**Rodrigo Marnon Veloso Barros de Morais**

**TÍTULO: Sistema de gerenciamento escolar.**

Divinópolis - MG

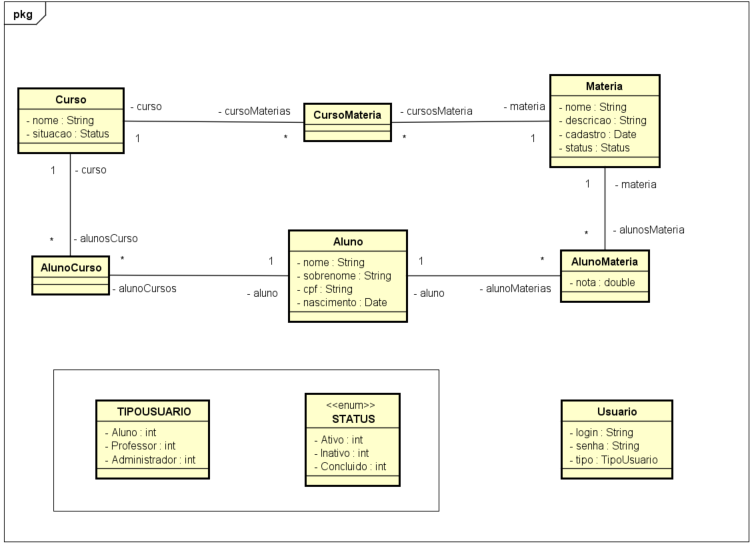
12/2022

**1.Introdução**

A não utilização de métodos de engenharia de software tem gerado maiores custos e sistemas menos confiáveis, e, para seguir as melhores práticas e metodologias disponíveis no mercado, será criada uma implementação asp. Net core na linguagem C#, e banco de dados relacional SQL Server, cujo objetivo é ser um sistema que faz controle e gestão de escolas, nas quais será possível cadastrar desde cursos, alunos e matérias, vincular nota e fazer login no sistema. A aplicação também terá testes unitários para garantir sua qualidade e bom funcionamento. O motivo pelo qual decidi fazer esse projeto CRUD, é entender melhor as tecnologias asp .Net, como trabalhar com sistema de camadas, banco de dados relacional e testes unitários, e espero que ele possa contribuir com meu desenvolvimento na área de tecnologia.

O sistema será desenvolvido seguindo a orientação de diagramas, e sendo implementado e visionado, para melhor acompanhamento da evolução do projeto, e seguirá um fortemente orientado a testes, que é um dos pilares no desenvolvimento da aplicação C#.

* **OBJETIVO Geral -** Este estudo de caso tem como objetivo mostrar na prática como o modelo pode ser implementado sobre o paradigma orientado a objetos, utilizando de um CRUD de sistema de gerenciamento escolar, cujo objetivo é cadastrar alunos, curso, matérias, fazer login e lançar notas no sistema de forma simples e objetiva, com tratamento de erros e validação de entrada de dados.
* **OBJETIVOS ESPECÍFICOS -** Nessa implementação em linguagem orientada a objetos, será abordado:
* Coleta de requisitos
* Leitura e entendimento do diagrama de classes
* Associações
* Conceito dependente
* Classe de associação
* Herança
* Enumerações
* Testes unitários
* Validação de entrada de dados
* Arquitetura baseada em camadas
* Banco de dados relacional



* **JUSTIFICATIVA** - Esta aplicação tem o objetivo de explorar e aprofundar os conhecimentos sobre desenvolvimento de software, abordando as melhores práticas utilizadas no mercado, para que se possa criar uma aplicação moderna, eficiente e segura. A importância do desenvolvimento desse CRUD, passa por me qualificar em uma linguagem que tenho pouco conhecimento, e ao mesmo tempo, me dar parâmetros de arquitetura em camadas e também na introdução de testes unitários, o que garante a qualidade do software.

**2. ESPECIFICAÇÕES INICIAIS DO SOFTWARE**

O software em questão, terá como requisitos funcionais, um backend em c#, capaz de cadastrar cursos, aluno, matérias e usuários, partindo do princípio que haverá regras de validação para cada entidade, com suas respectivas regras e com testes unitários bem definidos.

**2.1 Escopo do Produto**

O produto será um sistema web, porém em um primeiro momento só será feito o backend, com arquitetura em camadas MVC, contendo testes unitários e conexão com um banco de dados relacional SQL Server.

