

**UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CURSO DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS**

AUGUSTO CESAR TONELLO GIACOMINI

SISTEMA WEB PARA DIVULGAÇÃO DE EVENTOS

TRABALHO DE CONCLUSÃO DE CURSO

**PATO BRANCO
2017**

AUGUSTO CESAR TONELLO GIACOMINI

SISTEMA WEB PARA DIVULGAÇÃO DE EVENTOS

Trabalho de Conclusão de Curso de graduação, apresentado à disciplina de Trabalho de Conclusão de Curso 2, do Curso de Tecnologia em Análise e Desenvolvimento de Sistemas, da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco, como requisito parcial para obtenção do título de Tecnólogo.

Orientadora: Profa. Beatriz Terezinha Borsoi

**PATO BRANCO
2017**



Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Pato Branco
Departamento Acadêmico de Informática
Curso de Tecnologia em Análise e Desenvolvimento de
Sistemas



TERMO DE APROVAÇÃO
TRABALHO DE CONCLUSÃO DE CURSO
SISTEMA WEB PARA DIVULGAÇÃO DE EVENTOS

por

AUGUSTO CESAR TONELLO GIACOMINI

Este trabalho de conclusão de curso foi apresentado no dia 09 de junho de 2017, como requisito parcial para obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas, pela Universidade Tecnológica Federal do Paraná. O acadêmico foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Banca examinadora:

Prof^a Dr^a Beatriz Terezinha Borsoi
Orientador

Prof. Me. Andreia Scariot Beulke

Prof^a MSc. Vinicius Pegorini

Prof. Dr. Edilson Pontarolo
Coordenador do Curso de Tecnologia em
Análise e Desenvolvimento de Sistemas

Prof^a Dr^a Beatriz Terezinha Borsoi
Responsável pela Atividade de Trabalho de
Conclusão de Curso

A Folha de Aprovação assinada encontra-se na Coordenação do Curso.

RESUMO

GIACOMINI, Augusto Cesar Tonello. Sistema web para divulgação de eventos. 2017. 38 f. Monografia (Trabalho de Conclusão de Curso) - Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2017.

A divulgação de eventos pode ocorrer por meios e canais específicos de propaganda, dependendo do tipo e da abrangência do evento. As mídias sociais são bastante efetivas nesse tipo de divulgação, uma vez que elas podem atingir um público alvo bastante específico. Contudo, a existência de um sistema para que as pessoas possam divulgar eventos e os interessados pesquisar por eventos poderia ser bastante efetiva como propaganda. Imagens relacionadas aos eventos seriam postadas e as pessoas teriam a oportunidade de manifestar o interesse em participar ou informar presença no evento. Considerando esse contexto, este trabalho tem por objetivo apresentar um sistema *web* desenvolvido em Asp.Net para divulgação de eventos. Os usuários cadastrados, uma vez autenticados, poderão incluir eventos, acessar dados de eventos cadastrados, filtrando-os por região, tipo do evento, local e data, por exemplo. O sistema permite compartilhar eventos cadastrados na rede social Facebook.

Palavras-chave: Sistema web. Asp.Net. Rich Internet Application.

ABSTRACT

GIACOMINI, Augusto Cesar Tonello. Web application to advertise events. 2017. 38 f. Monografia (Trabalho de Conclusão de Curso) - Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2017.

The events announcement may occur by specific advertising depending on the type and scope of the event. Social media is very effective in this type of divulgation since it can reach a very specific public. Social media is very effective in this type of divulgation since it can reach a very specific target audience. However, the existence of a system for people to be able to advertise events and those interested in events could search by events of interest, since images would be posted and people would express their interest in participating. Considering this context, this work aims to develop a web system in Asp.Net to register events. Registered users, once logged in, will be able to create new events, access data of the events of their interest, filtering by region, type of event, place and date, for example. Through the system will also be possible to share events in the Facebook.

Keywords: Web system. Asp.Net. Rich Internet Application.

