APLICAÇÃO WEB – MEU BLOCO DE NOTAS

Acadêmicos[1]
Tutor Externo[2]

RESUMO

Este trabalho aborda o desenvolvimento web, destacando a evolução desde a web 1.0 até as atuais perspectivas da Web 3.0 e 4.0. O objetivo principal é analisar e aplicar conceitos fundamentais no projeto prático da aplicação "Meu Bloco de Notas". A metodologia adotada compreende a exploração das linguagens de programação, como C#, ASP.NET Core a utilização do conceito CRUD para operações eficientes, e a importância das tecnologias HTML e CSS na construção das interfaces e o banco de dados SQL Server para manipulação dos dados. A abordagem orientada objetos do C# são discutidos em relação as escolhas tecnológicas. As conclusões revelam uma integração eficaz dessas tecnologias, resultando em uma solução robusta e versátil para o gerenciamento, de notas online, proporcionando uma experiencia intuitiva para os usuários.

Palavra-chave: desenvolvimento web, aplicação web, banco de dados relacional, crud (create, read, update, delete), html, css, sql, diagrama de casos de uso, diagrama de calsses, diagrama de sequência,.

1. INTRODUÇÃO

A criação de aplicações web é um campo em constante evolução, e a capacidade de desenvolver funcionalidades interativas e eficazes é essencial para atender as crescentes demandas dos usuários. De fato, a tecnologia move o mundo, como destacado por Steve Jobs, inventor e fundador da Apple. A magia da tecnologia está presente na capacidade de simplificar tarefas complexas e tornar a interação com o sistema de informação algo aparentemente mágico proporcionando uma experiencia única ao usuário

Uma das principais tarefas no desenvolvimento web é a implementação de um CRUD, que é a sigla para Create (criar), Read (ler), Update (atualizar) e delete (apagar), que se refere as quatro operações básicas que podem ser realizados em um banco de dados ou em uma aplicação web. Essas operações fundamentais permitem que os usuários interajam com os dados armazenados em um sistema, podendo criar registros, ler informações existentes, atualizar

[1]Erick Pereira Kern, Francine dos Santos, Júlio Eduardo de Jesus Paim e Sidnei Martins Casimiro.
[2]José Paulo Feitosa Baía Viana.
Centro Universitário Leonardo da Vinci – UNIASSELVI – Análise e Desenvolvimento de Sistemas (FLC0039ADS) – Prática do Módulo V – 19/11/2023

registros e excluir registros que não são mais necessários. Criar um CRUD significa desenvolver uma interface que permite ao usuário realizar essas operações de forma simples e intuitiva.

Esse trabalho, que tem como título “Meu Bloco de Notas”, consiste em uma aplicação web usando CRUD onde envolve permitir aos usuários adicionar novas notas, ler o conteúdo das notas existentes, fazer atualizações conforme necessário e excluir aquelas notas que não são mais relevantes. A mágica aqui está na capacidade de simplificar a organização ao acesso às anotações diárias tornando o fluxo de trabalho mais eficiente

O projeto foi desenvolvido em linguagens HTML (Linguagem de Marcação de HiperTexto) organiza e formata as páginas web, CSS (Cascading Style Sheets ou Folhas de Estilos em Cascata) aprimora a apresentação do conteúdo, e C# junto com framework ASP .NET CORE é uma linguagem de programação robusta que garante a confiabilidade da aplicação sendo assim usando SQL Server como banco de dados.

O objetivo deste projeto, concluído entre outubro e dezembro de 2023, é proporciona uma solução simples e eficaz para as necessidades de anotações dos usuários, permitindo futuras consultas e tornando a tecnologia uma ferramenta mágica para o dia a dia.

2. EVOLUÇÃO DAS APLICAÇÕES WEB

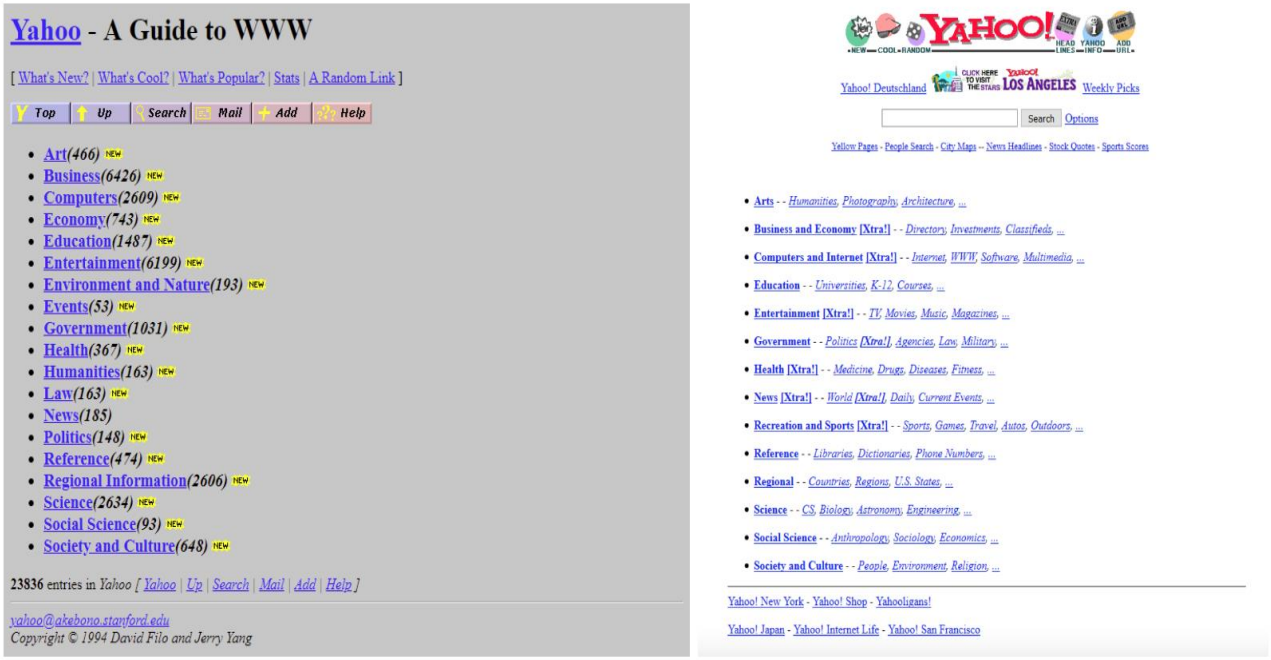
A evolução do desenvolvimento web ao longo dos nãos tem sido marcada por mudanças significativas que transformaram a paisagem online. Anteriormente composta por páginas estáticas, a web passou por uma metamorfose impressionante, tornando-se uma plataforma dinâmica e interativa que desempenha um papel crucial em diversos setores, incluindo o comercio eletrônico, redes sociais e sistemas de gerenciamento de conteúdo.

Para compreender essa transição, é essencial explorar os conceitos fundamentais que permeiam o desenvolvimento de aplicativos inovadores, como o exemplo de “Meu Bloco de Notas”. Nesta seção será aprofundada o conhecimento dos conceitos, oferecendo uma visão abrangente das tecnologias e práticas que impulsionam esse ambiente em constante evolução.

Há apenas uma década, a internet e a tecnologia eram muito diferentes do que são hoje. Naquela época, a tecnologia estava em ascensão, vislumbrando um futuro promissor. Hoje a tecnologia, se tornou uma parte “intima” de nossas vidas, com a internet disponível em nossos dispositivos pessoais, tornando-se uma ferramenta cotidiana. A web agora está literalmente ao alcance de nossas mãos, marcando uma mudança massiva em comparação como uma década atrás.

A história do desenvolvimento web está repleta de marcos que destaca essa progressão. Desde os primórdios dos websites estáticos até os atuais aplicativos web altamente dinâmicos e interativos, várias mudanças tecnológicas e conceituais moldaram essa evolução.

No virado dos anos 80 para os 90, a web 1.0 emergiu com a criação as World Wide Web (WWW) por Tim Berners-Lee, permitindo acesso ao conteúdo multimídia por meio de documentos. Nesse estágio inicial as páginas eram estáticas e simples, predominantemente com textos e poucas imagens, pertencendo principalmente a veículos de imagens e imprensa, a interação com usuário era limitada.

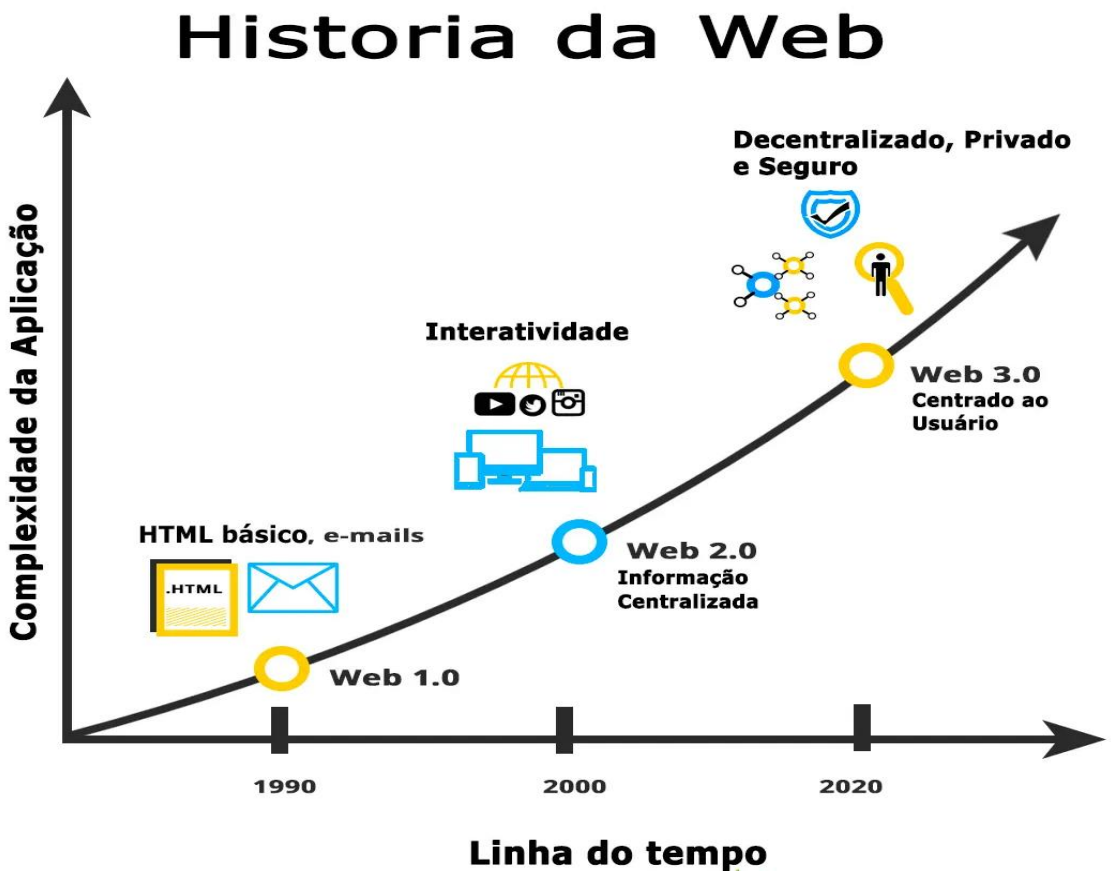


FONTE - Figura 1: imagens do site buscador Yahoo em 1994 a 1996.