Em um segundo momento pretendo desenvolver a parte frontend em Angular, utilizando o conceito de SPA, caso haja tempo hábil para entrega dele.

**2.2 Público-Alvo**

Esta aplicação se destina a funcionários de escolas e cursos em todo país, podendo ser utilizado tanto por diretores e professores, além da parte administrativa dessas instituições.

**2.3 Definições, Acrônimos e Abreviações**

|  |  |  |
| --- | --- | --- |
| **Sigla** | **Significado** | **Definição** |
| SPA | Single Page Application. | **Single Page Applications (SPA) são aplicações cuja funcionalidade está concentrada em uma única página**. Ao invés de recarregar toda a página ou redirecionar o usuário para uma página nova, apenas o conteúdo principal é atualizado de forma assíncrona, mantendo toda a estrutura da página estática. |
| MVC | Model View Controller | O **MVC** é uma sigla do termo em inglês Model (modelo) View (visão) e Controller (Controle) que facilita a troca de informações entre a interface do usuário aos dados no banco, fazendo com que as respostas sejam mais rápidas e dinâmicas. |

**Quadro 1:** Abreviações e Definições. **Fonte**: autor

**2.7 Funcionalidade do Produto**

O software terá um funcionamento bastante amigável, onde se poderá cadastrar, e validar um curso, não se permitindo duplica – lo no banco de dados, permitirá modificar status desse curso, vincular matérias cadastradas e ativas e, excluir curso, não utilizando soft delete.

Poderá também cadastrar matéria, podendo validar entradas, alterar status da matéria, vincular essa matéria a um curso cadastrado e ativo, e, excluir matéria que não possua vínculos.

Com relação a alunos, poderemos matricular um aluno em um curso válido e ativo, impedir rematrícula, caso já esteja matriculado no curso, inserir nota em matérias cadastradas e ativas, alterar nome e sobrenome respeitando as regras definidas, e, excluir aluno existente no banco de dados.

Usuário será uma pessoa capaz de se logar no sistema, podendo ele cadastrar um usuário, alterar login e senha, respeitando as regras e validações, conectar ao sistema se ativo, e, deletar usuário existente.

**2.9 Ambiente Operacional e Tecnologias**

Será uma aplicação .Net 6.0, desenvolvido em C# 10, utilizando o padrão MVC de arquitetura, e conversando com um banco de dados relacional SQL Server, que será capaz de rodar em qualquer ambiente operacional Linux, Windows e Mac.

Os testes unitários serão escritos utilizando o Microsoft Test Tools, e utilizando também a biblioteca do Fluent Assertions para facilitar a leitura e entendimento dos testes, além do Fluent Validator, que fará a validação de toda entrada de dados que será captada pelas controllers do sistema.

**3. METODOLOGIA**

Será utilizada metodologia ágil scrun, pois o projeto será dividido em ciclos, dos quais será planejada a criação das prioridades para os itens do sistema, que é o agrupamento de todas as funcionalidades a serem entregues.

Quadro 1 - Atividades para Construção de Projeto de Software

|  |  |
| --- | --- |
| **Atividade** | **Artefatos resultantes** |
| Engenharia de Requisitos | estudo de viabilidade, levantamento de requisitos, planejamento de teste. |
| Elaboração do Projeto de Software | diagrama de classes. |
| Tecnologias a serem utilizadas e os requisitos de implantação | Será utilizado linguagem orientada a objetos na linguagem C#, banco de dados SQL Server. |

Fonte: o autor.

3.1 CRONOGRAMA

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ATIVIDADES** | **Out** | **Out** | **Out** | **Out** | **Nov** | **Nov** | **Nov** | **Nov** | **Dez** | **Dez** | **Dez** | **Dez** |
| Criação da arquitetura e estrutura do back end |  | X | X |  |  |  |  |  |  |  |  |  |
| Criação das classes migration do banco de dados sql |  |  | X | X | X |  |  |  |  |  |  |  |
| Criação das validações classes com retorno de mensagem |  |  |  |  | X | X | X |  |  |  |  |  |
| Criação dos testes unitários e classes de exceções personalizada |  | X | X | X | X | X | X |  |  |  |  |  |
| Criação das controllers, e serviços que conversarão com a camada de repository |  |  |  |  |  |  |  | X | X | X | X |  |
| Fatoração e ajustes | x |  |  |  |  |  |  |  |  | x | x | X |

**4. DESENVOLVIMENTO**

A princípio, o projeto foi alterado porque o escopo do projeto inicial era grande demais para ser entregue em um curto espaço de tempo, então foi definido fazer um projeto menor, mas bem detalhado, com testes e com um bom funcionamento de toda sua estrutura que segue um padrão MVC que é muito utilizado nas principais aplicações no mercado.

