

**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE  
SÃO PAULO**

**CRISTIANO DE ALMEIDA TOMAZ**

**DISCIPLINA: PROGRAMAÇÃO ORIENTADA A OBJETOS**

**PROFESSOR: PAULO GIOVANI DE FARIA ZEFERINO**

**CURSO: ANÁLISE E DESENVOLVIMENTO DE SISTEMAS**

**SISTEMA LISTA DE AFAZERES IFSP**

**CAMPOS DO JORDÃO**

**2024**

## RESUMO

Este relatório apresenta o desenvolvimento de um sistema desktop intitulado Lista de Afazeres – IFSP, elaborado como parte da disciplina de Orientação a Objetos no curso de Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia de São Paulo – IFSP. O principal objetivo do sistema é auxiliar os alunos da instituição na organização de suas tarefas acadêmicas, permitindo o cadastro, visualização, marcação de conclusão e exclusão de atividades. A aplicação foi desenvolvida utilizando a linguagem C++ com o framework Qt, adotando os princípios da Programação Orientada a Objetos, como encapsulamento, modularização e reaproveitamento de código. Com interface intuitiva e funcionalidades práticas, o sistema visa promover maior controle e produtividade no cotidiano acadêmico dos usuários. O projeto reforça a aplicação prática dos conceitos aprendidos na disciplina, unindo teoria e prática no desenvolvimento de soluções úteis à comunidade estudantil.

**Palavras-chave:** Organização acadêmica. C++. Qt. Orientação a objetos. Sistema desktop.

## ABSTRACT

This report presents the development of a desktop system entitled Lista de Afazeres – IFSP, created as part of the Object-Oriented Programming course in the Analysis and Systems Development program at the Federal Institute of Education, Science and Technology of São Paulo – IFSP. The main objective of the system is to assist students in organizing their academic tasks by enabling the registration, viewing, completion marking, and deletion of activities. The application was developed using the C++ programming language and the Qt framework, following the principles of object-oriented programming such as encapsulation, modularization, and code reuse. With an intuitive interface and practical features, the system aims to promote better control and productivity in the academic routine of its users. The project reinforces the practical application of concepts learned during the course by integrating theory and real-world problem-solving.

**Keywords:** Academic organization. C++. Qt. Object-oriented programming. Desktop application.

## SUMÁRIO

<b>1</b>	<b>Introdução</b> _____	<b>5</b>
<b>1.1</b>	<b>Objetivos</b> _____	<b>6</b>
<b>1.2</b>	<b>Justificativa</b> _____	<b>6</b>
<b>1.3</b>	<b>Aspectos Metodológicos</b> _____	<b>7</b>
<b>1.4</b>	<b>Aporte Teórico</b> _____	<b>8</b>
<b>2</b>	<b>Metodologia</b> _____	<b>9</b>
<b>3</b>	<b>Resultados Obtidos</b> _____	<b>13</b>
<b>4</b>	<b>Conclusão</b> _____	<b>16</b>
<b>5</b>	<b>Referências Bibliográficas</b> _____	<b>17</b>

## 1 INTRODUÇÃO

A organização de tarefas acadêmicas é um desafio comum enfrentado por estudantes do ensino superior, especialmente diante de uma rotina repleta de atividades, prazos e responsabilidades. Em instituições como o Instituto Federal de Educação, Ciência e Tecnologia de São Paulo – IFSP, onde a carga horária e o nível de exigência demandam disciplina e planejamento, o uso de ferramentas tecnológicas pode contribuir significativamente para o gerenciamento eficaz do tempo e das obrigações acadêmicas.

Neste contexto, foi desenvolvido o sistema Lista de Afazeres – IFSP, uma aplicação desktop voltada aos alunos do IFSP com o objetivo de auxiliar na organização das tarefas diárias, como trabalhos, provas e outras atividades escolares. O sistema permite o cadastro de novas tarefas, visualização em tabela, marcação de conclusão e exclusão de registros, oferecendo ao usuário uma interface simples e funcional que visa tornar o controle de tarefas mais acessível e intuitivo.

