Relatório de Programação Orientada a Objetos (POO)

Aluno:

Pedro Faria (a23290)

Professor

Ernesto Casanova

Índice

Introdução

O presente relatório tem como objetivo descrever o trabalho desenvolvido com implementação em C#. Este projeto visa aplicar os diversos conceitos adquiridos na UC de Programação Orientada a Objetos (POO), tendo como objetivo consolidar as ideias de abstração, herança, polimorfismo e encapsulamento no contexto de um tema escolhido pelo aluno. Nesta situação, o sistema a desenvolver tem como objetivo gerir o acompanhamento de obras na área da construção civil de forma simples e organizada, incluindo funcionários, arquivo de projetos e documentos, gestão de stock, e entre outros.

Objetivos do Projeto

Os objetivos definidos para o projeto desenvolvido foram os seguintes:

1. Desenvolver e implementar uma solução orientada a objetos de modo a possibilitar uma gestão facilitada do processo de vários projetos na área da construção civil.
2. Aplicar os conceitos aprendidos em POO para aplicar funcionalidades que facilitem o funcionamento do sistema
3. Auxiliar na organização das operações dos projetos com o sistema desenvolvido, através de registo e gestão de documentos, funcionários, stock, etc.

Diagrama da Estrutura de Classes

O sistema desenvolvido contém 4 classes que funcionam como componentes principais de um projeto na área da construção civil: **User**, **Document**, **Project** e **Stock**.

Cada classe representa um sistema específico no processo de um projeto de uma obra. A estrutura dessas classes esta representada na figura 1 abaixo.

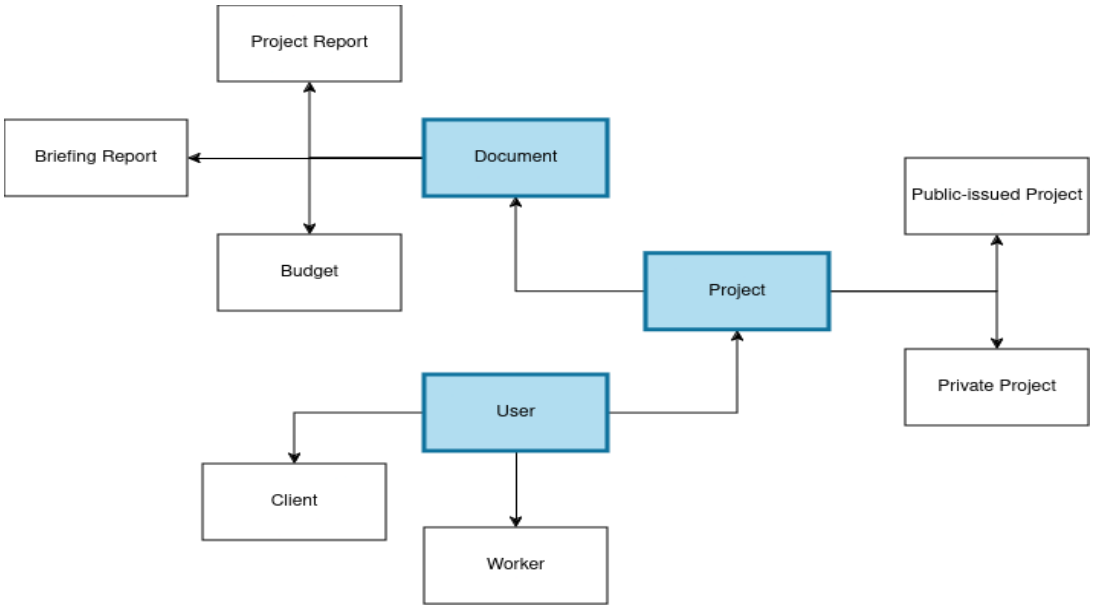

Figura 1 - Estrutura de Classes

1. **User**

   A classe User contem todos os utilizadores que participam em qualquer fase do processo de uma obra na área da Construção Civil, como os clientes e funcionários.