LISTA DE FIGURAS

Figura 1 – Arquitetura típica de uma aplicação RIA.....	15
Figura 2 – Padrão MVC	18
Figura 3 - Diagrama do funcionamento do Entity Framework.....	19
Figura 4 - Diagrama de Caso de Uso	22
Figura 5 - Diagrama de Entidade e Relacionamento	23
Figura 6 - Diagrama de Classes	24
Figura 7 – Página inicial	25
Figura 8 – Menu da página inicial	25
Figura 9 – Lista de eventos	26
Figura 10 – Informações do evento	26
Figura 11 – Login do usuário	27
Figura 13 – Cadastro de evento	28
Figura 14 – Confirmados no evento	28

LISTA DE QUADROS

Quadro 1 - Tecnologias e ferramentas utilizadas.....	16
---	-----------

LISTAGENS DE CÓDIGOS

Listagem 1 – _Layout.cshtml	30
Listagem 2 – _Home.cshtml	31
Listagem 3 – Script.js	32
Listagem 4 – Evento.cshtml	33
Listagem 5 – EventoController.cs (Método Index)	34

LISTA DE SIGLAS

ANSI SQL	<i>American National Standards Institute Structured Query Language</i>
Cetic.br	Desenvolvimento da Sociedade da Informação
CGI.br	Comitê Gestor da Internet no Brasil
CLR	<i>Common Language Runtime</i>
HTML	<i>HyperText Markup Language</i>
IBGE	Instituto Brasileiro de Geografia e Estatística
JSON	<i>JavaScript Object Notation</i>
MVC	<i>Model-View-Controller</i>
NIC.br	Núcleo de Informação e Coordenação do Ponto BR
ORM	<i>Object Relational Management</i>
PNAD	Pesquisa Nacional por Amostra de Domicílio
RIA	<i>Rich Internet Application</i>
SGBD	Sistema Gerenciador de Banco de Dados
TIC	Tecnologias de Informação e Comunicação
T-SQL	<i>Transact-SQL</i>
UML	<i>Unified Modeling Language</i>
XML	<i>eXtensible Markup Language</i>

SUMÁRIO

1 INTRODUÇÃO	10
1.1 CONSIDERAÇÕES INICIAIS	10
1.2 OBJETIVOS	11
1.2.1 Objetivo Geral	11
1.2.2 Objetivos Específicos	12
1.3 JUSTIFICATIVA	12
1.4 ESTRUTURA DO TRABALHO.....	13
2 RICH INTERNET APPLICATION	14
3 MATERIAIS E MÉTODO	16
3.1 MATERIAIS.....	16
3.1.1 Visual Studio 2017 Community	16
3.1.2 ASP.NET	17
3.1.3 C#	17
3.1.4 Padrão Arquitetônico MVC	18
3.1.5 Entity Framework	19
3.1.6 SQL Server Express	20
3.1.7 Astah Community	20
3.2 MÉTODO.....	21
4 RESULTADOS	22
4.1 ESCOPO DO SISTEMA.....	22
4.2 MODELAGEM DO SISTEMA	22
4.3 APRESENTAÇÃO DO SISTEMA.....	24
4.4 IMPLEMENTAÇÃO DO SISTEMA	29
5 CONCLUSÃO	36
REFERÊNCIAS	37

1 INTRODUÇÃO

Neste capítulo são apresentadas as considerações iniciais do trabalho, os seus objetivos, a justificativa e a estrutura do texto que ocorre por meio da apresentação dos capítulos subsequentes.

1.1 CONSIDERAÇÕES INICIAIS

A evolução das Tecnologias de Informação e Comunicação (TIC), percebida também pelo aumento do número de dispositivos com acesso à Internet e do número de usuários dessas tecnologias, vem causando diversas mudanças em vários setores da sociedade, seja em aspectos de negócio ou pessoais. Uma dessas mudanças é a disponibilização de uma quantidade cada vez maior de informações por meio da Internet, da facilidade de interação com pessoas, além das possibilidades de comércio em seus mais variados tipos. As redes sociais vêm desempenhando um papel bastante evidente nos novos usos das TICs. Além da disponibilização de informações, as redes sociais têm encontrado amplo espaço de expansão como ferramenta de interação entre as pessoas e para *marketing* pelas empresas e instituições.