O projeto foi desenvolvido no âmbito da disciplina de Orientação a Objetos, do curso de Análise e Desenvolvimento de Sistemas, e tem como finalidade aplicar, na prática, os fundamentos da programação orientada a objetos, como encapsulamento, abstração, herança e modularidade. A linguagem utilizada foi o C++, em conjunto com o framework Qt, amplamente adotado no desenvolvimento de interfaces gráficas multiplataforma.

Este relatório tem como objetivo apresentar o processo de desenvolvimento do sistema, descrever sua estrutura, funcionalidades e arquitetura, além de analisar os resultados obtidos e refletir sobre a contribuição do projeto para o aprendizado dos conceitos de orientação a objetos.

## 1.1 Objetivo

Desenvolver uma aplicação desktop utilizando os conceitos de programação orientada a objetos para auxiliar alunos do Instituto Federal de Educação, Ciência e Tecnologia de São Paulo – IFSP no gerenciamento de suas tarefas acadêmicas, promovendo organização, produtividade e controle sobre prazos.

### Objetivos específicos

- Aplicar os princípios da orientação a objetos (encapsulamento, abstração, modularidade) em um sistema real;
- Criar uma interface gráfica amigável e funcional, utilizando o framework Qt;
- Permitir ao usuário cadastrar, visualizar, editar e excluir tarefas acadêmicas;
- Implementar mecanismos de persistência de dados por meio de arquivos;
- Estimular a autonomia e a responsabilidade dos alunos com suas atividades escolares;
- Facilitar a aprendizagem prática dos conteúdos abordados na disciplina.

## 1.2 Justificativa

Durante a trajetória acadêmica, é comum que os estudantes se deparem com a necessidade de conciliar múltiplas disciplinas, prazos e atividades, o que exige um alto grau de organização. No contexto específico do Instituto Federal de Educação, Ciência e Tecnologia de São Paulo (IFSP), observamos que os professores adotam métodos distintos para comunicar a entrega de trabalhos, a realização de provas e demais tarefas avaliativas. Essa diversidade de abordagens pode dificultar a gestão do tempo por parte dos alunos, comprometendo seu rendimento e aumentando o risco de atrasos ou esquecimentos.

Diante dessa realidade, surgiu a proposta de desenvolver um sistema de controle de afazeres voltado aos estudantes do IFSP. O objetivo principal é centralizar em um único ambiente todas as informações relevantes sobre tarefas acadêmicas, facilitando a visualização, o planejamento e a execução das atividades.

Com essa centralização, os estudantes poderão organizar seu cronograma de forma mais clara e objetiva, otimizando sua produtividade e reduzindo o estresse associado à sobrecarga de compromissos.

Além de atender a uma demanda prática e cotidiana dos alunos, este projeto também se justifica como uma excelente oportunidade para aplicar, na prática, os conceitos estudados na disciplina de Orientação a Objetos. Ao projetar e desenvolver um sistema real, os estudantes têm a chance de aprimorar suas habilidades técnicas, compreender melhor os desafios do desenvolvimento de software e experimentar o ciclo completo de construção de uma aplicação, do levantamento de requisitos à implementação funcional.

### **1.3 Aspectos Metodológicos**

Este projeto foi desenvolvido com base na metodologia de prototipação evolutiva, que permite o aprimoramento contínuo do sistema a partir de versões incrementais. A escolha por essa abordagem se deu pela possibilidade de testar, validar e ajustar funcionalidades à medida que o sistema era construído, favorecendo o aprendizado prático e a consolidação dos conceitos de programação orientada a objetos.

A aplicação foi desenvolvida na linguagem C++, utilizando o Qt Creator como ambiente de desenvolvimento integrado (IDE). A adoção do Qt atendeu a um requisito estabelecido pelo professor e mostrou-se apropriada, uma vez que a ferramenta oferece ampla documentação, suporte robusto à criação de interfaces gráficas intuitivas e aderência aos princípios da programação orientada a objetos.

Foram seguidas as seguintes etapas para o desenvolvimento do sistema:

1. Levantamento de Requisitos: Identificação das necessidades dos usuários-alvo (alunos do IFSP), com foco em funcionalidades básicas como cadastro de tarefas, definição de tipo (prova ou trabalho), data de entrega, marcação de conclusão e exclusão de tarefas.