   A superclasse define as propriedades e métodos que são comuns a todos os tipos de utilizadores, tais como ID, nome, contacto, endereço, e entre outras.

   Esta classe esta dividida em duas subclasses, para os dois tipos de utilizadores: **Client** e **Worker**. Ambas tem propriedades únicas, como por exemplo um cargo para a subclasse Worker.

2. **Project**

   A classe Project define propriedades comuns a todos os tipos de obras. Cada obra pode ser de caracter publico ou privado, tais estando definidas como subclasses. Cada subclasse terá propriedades especificas, relativamente ao cliente associado (quer publico ou privado).

3. **Document**

A classe Document representa todos os documentos que estão de qualquer maneira associados a uma obra. Estes documentos estão divididos em subclasses para cada tipo de documento:

- **Briefing Report:** Referente para qualquer tipo de relatório feito para uma obra;
- **Budget:** Documentos relacionados a orçamentos;
- **Meeting Summary:** Documentos referentes a resumos de reuniões realizadas acerca de uma certa obra ou projeto;
- **Other:** Outros documentos.

Implementações Utilizadas

O desenvolvimento dos sistemas e das classes que este utiliza tem o objetivo de manterem-se a par com os conceitos da UC de POO, tais como:

- **Herança e utilização de subclasses**
  - A criação de heranças estabelece uma hierarquia organizada de variáveis e possibilita a reutilização de código.
- **Construtores**
  - A utilização de construtores permite a inicialização de classes de modo que nenhum dado seja perdido na execução do código.
- **Abstração**
  - O uso da abstração permite a implementação de funções que são exclusivas a certas classes, evitando o acesso indevido a essas funções.
- **Encapsulamento**
  - A utilização de encapsulamento na criação de propriedades privadas e publicas permite separar os dados dependendo da importância dos mesmos, garantindo a integridade do programa desenvolvido.

Considerações para o futuro

O desenvolvimento do sistema será continuado nos próximos tempos, e para promover o bom funcionamento do sistema e melhorar as suas funcionalidades, seguem abaixo algumas das possíveis tarefas a explorar para as próximas fases deste projeto:

- Correção de bugs e/ou código incorreto desenvolvido na primeira fase
    - Prevê-se rever toda a estrutura e código desenvolvido com o objetivo de resolver qualquer tipo de erro encontrado.
- Implementação gráfica através do Avalonia ou de outra framework
    - Devido ao desenvolvimento ser realizado em Linux, WinForms e WPF não são soluções ideais para este cenário. Portanto, procura-se aprender e desenvolver uma solução gráfica para este sistema.
- Documentação de código
    - Documentação completa de funções e de propriedades que necessitem de explicações adicionais. Procurar algum sistema similar ao Doxygen para a construção de páginas de documentação a partir dos comentários realizados no código.
- Melhoria do relatório
    - Rever o relatório ao longo do desenvolvimento, adicionar explicações para certas funcionalidades do sistema e corrigir possíveis erros caso sejam descobertos.

Os objetivos estabelecidos acima têm como objetivo melhorar o sistema de modo que se torne em algo mais completo, na ambição de que seja possível ser utilizado num ambiente real do dia-a-dia na área de construção civil. Com o desenvolvimento de uma versão gráfica do sistema através do Avalonia e com uma documentação mais clara, este sistema poderá ser convertido e utilizado facilmente num cenário real.

Conclusão

Neste projeto, desenvolveu-se um sistema de gestão de projetos na área de construção civil, com a implementação feita em C#, seguindo os conceitos adquiridos na UC de Programação Orientada a Objetos (POO), nomeadamente a utilização de heranças, encapsulamento, abstração e polimorfismo.

Com as próximas etapas e com os pontos a averiguar no futuro do desenvolvimento do programa, este sistema visa alcançar o objetivo de este ser usufruído num cenário real por uma empresa de gestão de construção civil, caso o cumprimento das etapas seja concluído com sucesso.