



UnB

Departamento de
Ciência da Computação

Disciplina Técnicas de Programação 1

Título do Projeto

Grupo X – 2025-2

Nome Completo Aluno 1 – Matrícula – Email

Nome Completo Aluno 2 – Matrícula – Email

Nome Completo Aluno 3 – Matrícula – Email

Nome Completo Aluno 4 – Matrícula – Email

Profa. Roberta Barbosa Oliveira

Resumo

O resumo do relatório deve abordar brevemente a **contextualização do problema**, apresentando a necessidade do software, por exemplo, como ele pode otimizar processos, organizar cadastros ou melhorar o atendimento de um determinado negócio. Deve também apresentar o **objetivo do trabalho**, descrevendo o propósito do software e de que maneira ele busca atender às demandas identificadas. Em seguida, é importante destacar a **solução proposta**, enfatizando as principais etapas de desenvolvimento realizadas, bem como as **tecnologias utilizadas**, indicando as ferramentas de desenvolvimento, linguagens e recursos empregados. O resumo deve abordar ainda as **técnicas de programação aplicadas**, como classes, objetos, herança, polimorfismo, modularização, tratamento de exceções e práticas de design, além de apresentar os **resultados alcançados**, com a descrição das funcionalidades mais relevantes do software e das características da solução, como confiabilidade, integração e usabilidade. Por fim, deve-se concluir ressaltando a **conclusão**, evidenciando a relevância do software e sua contribuição para o contexto em que foi desenvolvido.

1 Introdução

As próximas seções devem apresentar o contexto em que o trabalho foi desenvolvido, descrevendo as necessidades identificadas e a motivação para a criação do software. Também devem ser definidos os objetivos de desenvolvimento, que estabelecem o propósito do software e os resultados esperados com sua implementação. Além disso, devem detalhar os requisitos funcionais, que correspondem às funcionalidades que o software deve oferecer, e os requisitos não funcionais, que abrangem suas características de desempenho, confiabilidade, usabilidade e segurança. Por fim, devem descrever as regras de negócio, que representam as condições, restrições e normas a serem seguidas durante o funcionamento do software, assegurando que ele atenda adequadamente às necessidades do domínio aplicado.

1.1 Contexto do Projeto

O contexto representa o ambiente em que o projeto se insere. Deve descrever os problemas ou demandas existentes e a justificativa para o desenvolvimento do software. É importante apresentar o domínio de aplicação e indicar de que maneira a solução proposta busca atender às demandas. A contextualização deve mostrar a relevância do software no ambiente considerado, destacando como ele pode contribuir, por exemplo, para otimizar processos, organizar informações, aumentar a eficiência operacional ou melhorar a qualidade dos serviços prestados.

1.2 Objetivos de Desenvolvimento

Nesta seção devem definir o objetivo geral e os objetivos específicos do projeto. O objetivo geral refere-se à finalidade principal do software a ser desenvolvido, enquanto os objetivos específicos detalham funcionalidades ou características desejadas. O texto deve indicar quais processos serão automatizados, quais funcionalidades essenciais serão implementadas e de que forma a solução contribuirá para o ambiente em questão. Além disso, deve-se ressaltar como contribuição esperada a aplicação de conceitos de programação orientada a objetos e técnicas de design de software para garantir uma solução modular, extensível, confiável e alinhada às necessidades do domínio abordado.

1.3 Requisitos Funcionais

Devem ser listadas e descritas as funcionalidades que o sistema deve oferecer, ou seja, o que ele deve fazer. Cada requisito deve ser escrito de maneira clara e objetiva, representando serviços, operações ou comportamentos esperados; por exemplo, cadastrar usuários, gerar relatórios, registrar transações, realizar buscas.

1.4 Requisitos não Funcionais

Aqui devem ser apresentados os requisitos que descrevem como o sistema deve funcionar, ou seja, suas características de qualidade. Incluem aspectos como desempenho, segurança, confiabilidade, usabilidade, manutenibilidade, escalabilidade e portabilidade.