2. Modelagem: Definição da estrutura das classes e interação entre os objetos, com base nos princípios da orientação a objetos, como encapsulamento, abstração e responsabilidade única.
3. Desenvolvimento da Interface Gráfica (GUI): Construção da janela principal e do formulário de nova tarefa, utilizando recursos visuais do Qt para proporcionar uma experiência de uso simples e funcional.
4. Implementação da Lógica de Negócio: Desenvolvimento das funcionalidades essenciais, como salvar e carregar tarefas, além do tratamento de eventos, manipulação de arquivos e persistência em formato .csv.
5. Testes Funcionais: Validação das funcionalidades implementadas por meio da execução do sistema em diferentes cenários, garantindo a integridade e o correto funcionamento do fluxo proposto.
6. Documentação: Elaboração deste relatório técnico, detalhando todas as etapas do processo de desenvolvimento, desde a justificativa até os resultados obtidos.

## 1.4 Aporte Teórico

A construção de sistemas computacionais baseados na programação orientada a objetos (POO) tem se consolidado como uma abordagem eficaz no desenvolvimento de aplicações modulares, reutilizáveis e de fácil manutenção. A POO se baseia em conceitos fundamentais como classe, objeto, encapsulamento, herança e polimorfismo, os quais promovem uma estrutura de código mais organizada e aderente ao mundo real (DEITEL; DEITEL, 2016).

A utilização de interfaces gráficas (GUI - Graphical User Interface) desempenha papel essencial na experiência do usuário, facilitando a interação com o sistema de forma visual e intuitiva. Ferramentas como o Qt, amplamente reconhecidas na indústria de software, fornecem uma estrutura robusta para desenvolvimento de GUIs em C++, com suporte a recursos como layouts automáticos, sinais e slots para eventos, e integração com arquivos .ui (KLEE; BERTON, 2020).

O ambiente de desenvolvimento Qt Creator permite a aplicação prática desses



conceitos em um cenário educacional. Além disso, sistemas voltados à organização pessoal, como listas de afazeres e gerenciadores de tarefas, têm sido amplamente estudados em pesquisas sobre produtividade e métodos de estudo (FERREIRA; SANTOS, 2021), demonstrando que a gestão eficiente do tempo pode impactar positivamente o desempenho acadêmico dos alunos.

A escolha da linguagem C++ e da biblioteca Qt, portanto, se justifica tanto pelo alinhamento com os objetivos pedagógicos da disciplina quanto pela aderência a boas práticas de desenvolvimento de software.

## **2 Metodologia**

O desenvolvimento do sistema seguiu uma abordagem incremental, estruturada em etapas que compreendem: análise do problema, definição dos requisitos, modelagem orientada a objetos, implementação da interface gráfica e funcionalidades, testes e validação. O processo teve como base os princípios da Engenharia de Software e os conceitos fundamentais da Programação Orientada a Objetos (POO), conforme estabelecido na ementa da disciplina.

A linguagem de programação utilizada foi o C++, em conjunto com o framework Qt, por se tratar de uma exigência do componente curricular e por oferecer suporte completo ao paradigma orientado a objetos e ao desenvolvimento de interfaces gráficas (GUI). A IDE escolhida foi o Qt Creator, devido à sua integração nativa com o framework e por proporcionar ferramentas visuais para a construção das janelas do sistema.

A modelagem do sistema foi realizada com base no diagrama de classes e nos princípios de coesão e baixo acoplamento. As entidades do sistema foram representadas por meio de classes específicas, com métodos encapsulados e atributos privados, seguindo as boas práticas de design.

Para armazenamento temporário das tarefas, utilizou-se a estrutura de uma tabela em memória (QTableWidget) e, para persistência local, a gravação dos dados em arquivo do tipo .csv. Essa estratégia possibilita que os dados sejam recuperados

mesmo após o encerramento da aplicação, garantindo continuidade no uso do sistema.

Os testes foram conduzidos manualmente durante o processo de desenvolvimento, com foco na validação das funcionalidades principais: adição, exclusão, marcação de tarefas como concluídas e persistência de dados. A usabilidade também foi considerada por meio da escolha de ícones intuitivos, mensagens de confirmação e disposição adequada dos elementos na interface.