Posteriormente, no final dos anos 90, surgiram os motores de busca, como mostrado na Figura 1, como o Yahoo, Google e outros, que revolucionaram a experiencia na internet, permitindo o acesso de sites desconhecidos e a exploração de conteúdo de forma semelhante ao folhear uma revista, essa evolução transformou a maneira de como transformamos a internet.

A popularização da Web 2.0 no início do século XXI revolucionou a forma de como as pessoas interagem na internet, permitindo uma colaboração em larga escala e a criação de conteúdo por parte dos usuários. Plataformas como Wikipedia e redes sociais como o Facebook se destacaram nessa era, com um foco renovado na experiencia do usuário e na usabilidade. À medida que avançamos, a introdução das tecnologias AJAX (Asynchronous JavaScript and XML), possibilitaram a atualização de partes específicas de uma página web sem recarregar a

página inteira, trouxe uma experiencia mais rápida e responsiva para os usuários, marcando o ponto crucial na evolução da internet.



FONTE - Figura 2: Imagem uma visão da evolução da web.

Com a perspectiva da Web 3.0, e Web 4.0 no horizonte, como mostrado na Figura 2, essas transformações continuam a moldar o cenário online. A ascensão das bibliotecas e framworks front-end, como Angular, React e Vue.js, resultou em aplicações web mais complexas, reativas e eficientes. A arquitetura de micro-frontends emergiu como uma abordagem para dividir grandes aplicações web impulsionada pela crescente demanda por mobilidade, como a proliferação de dispositivos móveis e tecnologias responsivas, tornando necessário que as aplicações web se adaptam a diferentes tamanhos de tela e contextos de uso.

A trajetória evolutiva do C# completamente esse panorama dinâmico. Desenvolvido pela Microsoft no final dos anos 1990, o C# emergiu como uma linguagem de programação moderna e robusta, ganhando destaque no ambiente .NET Framework proporcionou um

ecossistema coeso para o desenvolvimento de aplicativos robustos, com suporte a diversas plataformas.

A tecnologia C# desempenhou um papel crucial na transição da Web 1.0 para a experiencia e interatividade da Web 2.0. Sua adição ao conjunto de linguagens do lado do servidor e a interoperabilidade com tecnologias do lado do cliente, como Angular e React, contribuíram para a criação de aplicativos web mais complexos e eficientes. Como apontado por Andrade (2021, p 22), “O C# é reconhecido como uma linguagem robusta, com anos de suporte e amplamente adotada pelo mercado de desenvolvimento de software profissional”, tornando-se uma escolha prevalente no setor.

Ao considerar o cenário mais amplo da evolução web, a interconexão entre linguagens como C# e os avanços em tecnologias front-end destaca a constante busca por maior interatividade e funcionalidade. O papel proeminente do C# na evolução tecnológica, aliado à perspectiva excitante da Web 3.0 e 4.0, sugere que o futuro reserva ainda mais inovações e avanços, continuando a moldar o cenário online com foco na interatividade, eficiência e acessibilidade.

O HTML (Hypertext Markup Language) desempenha um papel fundamental na criação e exibição de conteúdo na web. Sua Jornada começou em 1989 com a visão de Tim Berners-Lee, um cientista da computação britânico, e progrediu ao longo do tempo. A primeira versão, HTML 1.0, introduziu tags básicas em 1991. Em 1995, o HTML 2.0 trouxe suporte a tabelas, formulários, e framesets, acompanhando o aumento da popularidade do navegador Mosaic. O HTML 3.0 de 1997, incorporou folhas de estilos (CSS) e elementos multimidia, coincidindo com o sucesso da Netscape Navigator. O HTML 4.0 lançado em 1999, trouxe recursos avançados, incluindo suporte a Scripts ao lado do cliente JavaScript, e a combinação do XML em documento XHTML. Em 2000, o Html 4.01 corrigiu erros e inconsistência. Em 2014, o HTML5 foi lançado, introduzindo elementos semânticos, suporte a vários áudios, geolocalização e capacidade de armazenamento local. O HTML5 tornou a versão predominante e continua a evoluir para atender as demandas em constante mudança da web.

O CSS assim como o HTML, passou por uma jornada de aprimoramento aos longos dos anos. Inicialmente, o CSS era usado para aplicar os estilos básicos a elementos HTML. Com o tempo evoluiu para uma linguagem de estilização poderosa e flexível, no início, os estilos simples, como fontes e cores, eram aplicados, no entanto, a medida que a web se tornava mais dinâmica, o CSS progrediu para incluir recursos avançados, como o posicionamento de

elementos, animações transições e a capacidade de criar layouts responsivos. Esse desenvolvimento permitiu aos desenvolvedores web criar interfaces atraentes e funcionais.

Em conclusão, a evolução dos aplicativos, desde os primórdios da Web 1.0 até o cenário tecnológico atual, reflete em uma constante busca por maior interatividade e funcionalidade. À medida que a tecnologia avançou, vimos a transição de páginas estáticas para experiencia dinâmica e interativa, impulsionadas por avanços no desenvolvimento web, como a introdução de linguagens de programação no lado do servidor e tecnologias do lado do cliente. Além disso, o desenvolvimento de linguagens como Java e framworks front-end como Angular, React e Vue.js permitiu a criação de aplicativos web mais complexos e eficientes, atendendo as crescentes demandas dos usuários. A evolução do CSS também desempenhou um papel fundamental na criação de interfaces atraentes e responsivas. A tecnologia em C#, em particular, teve um impacto significativo, tornando-se amplamente adotada, de acordo com Andrade (2021, p. 22) “Uma linguagem robusta, com anos de suporte e adotada largamente pelo mercado de desenvolvimento de software profissional”. Sua disponibilização sob a licença de software livre e sua versatilidade contribuíram para sua popularidade. À medida que a evolução da web continua, com a perspectiva da Web 3.0 e 4.0 no horizonte, podemos esperar mais inovações e avanços que moldaram ainda mais o cenário online, impulsionando a interatividade, a eficiência e a acessibilidade.

O futuro promete oportunidades emocionantes para o desenvolvimento de aplicativos que atendam às crescentes expectativas dos usuários em um ambiente digital em constante evolução.

2.1 Conceitos relacionados ao desenvolvimento de aplicações web

No contexto de desenvolvimento web, é fundamental definir o que se entende por “aplicações web”. Conallen (1999) considera que uma aplicação web é um site na web que incorpora lógica de negócio cujo uso pode alterar o estado nesse negócio. Além disso, Paula Filho (2003) define aplicações web como produtos de software ou sistemas de informática que fazem o uso de uma arquitetura distribuída, em parte sob o protocolo HTTP, com parte das interfaces de usuários acessíveis por meio de um navegador (browser). Nesse contexto, é importante destacar que as aplicações web também se baseiam em estruturas de hipertextos e/ou hipermídia. Este projeto envolve o desenvolvimento de uma aplicação web intitulada “Meu Bloco de Notas”, uma aplicação. No contexto específico deste projeto, “Meu Bloco de Notas” representa um exemplo de aplicação web. Vamos destacar esses conceitos:

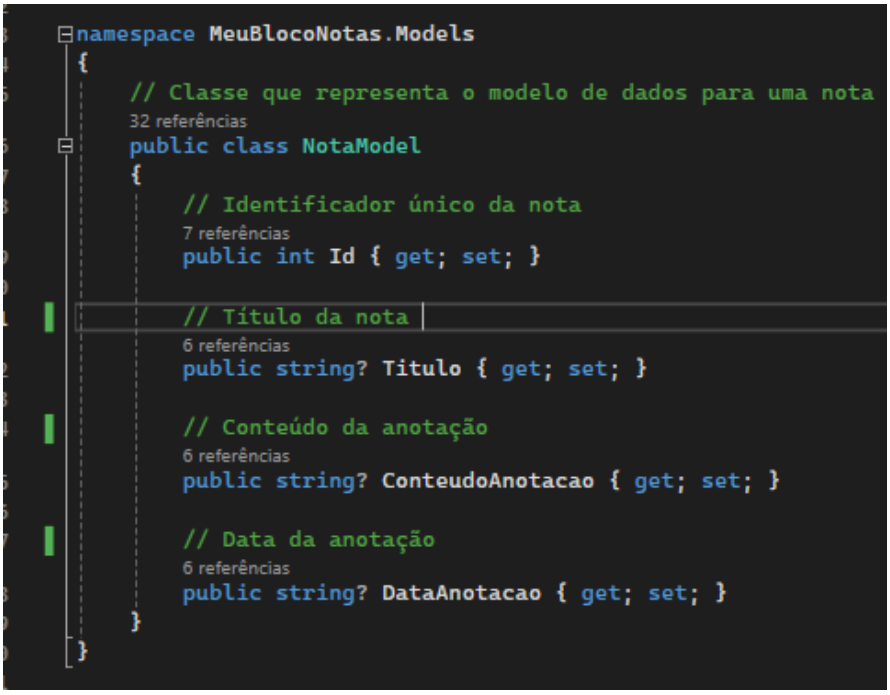
- **CRUD (Create, Read, Update, Delete):** É um conjunto de operações básicas que podem ser realizadas em um banco de dados ou em uma aplicação web. Isso envolve criar (adicionar), ler (visualizar), atualizar e excluir informações. No contexto do projeto, os usuários podem realizar essas operações com suas notas.
- **HTML (Hypertext Markup Language):** É uma linguagem usada para criar a estrutura e a marcação das páginas web. No projeto, o HTML é usado para organizar e formatar o conteúdo das notas.
- **CSS (Cascading Style Sheets):** É utilizado para aprimorar a apresentação visual das páginas web. No projeto, o CSS é responsável por estilizar e estilizar e tornar a interface mais atraente.
- **C#:** É uma Linguagem de programação orientada objetos moderna, inovadora e de código aberto, projetada para ser plataforma cruzada, reconhecida como uma das cinco principais linguagens de programação C# oferece uma abordagem robusta e eficiente para o desenvolvimento de software.
- **ASP .NET CORE:** Um framwork de desenvolvimento web de código aberto e plataforma cruzada, oferece eficiência, modularidade e alto desempenho para a criação de aplicativos web modernas usando a linguagem C#.
- **SQL Server:** É um sistema de gerenciamento de banco de dados relacional que armazena as notas e outras informações do projeto. Ele permite a organização, recuperação e gerenciamento de dados.

Esses conceitos são fundamentais para o desenvolvimento de aplicações web eficazes, e a combinação dessas tecnologias e linguagens permite que “Meu Bloco de Notas” onde que fornece uma solução eficiente para as necessidades de organização e acesso a notas diárias, tornando o fluxo de trabalho mais eficiente.