Após a definição, deu se início no desenvolvimento da aplicação, baseando se na arquitetura em camadas, onde foram criados uma solução, e projetos separados dentro desta solução, e que fica muito bem-organizado e distribuído internamente.

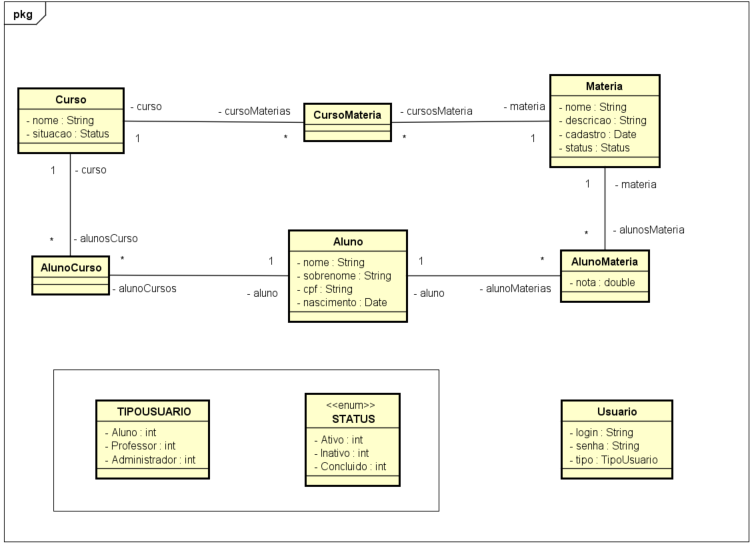
Após criado o primeiro projeto, foi definido que a classes que continha as entidades deveria ficar a cargo do projeto Models, que também é responsável pelas classes tipo enumeradas, onde elas podem ser acessadas, mas não podem acessar outros projetos dentro do sistema.

Em seguida, após mapeado e criados todas as classes, foi se criado o projeto Repositories, onde ele é responsável pela criação das migrations, classes de contexto e strings de conexão com o banco de dados e configurações para o mapeamento do Entity Framework fazer o mapeamento das classes para o banco de dados.

Após essa etapa, foi criado um projeto Validations, que é responsável por dizer as regras a serem seguidas pelas validações e como essas validações devem se comportar, em seguida, foi criado um projeto, Util, que é responsável por ser um utilitário dentro do sistema, que facilita as mensagens de forma genérica, do tipo de retorno que cada controller poderia receber, de forma a ter um maior reaproveitamento de código dentro do sistema.

Após essas etapas, foi se criado o projeto de Tests, onde pude validar todas as situações possíveis de erro, validar o que estava funcionando e o que precisei ajustar, de acordo com as regras levantadas, onde obtive bastante aprendizado, e consequentemente, melhoria na escrita de código.

E por último, foi criado o projeto BusinessRule, onde concentra toda lógica do sistema, que recebe as chamadas da controller, aplica os métodos que são definidos pelas interfaces e faz passa para a camada de repositor salvar as informações no banco de dados, caso elas sejam validadas pelo sistema.



O diagrama de classes apresenta uma relação de muitos para muitos em todas as suas ligações, exceto na classe usuário, que é feita de forma separada, porém, caberia ele ter uma FK na classe Aluno, o que acabou ficando de fora por descuido.

Link do GitHub contendo o projeto:

<https://github.com/rmarnon/Sistema-Escolar>

**5. CONSIDERAÇÕES FINAIS**

Foi uma experiência mito bacana e de muito aprendizado, onde pude pôr em prática um CRUD que em um primeiro momento parece simples, mas conforme vai se aprofundando e vai crescendo, vai se tornando desafiador e muito gratificante.

Pretendo ainda fazer o frontend da aplicação em Angular e aplicar testes utilizando karma/jasmine, para ter um melhor entendimento e fechar com chave de ouro, esses que é o primeiro de muitos softwares que pretendo desenvolver o longo da carreira.

**REFERÊNCIAS BIBLIOGRÁFICAS**

Martin, Robert **Princípios,** **Padrões e Práticas Ágeis em C#**. Bookman, 2011