### **Ferramentas Utilizadas**

Para o desenvolvimento da aplicação proposta neste projeto, foram empregadas ferramentas que oferecem suporte adequado ao paradigma de programação orientada a objetos, à construção de interfaces gráficas e à manutenção do código-fonte. As principais ferramentas utilizadas foram:

- **Linguagem C++**

A linguagem C++ foi escolhida por ser amplamente utilizada no desenvolvimento de sistemas com foco em desempenho e estruturação orientada a objetos. Possui suporte a encapsulamento, herança, polimorfismo e abstração, conceitos fundamentais para o projeto.

- **Qt Framework**

O Qt é um framework multiplataforma voltado para o desenvolvimento de aplicações com interface gráfica. Ele fornece bibliotecas que facilitam a criação de janelas, botões, tabelas e diversos componentes visuais, além de integrar bem com os recursos da linguagem C++.

- **Qt Creator**

Trata-se da IDE oficial do Qt, utilizada para escrever, compilar e depurar o código-fonte. O Qt Creator também oferece um editor visual de interfaces (.ui), facilitando a construção e personalização da interface do usuário. Sua integração com o sistema de build (qmake) e com o gerenciador de recursos foi essencial para a organização do projeto.

- **QMake**

Ferramenta de compilação utilizada para configurar e gerar os arquivos de

build do projeto. Através do arquivo .pro, foi possível listar as dependências e estruturar o projeto de forma modular.

- Arquivo CSV (Comma-Separated Values)

Para armazenar localmente os dados inseridos na aplicação, foi utilizado um arquivo .csv. Esse formato foi escolhido por ser simples, leve e facilmente manipulável com ferramentas externas como editores de texto e planilhas eletrônicas.

Essas ferramentas, utilizadas em conjunto, proporcionaram um ambiente de desenvolvimento eficiente e aderente às necessidades do projeto, além de facilitarem a organização e reutilização do código, conforme os princípios da programação orientada a objetos.

### **Construção da Interface Principal**

A interface principal da aplicação foi desenvolvida utilizando o Qt Designer, ferramenta visual integrada ao Qt Creator. Através dela, foi possível organizar os elementos gráficos de maneira intuitiva e alinhada com os princípios da usabilidade.

A Figura 1 apresenta a tela principal (mainwindow.ui) em modo de edição, onde é possível observar a disposição dos seguintes elementos:

- Um QLabel com a logomarca do Instituto Federal, representando a identidade visual da aplicação;
- Um QWidget com quatro colunas: “Nome”, “Tipo”, “Data” e “Status”, utilizadas para listar e acompanhar as tarefas cadastradas;
- Dois QPushButton, nomeados como “Nova Tarefa” e “Excluir Tarefa”, responsáveis por interações diretas do usuário com a tabela de afazeres.