2.2 Funcionamento e Características do ASP.NET Core MVC

O ASP.NET Core MVC, empregado na aplicação web “Meu Bloco de Notas”, adota a arquitetura Model-View-Controller (MVC) para proporcionar uma estrutura sólida e eficiente. Essa abordagem organiza a aplicação em três componentes principais, oferecendo uma base modular e de fácil manutenção

A camada de Modelo (Model) é responsável pela manipulação dos dados e pela execução da lógica de negócios, aqui ocorre o gerenciamento e armazenamento de informações, garantindo uma integração coesa com a lógica subjacente.



Fonte: Figura 3, Autor

O trecho de código apresenta na Figura 3 é uma classe chamada **NotaModel** que desempenha um papel fundamental na aplicação web "Meu Bloco de Notas". Analisando cada parte do código:

- **Nota Model**: Esta é uma classe que representa o modelo de dados para uma nota na aplicação. Um modelo de dados é uma estrutura que organiza e armazena informações específicas sobre um conceito ou entidade dentro da aplicação. No contexto do "Meu Bloco de Notas", a **Nota Model** define como os dados de uma nota devem ser estruturados e manipulados.
- **Id**: Este é um membro da classe que representa o identificador único de uma nota. Cada nota terá um identificador exclusivo, o que é crucial para distingui-la de outras notas e garantir a integridade dos dados.
- **Título**: Um membro que representa o título da nota. Permite armazenar e recuperar o título associado a cada anotação.

- **Conteúdo Anotação:** Este membro lida com o conteúdo da anotação. Ele armazena o texto ou informações específicas que o usuário deseja registrar em uma determinada nota.
- **Data Anotação**: Representa a data associada à anotação. Este membro permite registrar o momento em que a nota foi criada, possibilitando a organização cronológica das anotações.

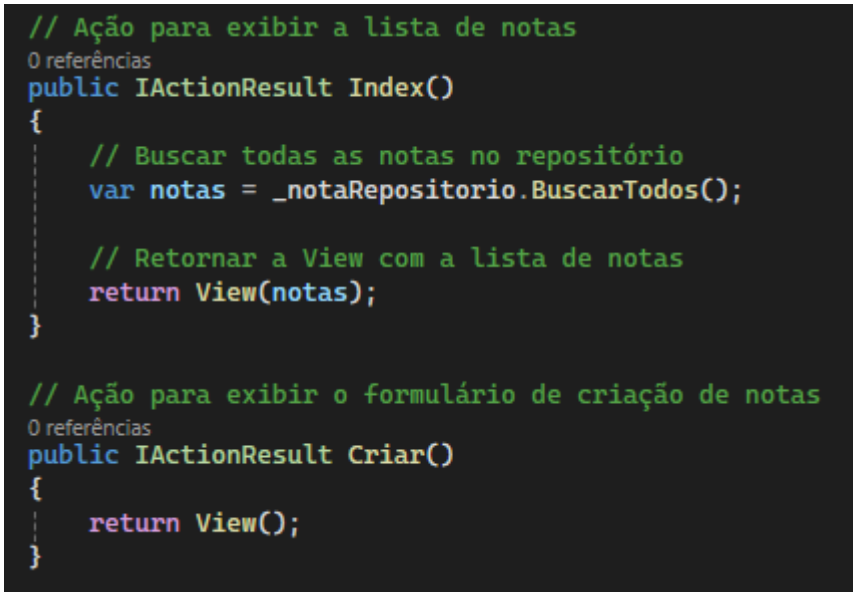
Essa classe **NotaModel** serve como uma estrutura fundamental para representar e manipular dados de anotações na aplicação. Cada instância dessa classe corresponde a uma nota específica, e seus membros definem os atributos essenciais associados a uma anotação, como título, conteúdo e data. Essa abordagem de modelagem de dados é crucial para o funcionamento eficiente e organizado da aplicação "Meu Bloco de Notas".

A camada de Visualização **(View)** assume o papel de apresentar os dados ao usuário, projetada para ser independentemente acoplada da lógica de negócios, a view proporciona uma experiencia de usuário clara, exibindo as informações processadas pela camada Model.

O Controlador **(Controller)** age como intermediário entre a lógica de negócios e apresentação, ao receber solicitações do usuário, o controlador coordena a interação entre a Modelo e a Visualização.

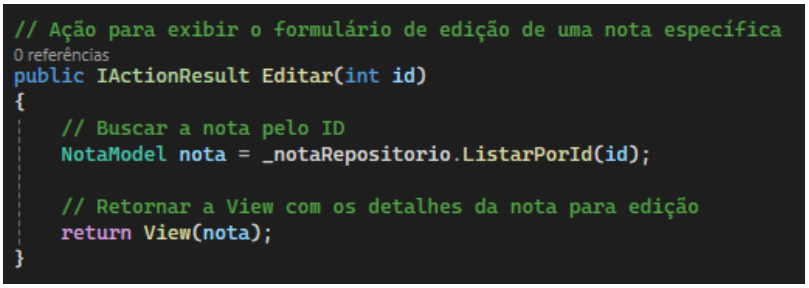
```
// Ação acionada quando o formulário de edição é enviado (HTTP POST)
[HttpPost]
0 referências
public IActionResult Alterar(NotaModel nota)
{
    try
    {
        // Verificar se o modelo é válido antes de atualizar
        if (ModelState.IsValid)
        {
            // Atualizar a nota no repositório
            _notaRepositorio.Atualizar(nota);

            // Mensagem de sucesso para exibir na próxima requisição
            TempData["MensagemSucesso"] = "Anotação Alterada com Sucesso !";

            // Redirecionar para a lista de notas (Index)
            return RedirectToAction("Index");
        }

        // Se o modelo não for válido, retornar para a View de edição com os erros
        return View("Editar", nota);
    }
    catch (Exception error)
    {
        // Mensagem de erro em caso de falha
        TempData["MensagemErro"] = $"Erro ao atualizar a Anotação, tente novamente, detalhe: {error.Message}";

        // Redirecionar para a lista de notas (Index)
        return RedirectToAction("Index");
    }
}
```

Fonte: Figura 4, Autor



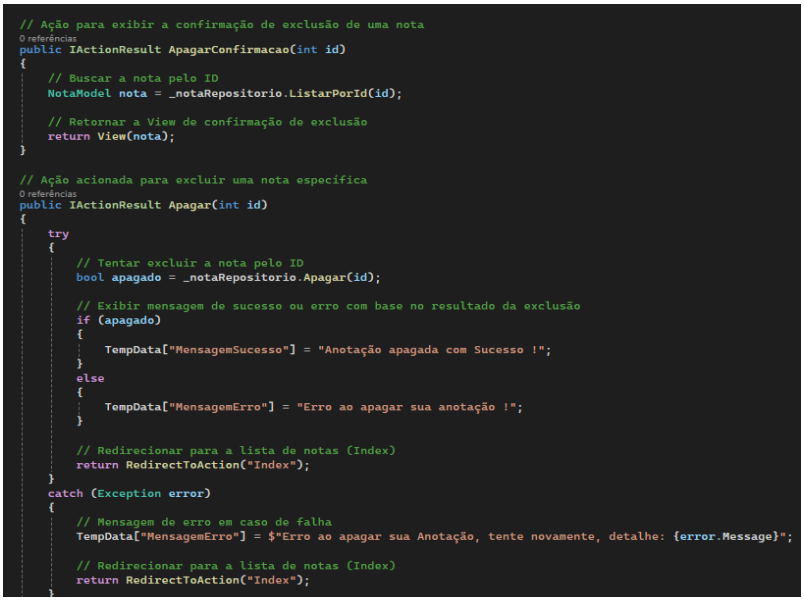
```
// Ação para exibir a lista de notas
0 referências
public IActionResult Index()
{
    // Buscar todas as notas no repositório
    var notas = _notaRepositorio.BuscarTodos();

    // Retornar a View com a lista de notas
    return View(notas);
}

// Ação para exibir o formulário de criação de notas
0 referências
public IActionResult Criar()
{
    return View();
}
```

Fonte: Figura 5, Autor



```
// Ação para exibir o formulário de edição de uma nota específica
0 referências
public IActionResult Editar(int id)
{
    // Buscar a nota pelo ID
    NotaModel nota = _notaRepositorio.ListarPorId(id);

    // Retornar a View com os detalhes da nota para edição
    return View(nota);
}
```

Fonte: Figura 6, Autor



```
// Ação para exibir a confirmação de exclusão de uma nota
0 referências
public IActionResult ApagarConfirmacao(int id)
{
    // Buscar a nota pelo ID
    NotaModel nota = _notaRepositorio.ListarPorId(id);

    // Retornar a View de confirmação de exclusão
    return View(nota);
}

// Ação acionada para excluir uma nota específica
0 referências
public IActionResult Apagar(int id)
{
    try
    {
        // Tentar excluir a nota pelo ID
        bool apagado = _notaRepositorio.Apagar(id);

        // Exibir mensagem de sucesso ou erro com base no resultado da exclusão
        if (apagado)
        {
            TempData["MensagemSucesso"] = "Anotação apagada com Sucesso !";
        }
        else
        {
            TempData["MensagemErro"] = "Erro ao apagar sua anotação !";
        }

        // Redirecionar para a lista de notas (Index)
        return RedirectToAction("Index");
    }
    catch (Exception error)
    {
        // Mensagem de erro em caso de falha
        TempData["MensagemErro"] = $"Erro ao apagar sua Anotação, tente novamente, detalhe: {error.Message}";

        // Redirecionar para a lista de notas (Index)
        return RedirectToAction("Index");
    }
```

Fonte: Figura 7, Autor

O código das Figuras 4, 5, 6 e 7 faz parte de um controlador em uma aplicação ASP.NET Core MVC para um bloco de notas, que realiza operações CRUD (Create, Read, Update, Delete) em notas. As principais funcionalidades de cada ação no controlador:

- **Index (Listagem de Notas):** Esta ação retorna uma visualização (View) que exibe a lista de todas as notas existentes, utiliza o repositório **_notaRepositorio** para buscar todas as notas. Passa a lista de notas para a View, onde será renderizada**.**
- **Criar (Processar Formulários de Criação – HTTP POST):** Recebe um objeto **NotaModel** como parâmetro, representando a nota a ser criada, adiciona a nota ao repositório **_notaRepositorio**, define uma mensagem de sucesso usando **TempData** para ser exibida na próxima requisição. Redireciona para a lista de notas (**Index**).
- **Editar (Exibir Formulário de Edição de uma Nota):** Recebe o ID de uma nota como parâmetro, busca a nota pelo ID no repositório, retorna a visualização para exibir o formulário de edição preenchido com os detalhes da nota.
- **Alterar (Processar Formulários de Edição – HTTP Post):** Recebe um objeto **NotaModel** como parâmetro, representando a nota a ser editada, verifica se o modelo é válido antes de realizar a atualização, atualiza a nota no repositório **_notaRepositorio**, define uma mensagem de sucesso usando **TempData** para ser exibida na próxima requisição, redireciona para a lista de notas (**Index**) se a atualização for bem-sucedida; caso contrário, retorna para a View de edição com os erros.
- **ApagarConfirmação (Exibi Confirmação de Exclusão de uma Anotação):** Recebe o ID de uma nota como parâmetro, busca a nota pelo ID no repositório, Retorna a visualização para exibir uma confirmação de exclusão.
- **Apagar (Processar Exclusão de Uma Anotação):** Recebe o ID de uma nota como parâmetro, tenta excluir a nota pelo ID usando o repositório **_notaRepositorio**, define mensagens de sucesso ou erro usando **TempData** com base no resultado da exclusão, redireciona para a lista de notas (**Index**).

