação.

→ Nenhum dado é armazenado.

Pós-condição: Professor cadastrado com sucesso ou nenhuma alteração realizada.

Caso de Uso 4: Cadastrar Aluno

Ator Principal: Administrador

Pré-condição: O administrador deve estar autenticado e a Pessoa Física correspondente ao aluno deve estar previamente cadastrada no sistema.

"Cenário Principal:"

1. **O administrador acessa a funcionalidade “Cadastrar Aluno”.**
2. **Seleciona uma Pessoa Física já registrada.**
3. **Preenche os campos: matrícula, curso, período e data de ingresso.**
4. **Clica em “Salvar”.**
5. **O sistema valida as informações e salva os dados.**
6. **O sistema exibe mensagem de cadastro realizado com sucesso.**

Cenário Alternativo 1: Pessoa Física não encontrada no sistema.
→ O sistema bloqueia o cadastro e solicita o registro da Pessoa Física antes de continuar.

Cenário Alternativo 2: Matrícula já utilizada por outro aluno.
→ O sistema impede o cadastro e solicita uma nova matrícula.

Cenário Alternativo 3: O administrador cancela o processo.
→ Nenhuma alteração é realizada.

Pós-condição: Aluno cadastrado com sucesso ou nenhuma modificação no sistema em caso de erro ou cancelamento.

Caso de Uso 5: Cadastrar Fornecedor

Ator Principal: Administrador

Pré-condição: O administrador deve estar autenticado e a Pessoa Jurídica correspondente ao fornecedor deve estar previamente cadastrada.

Cenário Principal:

1. **O administrador acessa a funcionalidade "Cadastrar Fornecedor".**
2. **Seleciona uma Pessoa Jurídica existente no sistema.**
3. **Preenche os dados de fornecedor: tipo de produto, status do contrato.**
4. **Clica em "Salvar".**
5. **O sistema valida as informações e salva o cadastro.**
6. **Uma mensagem de confirmação é exibida.**

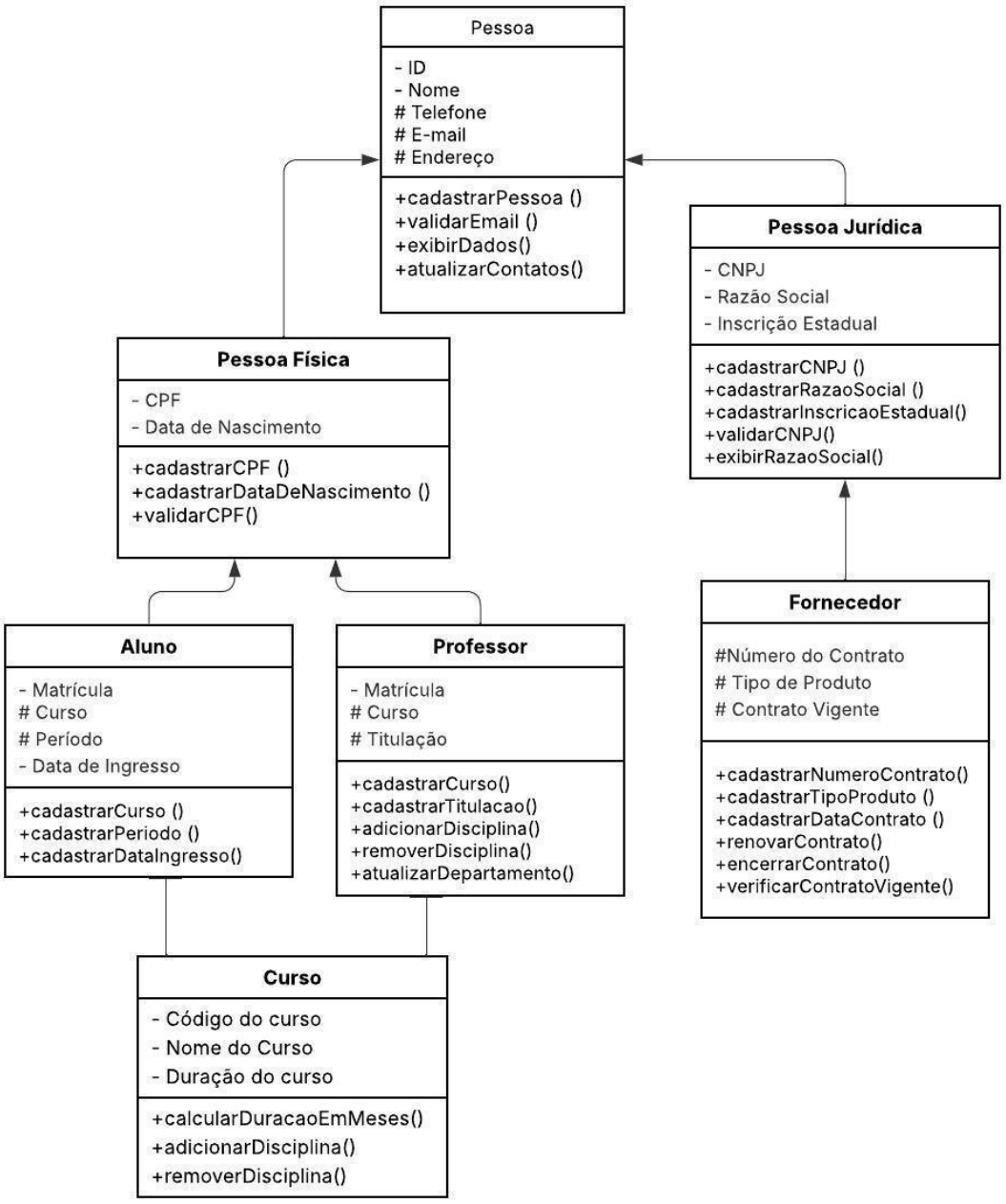
Cenário Alternativo 1: Pessoa Jurídica não cadastrada.
→ O sistema impede o cadastro e solicita o preenchimento dos dados da empresa primeiro.