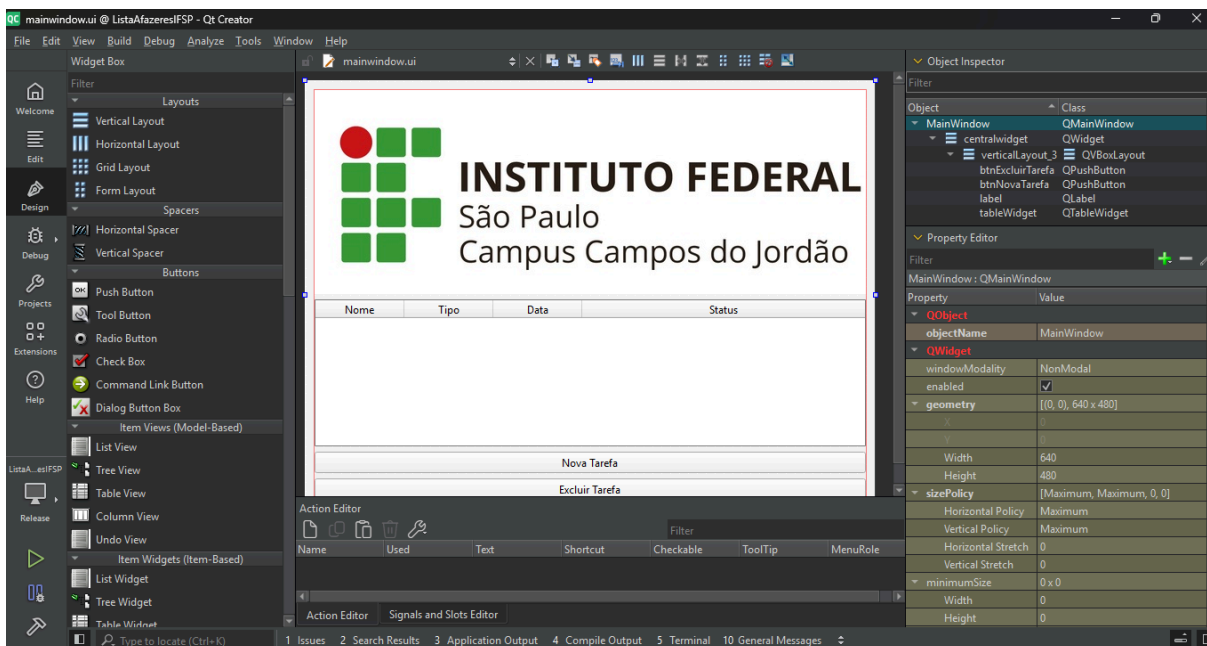


Figura 1 – Tela principal da aplicação no modo de design

A utilização de layouts verticais (QVBoxLayout) permitiu que os elementos se ajustassem dinamicamente ao redimensionamento da janela, oferecendo flexibilidade e melhor aproveitamento do espaço disponível.

### Construção da Janela de Cadastro de Tarefas

A tela de cadastro de tarefas foi construída utilizando o recurso de diálogo modal fornecido pelo Qt, mais especificamente através de um QDialog com layout em formulário. Essa interface é ativada ao clicar no botão "Nova Tarefa" da tela principal.

A Figura 2 mostra a janela NovaTarefa, desenvolvida com o auxílio do editor gráfico do Qt Designer. Ela é composta pelos seguintes elementos:

- QLineEdit para entrada do nome da tarefa;
- QComboBox para seleção do tipo da tarefa, com valores pré-definidos como "Prova" e "Trabalho";
- QDateTimeEdit que permite ao usuário selecionar a data de entrega;
- QDialogButtonBox com os botões "OK" e "Cancelar" para confirmar ou

cancelar o cadastro.