Essa estrutura facilita a expansão e manutenção da aplicação “Meu Bloco de Notas”, permitindo uma abordagem modular e uma clara separação de responsabilidades entre os

componentes, o ASP.NET Core MVC, ao adotar esse padrão arquitetônico, contribui para o desenvolvimento de uma aplicação web sólida, eficiente e de fácil evolução.

2.3 O que é e para que serve CRUD

No âmbito do desenvolvimento web, frequentemente nos deparamos com o conceito essencial conhecido como CRUD, que é a sigla para Criar, Ler, Atualizar e Excluir. Essas operações formam o núcleo das funcionalidades que esperamos de modelos e APIs. Em suma, o CRUD nos fornece as ferramentas necessárias para criar recursos, visualizar informações existentes, atualizar dados e remover elementos quando necessário,

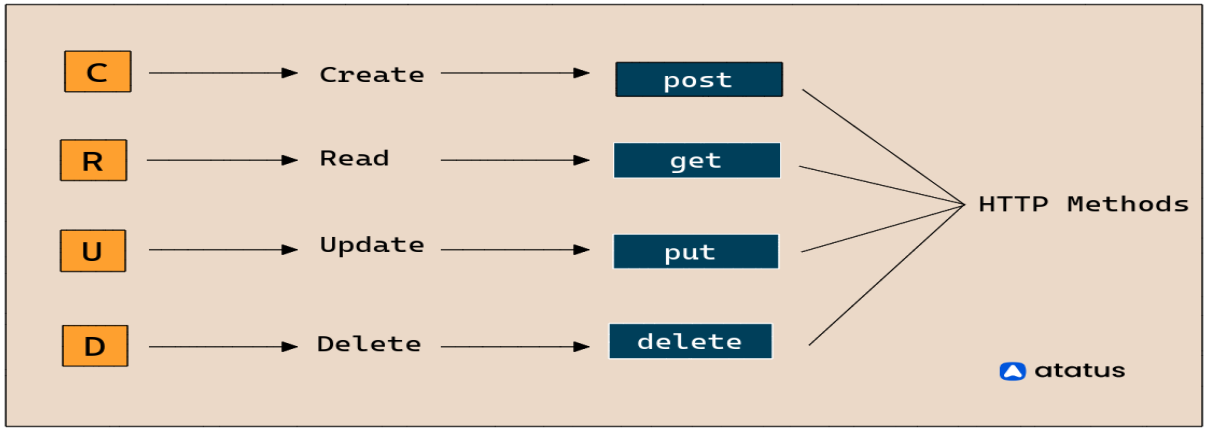
Os modelos APIs são projetados com o objetivo de oferecer essas quatro funcionalidades fundamentais, o que significa o processo de interação com os recursos. Esse acrônimo, CRUD, é amplamente utilizado na comunidade de cientistas da computação como uma maneira de descrever essas operações básicas. É importante notar que, para ser considerado um modelo completo, ele não deve ir além dessas quatro operações. Se uma ação não pode ser explicada por meio do Crias, Ler, atualizar ou Excluir, é provavelmente que mereça seu próprio modelo dedicado

O conceito do CRUD é amplamente empregado no desenvolvimento de aplicações web, fornecendo uma base sólida que os desenvolvedores podem recorrer quando construir modelos complexos e funcionais. Nesse cenário, é fundamental que o sistema ofereça meios claros para realizar cada operação do CRUD:

1. **Criar:** Sempre que uma nova informação de funcionário é adicionada ao banco de dados, uma função especifica deve ser acionada. Essa função receberá os dados receberá os dados relevantes, como “posição”, “departamento” e “número de telefone”, do aplicativo que a chama. Como resultado, uma nova entrada com essas informações será inserida nos registros de funcionários, junto com um ID exclusivo para acesso futuro.
2. **Ler:** Para a operação de leitura, é essencial disponibilizar um método que permita visualizar todos os funcionários armazenados no banco de dados. Essa função retornara os dados sem efetuar nenhuma modificação nas informações dos funcionários, onde podemos consultar ou atualizar informações como cargo, departamento ou número de telefone.

3. **Atualizar:** Quando for necessário modificar informações de um funcionário, uma função de atualização deve estar disponível. Ela aceitara os dados atualizados, como “posição”, “departamento” e “número de telefone”, e aplicará as alterações ao registro do funcionário correspondente.
4. **Excluir:** Finalmente, a operação de exclusão é crucial. Deve ser possível remover um funcionário no banco de dados por meio de uma função específica. A identificação do funcionário a ser excluído será baseado em critérios como “posição”, “departamento” e/ou “número de telefone”. Após execução dessa função, o registro do funcionário deverá manter detalhes inalterados exceto pelo funcionário que foi excluído.

FONTE Figura 8: imagem apresenta operações de CRUD.



Nesse contexto, os códigos de status HTTP desempenham um papel fundamental na comunicação entre o cliente e o servidor, como mostrado na Figura 8. Eles refletem as respostas na comunicação entre cliente e servidor. Eles refletem as respostas do servidor às socializações do navegador, permitindo que ambos interajam de maneira eficaz. Entender esses códigos de status ajudam a diagnosticar problemas e otimizar a experiencia do usuário, facilitando o acesso ao site e a comunicação com os mecanismos de pesquisa.

Em resumo, o CRUD é uma abordagem que simplifica o desenvolvimento web, permitindo a criação , leitura, atualização e exclusão de recursos dr forma eficiente. Além disso, os códigos de status HTTP desempenham um papel crucial na comunicação entre cliente e servidor, proporcionando uma experiencia mais suaves para os usuários. Portanto, é essencial entender e aplicasr adequadamente esses conceitos ao desenvolver aplicações web.

No contexto da aplicação web “Meu Blocos de Notas”, a abordagem CRUD desempenha um paopel cundamental ao possibilitar a criação, leitura, atualização e exclusão de notas de forma eficiente e organizada. Por meio dessa estrutura, os usuários podem criar novas

notas, acessar e visualizar informações exixtentes, atualizar o conteúdo das notas conforme necessário e até mesmo excluir notas que não são mais relevantes. Isso garante uma experiencia intuitiva e eficaz para o gerenciament de notas, seguindo os principios do CRUD para fornecer funcionalidades completas acessiveis aos usuários.

2.4 Linguagem C#

O surgimento do C# remonta ao final da década de 1990, quando a Microsoft, antecipando a era da computação na internet, identificou a necessidade premente de uma linguagem de programação moderna e robusta para competir com o Java da Sun Microsystems. Sob a liderança de Anders Hejlsberg, renomado arquiteto de linguagens de programação com experiência no desenvolvimento do Turbo Pascal e Delphi, o C# começou a ser concebido em meados de 1999.

Em 2002, o C# oficialmente lançado juntamente com o .Net Framework, proporcionando um ambiente de execução unificado para diversas linguagens de programação. A Microsoft, visando aumentar a aceitação e interoperabilidade, submeteu o C# à Ecma international e à ISO em 2021, resultando na padronização da linguagem.

O Visual Studio .NET, lançado no mesmo ano, tornou-se o ambiente de desenvolvimento integrado para o C# e outras linguagens .NET, oferecendo suporte abrangente para desenvolvimento e depuração de aplicativos. A evolução contínua do C# trouxe inovações como LINQ (Consulta Integrada à Linguagem), expressões lambda e tipos dinâmicos em versões posteriores. Em um marco significativo, a Microsoft anunciou, em 2014, a decisão de tornar o C# e o .NET Core open source. Essa mudança abriu as portas para execução do C# em diversas plataformas, incluindo Windows, Linux e macOS. O ano 2020 marcou o lançamento do .NET 5, consolidando o ecossistema .NET em uma única plataforma de desenvolvimento. Em 2021 o .NET 6 foi introduzido, apresentando aprimoramentos adicionais e solidificando ainda mais a posição do C# no desenvolvimento moderno de software

Ao longo de sua trajetória, o C# estabeleceu-se como uma linguagem e versátil, amplamente empregada no desenvolvimento de aplicativos desktop, web, móveis e em diversos domínios, desde sistemas embarcados até projetos de inteligência artificial. Esta análise cronológica destaca os principais marcos na evolução do C#, destacando sua importância no cenário de programação moderna.

2.5 Funcionalidades HTML e CSS

HTML, a linguagem de Marcação de Hipertext, desempenha um papel fundamental no desenvolvimento de páginas da web. Embora seja uma linguagem de computador, não se enquadra na categoria de linguagem de programação, como Java ou Python. O HTML foi concebido pela combinação dos padrões Hy Time e SGML. O HyTime lida com a representação estruturada de conteúdos hipermídia e multimidia, como áudio e vídeo, conectando os elementos por hiperlinks. Por outro lado, o SGML é um padrão de formatação de texto que ao longo do tempo se mostrou uma ferramenta útil na transformação de documentos e hiper objetos, revelando suas interconexões.

Benefícios do HTML

- Fácil aprendizado, escrita e codificação.
- Sintaxe flexível.
- Suporta o uso de modelos.
- Compatível com os navegadores.
- Amplamente utilizado em sites.
- Sua sintaxe se assemelha à do XML, uma linguagem comum para armazenamento de dados.
- É gratuito.

Vamos explorar uma analogia para entender melhor o papel do CSS no desenvolvimento web. Imagine-se como um construtor encarregado erigir uma casa. Você constrói uma estrutura, ergue as paredes, coloca o telhado e cria as aberturas para portas e janelas, deixando a base pronta. Essa estrutura básica pode ser comparada ao HTML de um site, que engloba o conteúdo, texto e descrições, no entanto, falta algo crucial: o acabamento, os detalhes como pisos, janelas, portas, revestimento e as ores das paredes não estão cobertos pelo HTML, é aqui que entra o CSS.

CSS, que significa Cascading Style Sheet, desempenha um papel fundamental no desenvolvimento web, pois sua principal função é adicionar estilo estética os elementos criados em HTML. O CSS cuida da aparência visual de um site, incluindo cores, fontes e escapamentos entre parágrafos, formatação de tabelas, layouts variedades, ajuste de imagens e muito mais.

Embora a aparência de um site possa aparecer secundário, ela desempenha um papel vital no mundo da web, foi por essa razão que o Word Wide Web Consortium (W3C) concebeu

o CSS em 1996 e foi amplamente aceito globalmente, W3C percebeu que o HTML sozinho não poderia atender as demandas de alterações visuais, pois seu foco estava principalmente na estrutura, tentativas anteriores, como a introdução da tag <font> na versão 3.2 do HTML, levaram a problemas e compilações para os desenvolvedores.