Cenário Alternativo 2: Tipo de produto não informado.
→ O sistema exibe mensagem solicitando o preenchimento obrigatório.

Cenário Alternativo 3: O administrador cancela o processo.
→ Nenhuma alteração é realizada no sistema.

Pós-condição: Fornecedor registrado com sucesso ou nenhuma alteração realizada, em caso de erro ou cancelamento.

4. Diagramas de Classes UML



4.1 Descrição textual das classes

Lista das classes principais:

- Pessoa (classe genérica)

Atributos:

-id: int

- nome: String

- telefone: String

- email: String

- endereco: String

atualizarContato(telefone, email, endereco): void

validarEmail(): boolean

exibirDados(): String

- Pessoa Física (herda de Pessoa)

- cpf: String

- dataNascimento: Date

- sexo: String

validarCPF(): boolean

- PessoaJurídica (herda de Pessoa)

- cnpj: String

- razaoSocial: String

- inscricaoEstadual: String

validarCNPJ(): boolean

exibirRazaoSocial(): String

- Aluno (herda de PessoaFísica)

- matricula: String

- curso: String

- periodo: String

- dataIngresso: Date

- Professor (herda de PessoaFísica)

- matricula: String

- departamento: String

- titulacao: String

adicionarDisciplina(disciplina): void

removerDisciplina(disciplina): void

atualizarDepartamento(departamento): void

- Fornecedor (herda de PessoaJurídica)

- tipoProduto: String

- contratoVigente: Boolean

renovarContrato(): void

encerrarContrato(): void

fornecerProduto(produto, quantidade): void

verificarContratoVigente(): boolean

- Curso

- codigoDoCurso: String

-nomeDoCurso: String

-duracao: Int

calcularDuracaoEmMeses(): Int

4.2 Proposta de implementação

Proposta de implementação na linguagem de programação proposta na disciplina Programação Orientada a Objetos.

5. PROTÓTIPO FUNCIONAL E EXPERIMENTOS DE USABILIDADE

5.1 Protótipo Funcional

O protótipo funcional será desenvolvido utilizando uma ferramenta de prototipagem, como o Figma ou o Adobe XD. A interface deve ser intuitiva, com menus claros e acessíveis para cada tipo de cadastro.

5.2 Experimento de Usabilidade

O experimento de usabilidade envolverá testes com usuários reais para avaliar a facilidade de uso e a eficiência da interface. Serão coletados feedbacks para melhorias contínuas.

CONCLUSÃO

Com base no que foi desenvolvido até aqui, é possível afirmar que os objetivos iniciais do projeto foram atendidos. O sistema proposto foi devidamente modelado com base nos conceitos de orientação a objetos, utilizando os diagramas UML exigidos, e o planejamento contemplou todas as etapas do ciclo de vida do software.

A principal dificuldade enfrentada foi organizar os requisitos de maneira clara para evitar sobreposição de funcionalidades nos cadastros. Ainda assim, o trabalho em grupo permitiu a divisão eficaz das tarefas e o aprofundamento dos conhecimentos em análise de sistemas.

Como continuidade, propõe-se a implementação do sistema modelado, utilizando os conceitos de POO e bancos de dados, além de ampliar os testes de usabilidade para simular cenários reais da rotina administrativa de uma universidade.

REFERÊNCIAS

PRESSMAN, Roger S. *Engenharia de Software: uma abordagem profissional*. 7. ed. São Paulo: McGraw-Hill, 2016.

LARMAN, Craig. *Utilizando UML e Padrões*. 3. ed. Porto Alegre: Bookman, 2007.

SOMMERVILLE, Ian. *Engenharia de Software*. 10. ed. São Paulo: Pearson, 2019.