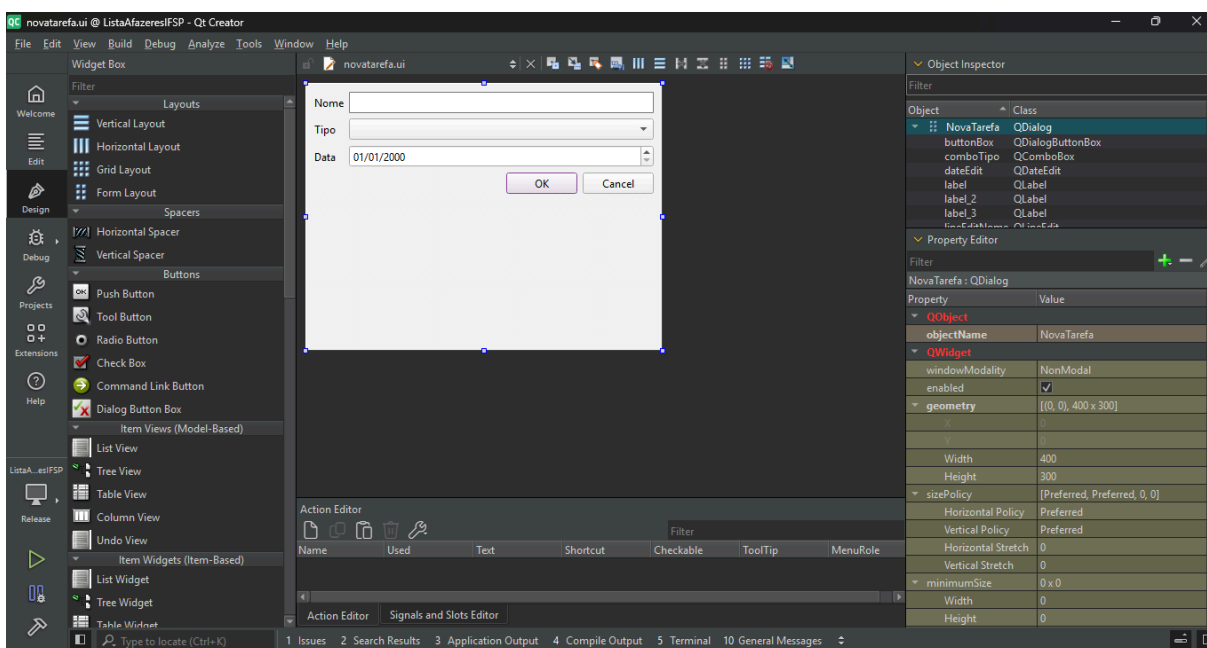


Figura 2 – Tela de cadastro de nova tarefa

A escolha por uma janela modal garante que o usuário complete ou descarte a operação antes de retornar à tela principal, promovendo maior controle e consistência nos dados inseridos. O uso do `QFormLayout` proporciona organização clara e alinhada dos campos, favorecendo a experiência do usuário.

### 3. Resultados Obtidos

A partir do desenvolvimento da aplicação “Lista de Afazeres – IFSP”, foi possível alcançar os objetivos propostos, entregando um sistema funcional, leve e intuitivo, voltado especificamente para os alunos do Instituto Federal de Educação, Ciência e Tecnologia de São Paulo – Campus Campos do Jordão.

O sistema permite o cadastro e gerenciamento de tarefas acadêmicas de forma prática, organizando os dados em uma tabela com as seguintes colunas: Nome da Tarefa, Tipo, Data e Status. Cada tarefa pode ser marcada como concluída ou pendente, através de um simples clique no ícone representativo da coluna “Status”.

## Interface Principal

A Figura 3 apresenta a interface principal do sistema, onde é possível visualizar a lista de tarefas cadastradas, adicionar novas tarefas ou excluir tarefas selecionadas. A tabela se ajusta automaticamente ao tamanho da janela, permitindo melhor legibilidade.



Figura 3 – Tela principal da aplicação

## Interface de Cadastro

A Figura 4 ilustra a janela modal responsável por cadastrar uma nova tarefa. O usuário informa o nome da tarefa, escolhe o tipo (entre “Prova” e “Trabalho”) e

seleciona a data de entrega. O botão “OK” grava as informações na tabela principal.

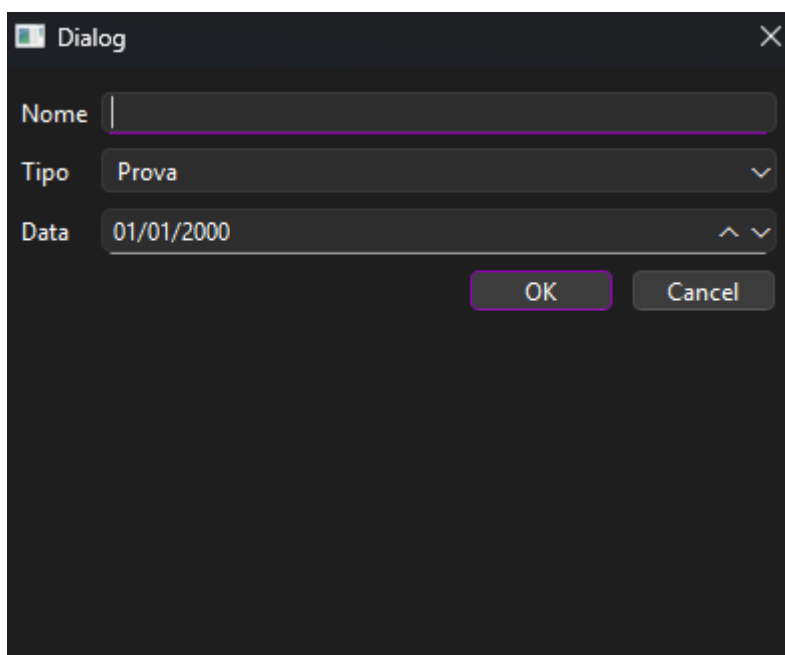
A screenshot of a 'Dialog' window with a dark background. The window has a title bar with a close button (X). Inside, there are three input fields: 'Nome' (empty), 'Tipo' (set to 'Prova'), and 'Data' (set to '01/01/2000'). The 'Data' field has up and down arrow icons. At the bottom right, there are 'OK' and 'Cancel' buttons.