É por isso que o CSS se tornou a solução, ele permitiu a separação da estrutura de estilo, tornando mais fácil e eficiente a estilização de elementos web, mesmo após mais de 25 anos, a relação entre HTML e CSS continua forte, com HTML cuidando da estrutura e o CSS tornando visualmente atraente. Alguns podem considerar o CSS como uma mera camada estética, mas a realidade é que, sem ele, um site assemelharia uma casa abandonada, o código CSS pode ser incorporada diretamente nas telas HTML ou armazenado em seu arquivo separado, a última opção permite alterar o estilo da aplicação de forma mais eficiente, bastando editar arquivos CSS vinculado quando necessário, veja um exemplo do código de estilização.

Benefícios do CSS

- Com um único arquivo pode ser vinculado em várias, páginas economizando largura de bandas
- As mudanças na aparência visual de páginas são simples e rápidas, sem afetar o conteúdo.
- Códigos mais limpos e organizados, graças a simplicidades das propriedades e atributos.
- Folhas de estilo multifuncionais.

Na aplicação do “Meu Blocos de Notas”, foi implementada o padrão Razor, aonde que é uma característica fundamental do ASP.NET Core MVC, desempenha um papel crucial na construção da aplicação web “Meu Bloco de Notas”, ao incorporar ‘@’ diretamente na paginas web, o Razor proporciona uma integração fluida entre código C# do servidor e o conteúdo HTML apresentando ao usuário, esta abordagem dinâmica e intuitiva permite a criação de páginas interativas e reativas, tornando o desenvolvimento mais eficiente simplificado.

Os trechos de código fornecidos são exemplos claros do uso do padrão Razor na aplicação com modelos de dados, o Razor oferece uma sintaxe concisa legível, promovendo uma fácil compreensão da lógica do servidor incorporada nas páginas web, neste contexto, exploremos como o padrão Razor é empregado em diversas páginas da aplicação, como a confirmação da exclusão, na edição das anotações, e nas listagens dessas anotações. Ao compreender a natureza flexível e poderosa do Razor, é responsável criar páginas web reativas,

funcionais e visualmente agradáveis, proporcionando uma experiencia eficiente aos usuários “Meu Blocos de Notas”

```
@model List<NotaModel>
@{
    ViewData["Title"] = "Listagem de Anotações";
}
```

FONTE Figura 9

```
<form>
    <h1 class="display-4">Listagem de Anotações</h1>

    <br />
    <table class="table table-striped text-center" id="table-notas">
        <thead>
            <tr>
                <th scope="col">#</th>
                <th scope="col">Data</th>
                <th scope="col">Titulo</th>
                <th scope="col">Anotação</th>
                <th scope="col"></th>
            </tr>
        </thead>
        <tbody>
            @if (Model != null && Model.Any())
            {
                foreach (NotaModel anotacao in Model)
                {
                    <tr>
                        <th scope="row">@anotacao.Id</th>
                        <td>@anotacao.DataAnotacao</td>
                        <td>@anotacao.Titulo</td>
                        <td>@anotacao.ConteudoAnotacao</td>
                        <td>
                            <div class="btn-group" role="group" aria-label="Basic mixed styles example">
                                <a role="button" class="btn btn-outline-danger" asp-route-id="@anotacao.Id"
                                asp-controller="Nota" asp-action="ApagarConfirmacao">Apagar</a>

                                <a role="button" class="btn btn-outline-info" asp-route-id="@anotacao.Id"
                                asp-controller="Nota" asp-action="Editar">Editar</a>
                            </div>
                        </td>
                    </tr>

                }
            }
        </tbody>
    </table>

</form>
```

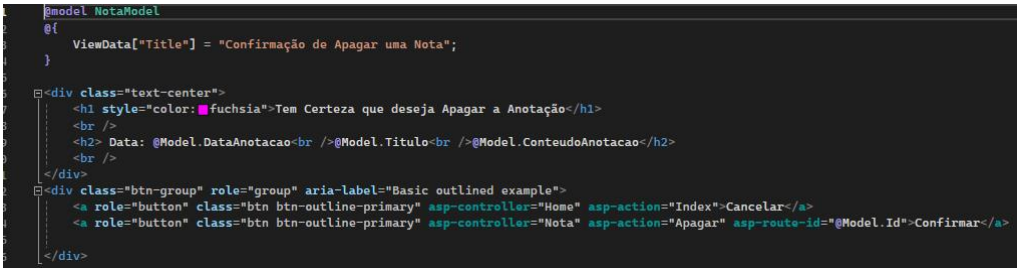
Fonte: Figura 10, Autor

```
@model NotaModel

@{
    ViewData["Title"] = "Editar Anotação";
}


<div class="text-center">
    <h1 class="display-4">Editar Anotação</h1>
</div>
<form asp-controller="Nota" asp-action="Alterar" method="post">

    <input type="hidden" asp-for="Id" />

    <div class="mb-3">
        <label for="exampleFormControlInput1" class="form-label">Titulo</label>
        <input type="text" asp-for="Titulo" class="form-control" id="exampleFormControlInput1">

    </div>
    <div class="mb-3">
        <label for="exampleFormControlInput1" class="form-label">Data</label>
        <input type="date" asp-for="DataAnotacao" class="form-control" id="exampleFormControlInput1">

    </div>
    <div class="mb-3">
        <label for="exampleFormControlTextarea1" class="form-label">Anotação</label>
        <textarea asp-for="ConteudoAnotacao" class="form-control" id="exampleFormControlTextarea1" rows="3"></textarea>


        <br />
        <div class="btn-group" role="group" aria-label="Basic radio toggle button group">
            <button type="submit" class="btn-check btn-lg btn-block" name="btnradio" id="btnradio1"></button>
            <label class="btn btn-outline-primary" for="btnradio1">Editar</label>

            <button type="button" class="btn-check btn-lg btn-block" name="btnradio" id="btnradio2"></button>
            <a class="btn btn-outline-primary" for="btnradio2" asp-action="Index" asp-controller="Home">Voltar</a>
        </div>
    </div>
</form>
```

Fonte: Figura 11, Autor

```
@model NotaModel
@{
    ViewData["Title"] = "Adicionar Anotação";
}

<div class="text-center">
    <h1 class="display-4">Adicionar uma Nova Anotação</h1>
</div>

<form asp-controller="Nota" asp-action="Criar" method="post">
    @if (TempData["MensagemSucesso"] != null)
    {

        <div class="alert alert-success" role="alert">
            <button type="button" class="btn btn-danger btn-sm close-alert" arial-label="Close">X</button>
            @TempData["MensagemSucesso"]
        </div>
    }
    @if (TempData["MensagemErro"] != null)
    {

        <div class="alert alert-success" role="alert">
            <button type="button" class="btn btn-danger btn-sm close-alert" arial-label="Close">X</button>
            @TempData["MensagemErro"]
        </div>
    }
    <br />
    <div class="mb-3">
        <label for="exampleFormControlInput1" class="form-label">Titulo</label>
        <input type="text" asp-for="Titulo" class="form-control" id="exampleFormControlInput1">
    </div>
    <div class="mb-3">
        <label for="exampleFormControlInput1" class="form-label">Data</label>
        <input type="date" asp-for="DataAnotacao"  class="form-control data" id="exampleFormControlInput1">
    </div>
    <div class="mb-3">
        <label for="exampleFormControlTextarea1" class="form-label">Anotação</label>
        <textarea asp-for="ConteudoAnotacao" class="form-control" id="exampleFormControlTextarea1" rows="10"></textarea>
    </div>

    <div class="btn-group" role="group" aria-label="Basic radio toggle button group">
        <button type="submit" class="btn-check btn-lg btn-block" name="btnradio" id="btnradio1"></button>
        <label class="btn btn-outline-primary" for="btnradio1">Salvar</label>
```

Fonte: Figura 12, Autor



```
@model NotaModel
@{
    ViewData["Title"] = "Confirmação de Apagar uma Nota";
}

<div class="text-center">
    <h1 style="color:fuchsia">Tem Certeza que deseja Apagar a Anotação</h1>
    <br />
    <h2> Data: @Model.DataAnotacao<br />@Model.Titulo<br />@Model.ConteudoAnotacao</h2>
    <br />
</div>
<div class="btn-group" role="group" aria-label="Basic outlined example">
    <a role="button" class="btn btn-outline-primary" asp-controller="Home" asp-action="Index">Cancelar</a>
    <a role="button" class="btn btn-outline-primary" asp-controller="Nota" asp-action="Apagar" asp-route-id="@Model.Id">Confirmar</a>

</div>
```

Fonte: Figura 13, Autor

Explicação do trecho do código das Figuras 9, 10, 11, 12 e 13:

- **Listagem de Anotações (ListagemAnotacoes.cshtml):** Exibe uma lista de anotações em uma tabela, com colunas para data, título e o conteúdo da anotação, possui botões para apagar ou editar cada anotação, redirecionando para as páginas correspondentes no controlador ‘Nota’, apresenta mensagens de sucesso ou erro utilizando TempData, inclui um botão para adiciona uma nova anotação
- **Editar Anotação (EditarAnotacao.cshtml):** Similar á página de adoção, mas destinada á edição de uma anotação existente, utiliza a model ‘NotaModel’ para preencher os campos existentes, inclui um botão para editar e outro para voltar a página inicial.
- **Adicionar Anotação (AdicionarAnotacao.cshtml):** Formulário para adicionar uma nova anotação, usando a model **NotaModel**. Utiliza o método POST para enviar os dados para o controlador **Nota** e a ação **Criar**, exibe mensagens de sucesso ou erro, caso existam, utilizando a TempData, fornece campos para o título, data e conteúdo da anotação, além de botões para salvar e voltar para a página inicial.
- **Confirmação de Apagar uma Nota (ConfirmacaoApagarNota.cshtml):** A página exibe uma mensagem de confirmação para apagar uma nota específica. Utiliza a model **NotaModel** para acessar os dados da nota. Apresenta dois botões, um para cancelar (**Cancelar**) e outro para confirmar (**Confirmar**) a exclusão, ambos com links para controladores correspondentes.

O padrão Razor permite a mistura de código C# com HTML de forma fluída. As marcações **@** indicam quando o código C# está sendo incorporado nas páginas, facilitando a interação entre o código do servidor e o conteúdo visual apresentado ao usuário. Isso torna o desenvolvimento mais dinâmico e simplifica a criação de páginas interativas.