1.5 Regras de negócio

Devem ser especificadas as condições, restrições e normas do domínio da aplicação que impactam diretamente o funcionamento do sistema. São regras que refletem o funcionamento real do ambiente em que o software será utilizado; por exemplo, um funcionário só pode ser promovido após um ano de contrato, um candidato só pode participar de um processo seletivo se preencher determinados requisitos, o cálculo de férias deve considerar a legislação trabalhista vigente.

1.6 Estrutura do Relatório

As próximas seções do relatório detalham a análise e o desenvolvimento do software implementado. Na Seção 2 são apresentadas a modelagem, a estrutura, os módulos, a arquitetura

e as funcionalidades desenvolvidas, descrevendo como cada etapa do projeto contribuiu para a construção do software. Na Seção 3, discute-se a análise das funcionalidades desenvolvidas, destacando resultados alcançados, integração entre os módulos, confiabilidade, usabilidade e outras características do software, além de eventuais limitações ou desafios encontrados durante o desenvolvimento. Por fim, a Na Seção 4 sintetiza a relevância do trabalho desenvolvido, avaliando seu impacto no contexto aplicado e apontando possíveis melhorias e extensões para evoluções futuras.

Nos relatórios parciais, devem ser incluídas apenas as partes que já foram desenvolvidas até o momento. Cada entrega deve corresponder a uma etapa específica da seção de Solução Proposta: Estrutura Inicial, Design e Modelagem, Implementação da Lógica do Software e Integração e Navegabilidade. Nessas versões, é obrigatório incluir o Resumo e a Introdução, enquanto a Solução Proposta deve refletir a etapa de desenvolvimento atual, detalhando funcionalidades, diagramas e técnicas de programação aplicadas. Alterações ou melhorias em relação a versões anteriores devem ser documentadas na versão atual.

No relatório final, o documento deve ser revisado e atualizado, mantendo apenas o conteúdo referente ao que foi de fato desenvolvido, contemplando todas as implementações, funcionalidades, diagramas, módulos, técnicas de programação aplicadas e integração final. Além do Resumo, da Introdução e da Solução Proposta, o relatório final deve incluir as seções Resultados e Discussão, e Conclusão e Evolução Futura, para refletir a solução completa desenvolvida pelo grupo e sua relevância para o domínio aplicado.

2 Solução Proposta

As etapas de desenvolvimento do software devem ser descritas de maneira organizada nas próximas seções deste relatório.

2.1 Estrutura Inicial

Na Estrutura Inicial, são descritas a organização do ambiente de desenvolvimento, a definição da equipe de trabalho com a respectiva divisão de responsabilidades e os primeiros protótipos das telas desenvolvidas a partir dos requisitos funcionais levantados.

2.2 Design e Modelagem

Na seção de Design e Modelagem, é elaborado o Diagrama de Casos de Uso, que apresenta as principais interações entre os usuários e o sistema. Além disso, é desenvolvida a representação conceitual do sistema por meio de diagramas de classes, estruturando seus principais elementos e relacionamentos.

2.2.1 Diagrama de Classes

Coloque seu diagrama na pasta “Diagramas” e substitua pelo nome do seu arquivo.

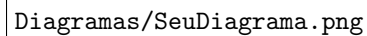
A placeholder for a domain class diagram, represented by a rectangular box with the filename 'Diagramas/SeuDiagrama.png' inside.

Figura 1: Diagrama de Classes de Domínio

Descrição das Classes

Descreva as principais classes do seu software, destacando suas responsabilidades, atributos e métodos. Exemplo:

- **Classe BancoDeDados:** Essa classe é responsável por toda a leitura e escrita de dados em arquivos CSV, sendo responsável por criar, ler, editar e excluir registros (CRUD). Ela organiza arquivos separados para cada tipo de entidade, como Funcionário, Atendente, Cliente, garantindo que todas as informações do sistema fiquem bem armazenadas e acessíveis.

2.3 Implementação da Lógica do Software

A transformação da modelagem em código funcional deve ser abordada, detalhando como as regras de negócio, os mecanismos de persistência e o tratamento de exceções foram implementados. As regras de negócio implementadas devem ser explicadas, mostrando como o software aplica as normas e restrições do domínio. Para os mecanismos de persistência, é necessário detalhar como os dados são armazenados e recuperados, como por meio de arquivos, listas, coleções ou outros recursos de persistência utilizados no projeto. Para o tratamento de exceções, devem ser apresentados exemplos de situações previstas e como o sistema lida com elas, como entradas inválidas, arquivos não encontrados ou operações inválidas, garantindo robustez e confiabilidade.