Figura 4 – Janela de cadastro de nova tarefa

### Funcionalidade de Conclusão e Exclusão

Ao clicar no ícone de status (✗), o sistema alterna para (✓), indicando que a tarefa foi concluída. Da mesma forma, ao selecionar uma linha e clicar no botão “Excluir Tarefa”, o sistema solicita confirmação antes de remover a tarefa da lista, o que evita exclusões acidentais.

### Salvamento e Persistência

Outro resultado importante foi a implementação da persistência de dados. As tarefas são automaticamente salvas em um arquivo CSV (tarefas.csv) ao fechar a aplicação, e recarregadas ao abri-la novamente, garantindo que o usuário não perca informações cadastradas.

#### **4. Conclusão**

O desenvolvimento da aplicação Lista de Afazeres – IFSP proporcionou uma oportunidade prática de aplicar os conceitos aprendidos na disciplina de Programação Orientada a Objetos, utilizando a linguagem C++ e o ambiente Qt Creator. A experiência favoreceu a compreensão de aspectos como encapsulamento, modularização, criação de interfaces gráficas, tratamento de eventos e persistência de dados.

O sistema atendeu aos objetivos propostos, oferecendo uma solução funcional para o gerenciamento de tarefas escolares, especialmente útil para os alunos do Instituto Federal que desejam se organizar melhor diante da diversidade de prazos e métodos de avaliação utilizados pelos docentes. A possibilidade de registrar, visualizar, marcar como concluídas e excluir tarefas de maneira simples e intuitiva mostra-se bastante eficiente no contexto educacional.

#### **Sugestões de Melhorias**

Apesar do sucesso da implementação, o sistema pode ser ampliado e melhorado em versões futuras. Algumas sugestões incluem:

- Filtro por tipo de tarefa ou por data: permitir que o usuário visualize apenas tarefas do tipo "Prova", "Trabalho", ou tarefas previstas para uma semana específica.
- Notificações de tarefas próximas do vencimento: adicionar alertas visuais ou sonoros para lembrar o aluno de prazos iminentes.
- Integração com calendário externo (Google Calendar ou Outlook): sincronizar as tarefas com ferramentas já utilizadas pelos estudantes.
- Interface responsiva e suporte a temas: melhorar a estética da aplicação, adaptando a interface a diferentes tamanhos de tela e adicionando modos escuro/claro.
- Versão multiplataforma ou mobile: desenvolver a aplicação para Android ou Web, facilitando o uso em dispositivos móveis.



- Validação dos campos: impedir que o usuário salve uma tarefa com nome vazio ou data inválida.

Essas melhorias representam um caminho natural de evolução para o sistema, demonstrando que o projeto, além de cumprir sua finalidade didática, possui potencial de aplicação real e escalabilidade.

## **5. Referências Bibliográficas**

DEITEL, Paul; DEITEL, Harvey. C++: Como programar. 10. ed. São Paulo: Pearson Education, 2016.

QT. Qt Documentation. Qt Group. Disponível em: <https://doc.qt.io>. Acesso em: 15 jun. 2025.

FARIA, Gustavo. Qt Creator - Primeiros passos. YouTube, jan. 2021. Disponível em: [https://www.youtube.com/watch?v=n71zD8QZXmY&list=PLx4x\\_zx8csUhzAyii9-cY-IJwo00p\\_5AC&index=2](https://www.youtube.com/watch?v=n71zD8QZXmY&list=PLx4x_zx8csUhzAyii9-cY-IJwo00p_5AC&index=2). Acesso em: 15 jun. 2025.