Na aplicação “Meu Bloco de Notas”, a estilização da interface é cuidadosamente planejada e implementada utilizando CSS de maneira estruturada e responsiva. A tipografia do corpo da página é escolhida para oferecer uma leitura clara e adaptável, enquanto o cabeçalho se destaca com uma coloração descontraída usando uma imagem de um tipo de “lembrete”, proporcionando uma identidade visual destinta. A estrutura página inicial, tem uma animação com uma pequena transição das palavras “Meu Bloco de Notas”. A paleta de cores suaves contribui para uma experiencia visual agradável, enfatizando a clareza e legibilidade do conteúdo apresentado, o cuidado com os detalhes na estilização não apenas contribui para a estética visual da interface, mas também desempenha um papel crucial na usabilidade, tornando a interação do usuário mais intuitiva e agradável. Essa abordagem reflete a busca constante por uma experiencia de usuário aprimorada e alinhada com as melhores práticas de design de interfaces. Nas Figuras de 14 até 21 mostras como foi feito as estilizações da aplicação “Meu Bloco de Notas”.

```css
/* Definição do tamanho padrão da fonte para o elemento HTML */
html {
    font-size: 14px;
}

/* Mídia query para ajustar o tamanho da fonte em telas maiores */
@media (min-width: 768px) {
    html {
        font-size: 16px;
    }
}

/* Estilização para realce visual em elementos com foco */
.btn:focus,
.btn:active:focus,
.btn-link.nav-link:focus,
.form-control:focus,
.form-check-input:focus {
    box-shadow: 0 0 0 0.1rem white, 0 0 0 0.25rem #258cfb;
}

/* Estilo para o corpo do documento e posicionamento do HTML */
html {
    position: relative;
    min-height: 100%;
}

body {
    margin-bottom: 60px;
}
```

Fonte: Figura 24: Autor

```
/* Reset de estilos padrão para todos os elementos */
* {
    margin: 0 auto;
    padding: 0;
    box-sizing: border-box;
}

/* Estilo de fundo do corpo e imagem de plano de fundo */
body {
    background-image: url(https://img.freepik.com/vetores-gratis/nota-de-papel-sobre-fundo-de-memphis_53876-99284.jpg);
    background-size: 100%;
}

/* Estilo para campos de entrada de dados e textarea */
input[type="data"],
input[type="text"],
textarea {
    /* Estilos comuns para os campos */
    width: 100%;
    padding: 20px;
    box-sizing: border-box;
    margin-bottom: 10px;
    border: 1px solid var(--corBorda);
    border-radius: 5px;
    font-size: 16px;
    height: 20px;
    cursor: pointer;
}
```

Fonte: Figura 15: Autor

```
/* Estilo específico para textarea com
    capacidade de redimensionamento vertical */
textarea {
    resize: vertical;
}

/* Estilo adicional para o campo de data */
input[type="data"]::after {
    padding-right: 10%;
}

/* Estilos para o botão de data */
.data {
    font-size: 20px;
    margin-left: 15px;
    border: none;
    cursor: pointer;
}

/* Estilos para o cabeçalho h1 com animações */
header h1 {
    /* Estilos comuns */
    width: 520px;
    text-align: center;
    font-size: 45px;
    color: white;
    text-shadow: 5px 5px 10px #000000;
    position: relative;
    /* Animação do título */
    animation: titulo 8s ease-out infinite;
    font-family: cursive;
}
```

Fonte: Figura 15: Autor

```
/* Animação personalizada do título */
@keyframes titulo {
    0% {
        color: #6894F7;
        left: -300px;
    }

    50% {
        color: #66EC3D;
        left: 0px;
    }

    100% {
        color: #F7E40F;
        left: 300px;
    }
}
```

Fonte: Figura 17: Autor

```
/* Estilos para a barra de navegação */
nav {
    margin-top: 8%;
    text-align: center;
    height: 80px;
}

    /* Estilos para a lista não ordenada dentro da navegação */
    nav ul {
        list-style: none;
        padding: 0;
    }

    /* Estilos para os itens de lista na navegação */
    nav li {
        font-size: 40px;
        margin: 0 5% 0 5%;
        padding: 2px 10px 2px 10px;
        display: inline-block;
        font-family: French Script MT;
        border-radius: 10%;
        border: 2px solid black;
    }

/* Efeito de escala e animação no hover dos itens de lista */
li:hover {
    -webkit-transform: scale(1.2);
    -moz-transform: scale(1.2);
    -ms-transform: scale(1.2);
    -o-transform: scale(1.2);
    transform: scale(1.2);
    animation: rosa 8s ease-out infinite;
}
```

Fonte: Figura 18: Autor

```css

/* Estilos para os links dentro dos itens de lista */
nav a {
    text-decoration: none;
    color: black;
}

/* Efeito de transição de cor no hover dos links */
nav li a:hover {
    -webkit-transition: all .5s ease;
    -moz-transition: all .5s ease;
    -ms-transition: all .5s ease;
    -o-transition: all .5s ease;
    transition: all .5s ease;
    color: white;
    text-shadow: 1px 1px 2px black, 0 0 25px blue, 0 0 5px darkblue;
}

/* Estilos para a imagem responsiva */
.imagem {
    max-width: 100%;
}

```

Fonte: Figura 19: Autor

```css
/* Estilos responsivos para telas menores que 480px */
@media all and (max-width: 480px) {
    /* Ajuste do tamanho do título e animação para telas pequenas */
    header h1 {
        font-size: 20px;
        text-align: left;
        animation: titulo 2s linear infinite;
    }

    /* Animação personalizada do título para telas pequenas */
    @keyframes titulo {
        0% {
            left: -5px;
        }

        50% {
            left: 0px;
        }

        100% {
            left: 15px;
        }
    }
}
```

Fonte: Figura 20: Autor

```css

/* Estilos responsivos para telas entre 480px e 760px */
@media all and (max-width: 760px) and (min-width: 480px) {
    /* Ajuste do tamanho do título e animação para telas médias */
    header h1 {
        font-size: 35px;
        text-align: left;
        animation: titulo 3s linear infinite;
    }

    /* Animação personalizada do título para telas médias */
    @keyframes titulo {
        0% {
            left: 5px;
        }

        50% {
            left: 0px;
        }
    }

    /* Ajuste do tamanho dos itens de lista para telas médias */
    nav li {
        font-size: 33px;
        display: block;
        margin: 20px 120px 20px 120px;
    }
}
```

Fonte: Figura 21: Autor

Explicação do Código CSS: Estilo Responsivo e Animações

O código fornecido nas imagens das Figuras 14 até figura 21 é um conjunto abrangente de estilos para uma página web, incorporando técnicas de design responsivo e animação para proporcionar uma experiencia visualmente atraente. Explorando as principais se seções do código e entender seu funcionamento.

- **Definição Global e Reset de Estilos:** O código começa com a definição global de estilos, como tamanho da fonte padrão para o elemento HTML e um reset de estilos padrão para todos os elementos **(‘ * ’),** isso garante uma base consistente.
- **Estilo Responsivo:** Utiliza a técnica query para ajustar o tamanho da fonte em diferentes larguras de tela (**@media (min-width: 768px),** isso proporciona uma experiencia de usuário adequada para dispositivos com telas variadas.
- **Realce Visual em Elementos com Foco:** Aplica um estilo de realce visual para elementos como botões e campos de formulário quando estão em foco.
- **Estilo de Corpo e Imagem de Fundo:** Define o estilo de fundo do corpo da página, incorporando uma imagem de fundo cobre 100% da largura da página.
- **Estilo de Campos de Entrada de Dados e Textarea:** Estabelece estilos uniformes para campos de entrada de dados e textarea, garantindo uma aparência consistente e responsiva.
- **Estilos de Cabeçalho (Header):** Aplica estilos ao cabeçalho da página, incluindo um título **(‘ h1 ’)** com uma animação personalizada (**@keyframes titulo**), a animação faz com que o título se mova lateralmente, alterando sua cor ao longo do tempo de transição.
- **Estilos da Barra de Navegação (Nav):** Configura estilos para a barra de navegação listando itens horizontalmente, os itens de lista (**nav li**) tem bordas arredondadas e um efeito de escala **(‘ :hover’).**
- **Estilo Responsivos para Telas Menores:** Inclui ajustes de estilo específicos para telas menores de 480px e telas entre 480px e 760px, isso garante uma experiencia de usuário otimizada em dispositivos móveis, tablets e computadores.
- **Animações Adicionais:** Introduz uma animação adicional **(‘@keyframes example’)** para demonstrar a capacidade de implementar transições de cores em elementos específicos.

O código CSS apresentado demonstra uma abordagem abrangente para estilizar uma página web de maneira responsiva e dinâmica, ao incorporar média queries, animação personalizada e estilos coerentes, o código busca criar uma experiência visualmente agradável em diferentes dispositivos, adaptando as demandas do designe moderno, o uso de técnicas avançadas como animação e transição, contribui, para uma apresentação elegante e envolvente do conteúdo.

2.6 Funcionalidades Banco de Dados SQL Server

Segundo DATE (2004, p. 10), um banco de dados "é uma coleção de dados persistentes, usada pelos sistemas de aplicação de uma determinada empresa ", em outras palavras, em outras palavras, um banco de dados é um local onde são armazenados os dados necessários à manutenção das atividades de determinada organização, sendo este repositório a fonte de dados para as aplicações atuais e as que vierem existir. Para ELMASRI e NAVATHE (2011, p. 3), na expressão Banco de Dados estão subentendidas as propriedades abaixo:

> "Um banco de dados representa algum aspecto do mundo real, às vezes chamado de minimundo ou de universo de discurso (UoD – Universe of Discurse). As mudanças no minimundo são refletidas no Banco de Dados. Um banco de dados é uma coleção logicamente coerente de dados com algum significado inerente. Uma variedade aleatória de dados não pode ser corretamente chamada de banco de dados. Um banco de dados é projetado, construído é populado com dados para uma finalidade específica. Ele possui um grupo definido de usuários e algumas aplicações previamente concebidas nas quais esses usuários estão interessados.

Assim, um banco de dados é um conjunto organizado de dados relacionados, criado com determinado objetivo e que atende uma comunidade de usuários. Sistema de Gerenciamento de Banco de Dados (SGBD) é um software que possui recursos para manipular as informações do banco de dados e interagir com usuário.

Além disso, é importante conceituar um sistema de banco de dados como um conjunto de quatro componentes fundamentais: dados, hardware, software e usuários. Como Date escreveu um sistema de banco de dados pode ser considerado como uma sala de arquivos eletrônicos.

Os objetivos de um sistema de banco de dados são o de isolar os usuários dos detalhes internos de banco de dados (promover a abstração de dados) e promover a independência dos dados em relação às aplicações, ou seja, tornar independente da aplicação, a estratégia de acesso e a forma de armazenamento. O sistema de banco de dados deve garantir uma visão totalmente

abstrata do banco de dados para o usuário, ou seja, para o usuário de banco de dados pouco importa qual unidade de armazenamento está sendo usado para guardar seus dados, contando que eles estejam disponíveis no momento necessário.

- **Nível de visão do usuário:** as partes do banco de dados que são acessíveis pelo usuário de acordo com as suas necessidades individuais ou em grupo.
- **Nível conceitual:** define quais dados estão armazenados no banco de dados e como eles se relacionam.
- **Nível físico:** trata da implementação real de como os dados são armazenados.