2.4 Integração e navegabilidade

Deve explicar como a interface gráfica do sistema se conecta à lógica de negócio e aos mecanismos de persistência de dados. Além disso, deve detalhar de que maneira as funcionalidades dos diferentes módulos do software foram integradas, garantindo a interação coerente entre as partes, a navegabilidade entre telas e a usabilidade do sistema.

3 Resultados e Discussão

Nesta seção, deve ser apresentada a análise do software desenvolvido, destacando as funcionalidades implementadas, a integração entre os módulos e a eficácia da solução frente aos objetivos propostos.

3.1 Tecnologia Utilizadas

Deve-se descrever as tecnologias adotadas e suas versões, incluindo a IDE utilizada, as linguagens de programação, frameworks e demais ferramentas que deram suporte à implementação do software.

3.2 Demonstração da Solução

Deve-se apresentar exemplos de telas do software, mostrando a navegação entre elas e a interação com as funcionalidades implementadas. Deve-se incluir a demonstração das principais funcionalidades, como cadastros, consultas e geração de relatórios, evidenciando como os diferentes módulos do software se comunicam e operam de forma integrada para garantir a consistência e a usabilidade do sistema.

Apresente as imagens das telas (coloque as imagens na pasta “Telas”) juntamente com as descrições de cada uma. Divida em subseções para melhor organização. Exemplo:

3.2.1 Modulo Inicial



Figura 2: Tela Inicial

A Tela Inicial é apenas uma apresentação do programa a ser explorado nas telas seguintes.



Figura 3: Tela Login

A Tela Login permite ao usuário acessar sua área de trabalho a partir de um login (CPF) e uma senha. A existência desses dados é verificada no Banco de Dados para liberar o acesso.

3.2.2 Modulo Administrador



Figura 4: Tela Administrador

Essa tela é responsável pela conexão das principais funções do usuário Administrador, sendo: o cadastro e a pesquisa de funcionários, e o cadastro, edição, pesquisa e deleção de documentos.

3.3 Técnicas de programação Aplicadas

Deve-se discutir e apresentar exemplos de partes do código comentadas, destacando onde e como foram aplicadas as principais técnicas de programação, como encapsulamento, abstração, herança, polimorfismo, interface, modularização, tratamento de exceções e boas práticas de design. É importante indicar em quais arquivos ou módulos o código está localizado, permitindo compreender a estrutura do sistema e a aplicação prática dessas técnicas.

3.4 Desafios Encontrados e Soluções Adotadas

Descrever os principais desafios enfrentados durante o desenvolvimento do software, bem como as soluções adotadas para superá-los, podendo incluir técnicas aplicadas fora do contexto do trabalho. Além disso, é importante mencionar técnicas ou funcionalidades planejadas que não foram implementadas, explicando os motivos e os possíveis impactos no uso do software.

3.5 Análise Crítica da Qualidade da Solução

Apresentar os pontos fortes e limitações da solução proposta, como confiabilidade, usabilidade e modularidade.

4 Conclusão e Evolução Futura

Apresentar uma síntese dos resultados obtidos com a solução desenvolvida, destacando os principais benefícios alcançados. Além disso, indicar possíveis evoluções futuras, incluindo a adição de novos módulos, melhorias na manutenção, otimizações ou aprimoramentos de funcionalidades, mostrando como o software pode continuar a se desenvolver e atender de forma ainda mais eficaz às necessidades do domínio aplicado.

Referências

- [1] Paul Deitel and Harvey Deitel. *Java: Como Programar*. Pearson, 1993.
- [2] Geeks for Geeks. Introduction to java swing. <https://www.geeksforgeeks.org/java-swing/>. Acessado em jun. 2025.
- [3] Gustavo Guanabara. Curso de java para iniciantes. https://www.youtube.com/playlist?list=PLHz_AreHm4dkI2ZdjTwZA4mPMxWTfNSpR, 2015. Curso em vídeo.