No entanto, é necessário descrever o banco de dados em um nível que depende do tipo específico de SGBD a ser utilizada, como SGBD relacional, nesse caso os dados são organizados em tabelas, como exemplificado nas tabelas abaixo. O modelo lógico do Banco de Dados Relacional deve definir as tabelas e os nomes das colunas que compõem essas tabelas, para nosso exemplo, poderíamos definir o modelo lógico da seguinte forma

Tabela Aluno (mat_aluno, nome, endereco)
Tabela Turma (cod_turma, sala, periodo)

No contexto da aplicação de “Meu Bloco de Notas”, o SQL Server será utilizado para gerenciar eficientemente o armazenamento e recuperação de notas, garantindo um acesso rápido e seguro aos dados, além disso, o SQL Server permite a organização dos dados em tabelas relacionadas facilitando a busca e a organização das notas, isso contribuirá para uma experiencia de uso mais eficaz e confiável da aplicação de bloco de notas, tornando mais simples a manutenção e a pesquisa de informações.

O trecho de código SQL apresentado desempenha um papel central na estruturação do banco de dados para a aplicação “Meu Bloco de Notas”. Ao criar e modificar a tabela “Nota”. A seguir, na figura 28 mostra a criação da tabela Nota, mediante a imagem da tabela na imagem da Figura 29.

Código SQL:

```
CREATE TABLE [dbo].[Nota](
    [Id] [int] IDENTITY(1,1) NOT NULL,
    [Titulo] [nvarchar](max) NULL,
    [ConteudoAnotacao] [nvarchar](max) NULL,
    [DataAnotacao] [nvarchar](max) NULL,
 CONSTRAINT [PK_Nota] PRIMARY KEY CLUSTERED
(
```

Fonte: Figura 22: Autor

	Nome da Coluna	Tipo de Dados	Permitir Nul...
🔑	Id	int	☐
	Titulo	nvarchar(MAX)	☑
	ConteudoAnotacao	nvarchar(MAX)	☑
	DataAnotacao	nvarchar(MAX)	☑
			☐

Fonte: Figura 23: Autor

3. MODELO DIAGRAMA CASOS DE USO

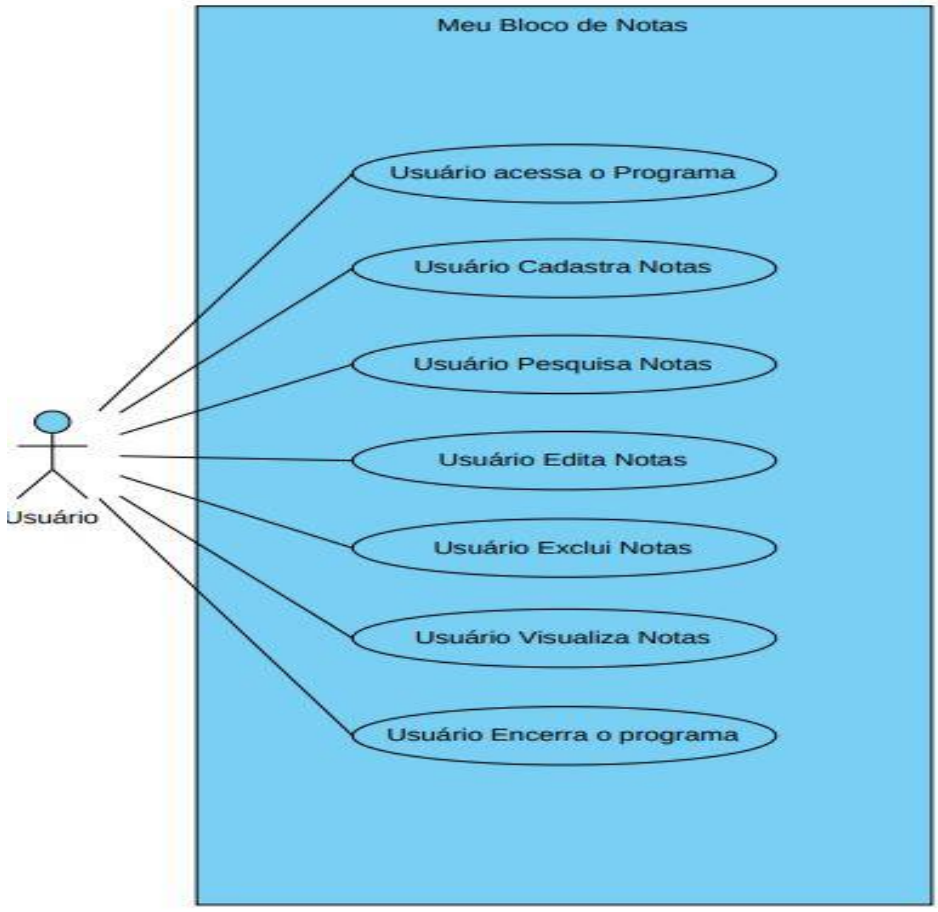
O desenvolvimento do aplicativo “Meu Bloco de Notas” incorpora práticas eficazes de modelagem, sendo essencial destacar a utilidade de diagramas de casos de uso. Estes desempenham um papel crucial na documentação das interações entre usuários e sistema, concentrando-se nas principais funcionalidades sem entrar em detalhes técnicos.

O Diagrama de Casos de Uso retrata as ações do sistema do ponto de vista do usuário, delineando as principais funcionalidades e suas interações. No contexto do “Meu Bloco de Notas”, os elementos são:

1. **Cenário:** Sequência de eventos quando o usuário interage com o sistema.
2. **Ator:** Representa o usuário do sistema, categorizado por tipos específicos de usuários.
3. **Use Case:** Refere-se as tarefas ou funcionalidades realizadas pelos atores (usuários).

4. **Comunicação:** Estabelece as conexões entre atores e casos de uso, delineando a interação.

Para ilustrar, consideramos um cenário inicial com o ator principal, como demonstrado na Figura 23:



FONTE: Figura 23: Autor

- Usuário:
  Ações Associadas:
  Acesso ao Programa
  Cadastro de Notas
  Pesquisa de Notas
  Editar Notas
  Excluir Notas

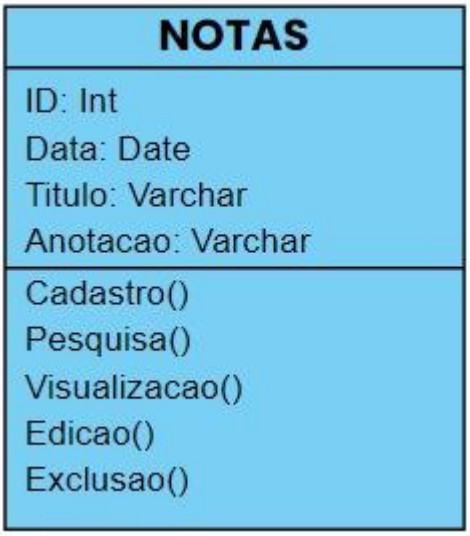
Visualizar Notas

Fechamento do Programa

Essas funcionalidades são mapeadas no Diagrama de Casos de Uso, proporcionando uma visão clara das interações entre o usuário e o sistema “Meu Bloco de Notas”. Este artefato não apenas simplifica a compreensão das funcionalidades principais, mas também serve como uma base para a criação do documento de requisitos, contribuindo para um desenvolvimento mais eficiente e centrado no usuário.

4. DIAGRAMA DE CLASSE UML

O diagrama de classes é uma ferramenta visual n modelagem de sistemas orientados objetos fundamentais para representar a estrutura estática do sistema. No contexto do desenvolvimento web, o aplicativo “Meu Bloco de Notas” se beneficia da clareza proporcionada por esse diagrama, que descreve as classes e relacionamento, como mostrado na figura 24.



Fonte: Figura 24 Autor

Significado das Classes:

- **ID (identificador):**

  Tipo: Inteiro (Int)

Representa identificação única

- **Data:**

  Tipo: Data (Date)

  Armazena a data associada a uma anotação no bloco de notas, é importante para organização por ordem de entrada.

- **Título:**

  Tipo: Varchar (texto variável).

  Contém o corpo das anotações propriamente dita, armazena o conteúdo textual da anotação.

Métodos Associados:

- **Cadastro():**

  Método responsável por adicionar novas entradas ao bloco de notas, incorporando informações como título, conteúdo e data.

- **Pesquisa():**

  Método destinado a buscar anotações com base em critérios específicos, oferecendo flexibilidades ao usuário para localizar informações.

- **Visualização():**

  Método que possibilita a visualização detalhada de uma anotação especifica exibindo seu título, conteúdo e data associada.

- **Edicao():**

  Método que permite a modificação de informações em uma anotação existente, possibilitando a atualização se título, conteúdo ou data.

- **Exclusao():**

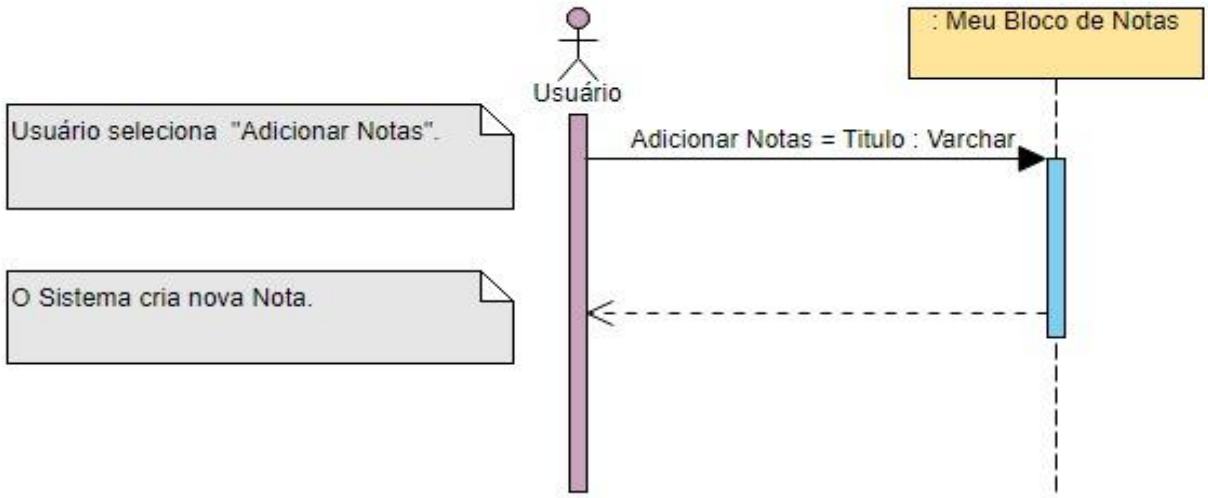
  Método encarregado de remover uma anotação de bloco de notas, eliminando-a de forma seguro e definitiva.

O diagrama de classes oferece uma apresentação estruturada e visual das entidades centrais no “Meu Bloco de Notas”, sendo crucial para o desenvolvimento eficaz. Ao destacar as propriedades e métodos das classes, proporcionar uma base sólida para a implementação do sistema, garantindo uma arquitetura robusta e coerente.

5. DIAGRAMA DE SEQUÊNCIA

No âmbito do desenvolvimento web para o aplicativo “Meu Bloco de Notas”, o Diagrama de Sequência é uma ferramenta valiosa para visualizar e compreender as interações dinâmicas entre os diferentes componentes do sistema. Ele ilustra a ordem e a natureza das trocas de mensagens entre objetos ao longo do tempo, destacando o fluxo de execução.

O objetivo do Diagrama de Sequência é empregado para representar cenários específicos de uso, demostrando como os objetivos colaboram entre si para atingir um objetivo. No contexto do “Meu Bloco de Notas”, ele proporciona uma visão clara das etapas envolvidas em operações críticas, como a adição, edição e exclusão de anotações, como demonstra na figura 25.



Fonte: Figura 25 Autor

As Sequencias de Ações:

- **Usuário Inicia Cadastro de Adicionar Notas:**
  O usuário interage com a interface do aplicativo, iniciando o processo de adicionar uma nota anotação.
- **Verificação de Dados e Atribuição de ID:**

A lógica verifica os dados fornecidos pelo usuário, atribui um ID único à nova anotação e prossegue com o armazenamento.

- **Atualização da Interface com a Confirmação de Adição de Notas:**
  A interface é notificada sobre o sucesso de adição das notas, atualizando a exibição para refletir a inclusão da nova anotação.
- **Usuário Realiza Edição em Anotação Existente:**
  O usuário seleciona uma anotação existente para edição.
- **Solicitação de Edição Enviada a Lógica:**
  A interface envia a solicitação de edição, indicando as alterações desejada na anotação selecionada.
- **Validação e Atualização de Dados:**
  A lógica valida as alterações propostas, atualiza os dados da anotação e confirma a conclusão da edição.
- **Atualização da Interface com Confirmação de Edição:**
  A interface reflete as modificações bem-sucedidas na anotação, proporcionando feedback ao usuário.

Diante essas informações destaca como o Diagrama de Sequência é uma ferramenta crucial para visualizar e compreender a dinâmica das operações essenciais no “Meu Bloco de Notas”, contribuindo para um desenvolvimento eficaz e uma experiencia de usuário fluida.

6. METODOLOGIA

A pesquisa possui características exploratórias para investigar a evolução do desenvolvimento web ao longo do tempo e descritivas para apresentar a aplicação “Meu Bloco de Notas” e seus elementos fundamentais. A abordagem exploratória permite uma compreensão aprofundada do cenário, enquanto a descritiva destaca as características e funcionalidades do aplicativo.

As informações foram obtidas por meio de uma revisão extensiva da literatura, incluindo artigos científicos, blogs especializados, sites de tecnologia, e materiais educacionais. Referências bibliográficas, tanto de autores acadêmicos quanto de fontes online confiáveis, foram consultas para garantir a validade e a abrangência das informações apresentadas no trabalho.

Os resultados serão apresentados de forma qualitativa, destacando as características, evolução e conceitos relacionados ao desenvolvimento web. Serão utilizadas descrições detalhadas para explicar a aplicação “Meu Bloco de Notas”, suas funcionalidades e as escolhas das tecnologias. A análise qualitativa proporcionará uma compreensão aprofundada do conceito, enfatizando a experiencia do usuário e a eficácia da solução proposta.

Este método integrado de pesquisa, combinando aspectos exploratórios e descritivos, oferece uma visão abrangente do desenvolvimento web, desde as mudanças tecnológicas até a implementação prática em um aplicativo específico. A diversidade de fontes contribui para uma base sólida de conhecimento e eficácia da aplicação “Meu Bloco de Notas”.

7. RESULTADOS E DISCUSSÕES

Com base nas informações fornecidas sobre a aplicação web "Meu Bloco de Notas", fica evidente que se trata de um sistema robusto desenvolvido utilizando ASP.NET Core MVC, um framework da Microsoft para o desenvolvimento web. O padrão Razor é claramente empregado nas marcações, permitindo a incorporação de código C# diretamente nas páginas web, facilitando a interação entre a lógica de apresentação e o código do lado do servidor.

A estrutura da aplicação segue o padrão Model-View-Controller (MVC), onde o modelo (NotaModel) representa os dados de uma nota, a visualização (View) é responsável pela apresentação da interface ao usuário, e o controlador (Controller) gerencia as interações do

usuário, respondendo às requisições e manipulando os dados conforme necessário. O repositório de notas, implementado pela interface INotaRepositorio, segue o padrão de repositório, abstraindo o acesso aos dados e permitindo a flexibilidade para diferentes implementações de armazenamento. Essa abordagem facilita a manutenção, testabilidade e escalabilidade da aplicação. Quanto às funcionalidades específicas da aplicação, as ações no controlador (NotaController) desempenham papéis cruciais.

A ação Index recupera todas as notas do repositório para exibição na lista, enquanto as ações Criar, Editar, Alterar, apagar gerenciam a adição, edição, atualização e exclusão de notas, respectivamente. A interface de usuário (UI) é construída de maneira responsiva, apresentando formulários para entrada e exibição de notas. A aplicação utiliza mensagens temporárias (TempData) para fornecer feedback ao usuário após a execução de ações, como salvar ou excluir uma nota.

No entanto, vale ressaltar que a aplicação pode se beneficiar de melhorias, como validação mais detalhada dos dados de entrada, tratamento mais sofisticado de erros, e possivelmente a implementação de recursos de segurança, dependendo do contexto de uso. Em termos de discussões, a modularidade da aplicação, evidenciada pela separação clara entre a lógica de apresentação, a lógica de negócios e a camada de dados, contribui para a manutenção e escalabilidade. Além disso, a aplicação incorpora boas práticas, como o uso de interfaces e o emprego do padrão de repositório

Em resumo, o desenvolvimento da aplicação "Meu Bloco de Notas" demonstra uma sólida compreensão dos conceitos de desenvolvimento web, utilizando tecnologias modernas e boas práticas de programação para criar uma aplicação funcional e eficiente.

8. CONCLUSÃO

O trabalho científico sobre o desenvolvimento do aplicativo “Meu Bloco de Notas” destaca a evolução significativa do desenvolvimento web, desde as páginas estáticas da Web 1.0 até a dinâmica e interativa Web 3.0 e 4.0. A transição tecnológica, ilustra pela aplicação prática, evidencia as mudanças conceituais e tecnológicas de moldaram essa evolução.

Os conceitos fundamentais, como CRUD, HTML, CSS e SQL Server, são peças-chave na construção da aplicação. O CRUD facilita as operações essenciais de criação, leitura, atualização e exclusão de notas, proporcionando uma experiencia intuitiva aos usuários. As

funcionalidades HTML e CSS contribuem para uma interface atraente e intuitiva, enquanto o SQL Server viabiliza o armazenamento e manipulação eficiente de dados. Ao adotar o padrão Model-View-Controller (MVC) no desenvolvimento de um aplicativo web utilizando ASP.NET Core, os benefícios são notáveis em termos de organização, manutenção e eficiência. A estrutura clara e modular proporcionada pelo MVC permite uma gestão mais eficaz do código-fonte, facilitando a identificação e correção de problemas, além de agilizar a implementação de novos recursos.

A separação distintiva entre os componentes - Modelo, Visualização e Controlador - não apenas facilita a escalabilidade do projeto, permitindo ajustes em partes específicas sem afetar o todo, mas também melhora a testabilidade do sistema. A capacidade de realizar testes unitários de forma isolada em cada componente contribui para a robustez e confiabilidade do aplicativo. A utilização de modelos que mapeiam diretamente para tabelas de banco de dados, visualizações em Razor que combinam HTML com código C# para renderização dinâmica e controladores como intermediários coesos, demonstra uma abordagem coesa e eficiente para o desenvolvimento web.

Em resumo, a aplicação do padrão MVC no ASP.NET Core não apenas simplifica o processo de desenvolvimento, mas também resulta em um sistema mais organizado, fácil de manter e capaz de se adaptar às demandas evolutivas, promovendo, assim, uma experiência de desenvolvimento web mais eficiente e sustentável. Foi concluído a importância dessas tecnologias integradas na criação de uma solução robusta e versátil para a questão de notas online. À medida que a evolução da web continua, espera-se mais inovações e avanços que impulsionaram a interatividade, eficiência e acessibilidade nas aplicações web. O “Meu Bloco de Notas” representa um exemplo prático desse progresso, oferecendo uma solução completa e eficaz para a necessidades dos usuários.

9. REFERÊNCIAS BIBLIOGRÁFICAS

ATATUS. **CÓDIGO DE STATUS HTTP A SER USADO PARA OPERAÇÕES CRUD** Atatus.2022.

AWARI, Code. **HTML E CSS: ENTENDA O QUE SÃO E PARA QUE SERVEM**. Awari. 2022.

RAMOS, Diego. **INTRODUÇÃO Á LINUAGEM DE PROGRAMAÇÃO C#**. Dio, Academia PME de Educação. 2022.

CARÇALTO, Celio Oliveira. **A EVOLUÇÃO DO DESENVOLVIMENTO WEB.** Linkedin. 2023.

CESAR, **A NOVA ERA DA INTERNET: O QUE ESPERAR DA WEB 3.0.** Cesar. 2022.

ANDRADE, Claudenir. **C# PARA INICIANTES.** Agência Hex. 2021.

CURVELO, Rakky. **DA WEB 1.0 Á 4.0: CONHECA A EVOLUÇÃO E ENTENDA AS DIFERENÇAS.** HubSpot. 2022.

MICROSOFT. **O QUE É ASP.NET CORE**. Microsoft. 2022.

MICROSOFT. **LINGUAGEM C#**. Microsoft. 2022..

ELMASRI, Ramez; NAVATHE, Shamkant B. **SISTEMAS DE BANCO DE DADOS**. 6. ed. São Paulo: Addison Wesley, 2011.

GARCIA, Guilherme**. CRUD: O QUE É, COMO CRIAR, COMO USAR, BACEND E CUIDADOS NECESSÁRIOS.** MOD Mercado Digital Blog. 2023.

GARCIA, Carlos. **COMO O DESENVOLVIMENTO WEB MUDOU NOS ULTIMOS 10 ANOS.** AppMaster, 2022.

HOSTGATOR. **CONHEÇA O HTML UMA DAS LINGUAGENS MAIS USADAS NA WEB**. HostGator. 2023.

HOSTBITS. **APLICAÇÃO WEB: ENTENDA O QUE É E COMO FUNCIONA.** Hostbits Blog. 2022.

MATOS, Évilin. **AVANÇO DA TECNOLOGIA NOS ULTIMOS 10 ANOS: DE CASA AO TRABALHO**. Blog Runrun.it. 2021.

NOLETO, Cairo. **CRUD: AS 4 OPERAÇÕES BÁSICAS DO BANCO DE DADOS.** Tecnologia. 2022.

SACRAMENTO, Gabriel. **APLICAÇÃO WEB: O QUE É, DIFERENÇA PARA WEBSITE, COMO FUCNIONA E MAIS.** Talent Network Blog. 2022.

SENHOR. **ARQUITERURA WEB – RESUMO COMPLETO.** Medium. 2023.