A facilidade de acesso às informações faz com que as pessoas procurem cada vez mais por novas tecnologias que as auxiliem e as mantenham informadas sobre o que está acontecendo, em termos de eventos, notícias e outros, na região de convivência e no mundo. Segundo o Instituto Brasileiro de Geografia e Estatística (INSTITUTO..., 2013), por meio da Pesquisa Nacional por Amostra de Domicílio (PNAD), de 2005 para 2011, o número de pessoas com acesso à internet cresceu 143,8% e de pessoas com celular, 107,2%. Esses dados confirmam que as mídias digitais com acesso à Internet alcançam uma parcela muito significativa da população, podendo ser utilizadas para atividades que requeiram abrangência ou divulgação em massa.

Dados do IBGE são sustentados pela 11^a edição da pesquisa TIC Domicílios 2015, que mede a posse, o uso, o acesso e os hábitos da população brasileira em relação às tecnologias de informação e de comunicação, apresenta que em 2015,

58% da população brasileira usava a Internet. Esse percentual representa 102 milhões de internautas (BOCCHINI, 2016).

Estudo realizado pelo Comitê Gestor da Internet no Brasil (CGI.br), pelo Centro Regional de Estudos para o Desenvolvimento da Sociedade da Informação (Cetic.br), e pelo Núcleo de Informação e Coordenação do Ponto BR (NIC.br), divulgados em 2016, mostra que o telefone celular é o dispositivo utilizado para o acesso individual da Internet pela maioria dos usuários, 89%, seguido pelo computador de mesa (40%), computador portátil ou *notebook* (39%), *tablet* (19%), televisão (13%) e videogame (8%) (BOCCHINI, 2016).

Os dados de uso de dispositivos móveis, enfatizando telefones celulares e *tablets*, para acesso à Internet ressaltam a importância do desenvolvimento de páginas e de sistemas *web* responsivos. O leiaute dessas páginas e de sistemas ou sistemas se ajusta conforme o tamanho da tela do hardware (meio ou dispositivo) pelo qual eles estão sendo acessados.

Considerando a expansão de uso da Internet, das redes sociais e de dispositivos móveis, por meio da realização deste trabalho foi desenvolvido um sistema *web* para que usuários possam consultar eventos que serão realizados em sua região, por exemplo, e cadastrar novos eventos para divulgação. O sistema permite compartilhar eventos cadastrados por meio do Facebook.

1.2 OBJETIVOS

O objetivo geral se refere ao resultado principal da realização deste trabalho, ou seja, o desenvolvimento de um sistema *web*. Os objetivos específicos complementam o objetivo geral, especificando as finalidades do sistema desenvolvido.

1.2.1 Objetivo Geral

Implementar um sistema *web* responsivo para divulgação de eventos.

1.2.2 Objetivos Específicos

Os objetivos específicos do trabalho são:

- Possibilitar o cadastro e a divulgação de eventos, a exemplo dos sociais, de entretenimento e culturais, por meio de um sistema *web* com interface responsiva e com compartilhamento para divulgação por meio do Facebook;
- Facilitar a busca por informações sobre eventos, como data e local de ocorrência;
- Auxiliar na divulgação e na localização de eventos;
- Oferecer informação aos usuários do sistema sobre eventos que irão ocorrer, facilitando a identificação de eventos próximos de sua localização e/ou de seu interesse.

1.3 JUSTIFICATIVA

A localização de eventos sociais, como *shows*, agendados é uma atividade que pode ser facilitada pelo uso de redes sociais. O Facebook possui um recurso para divulgar eventos, mas esse recurso apresenta algumas desvantagens, como, por exemplo, a de não disponibilizar uma maneira de pesquisar por eventos filtrando por região ou pelo tipo de evento.

Visando fornecer uma ferramenta para divulgação e localização de eventos, como resultado da realização deste trabalho foi desenvolvido um sistema que visa facilitar a publicação de eventos com a opção de compartilhamento da divulgação na rede social Facebook. O sistema visa facilitar a busca por eventos por meio da opção de filtros como data e local de realização do evento.

O sistema desenvolvido é para *web*, foi implementado com as tecnologias C# e Asp.Net e possui interface responsiva, facilitando, assim, o acesso e uso em diferentes dispositivos como computadores, *tablets* e *smartphones*.

1.4 ESTRUTURA DO TRABALHO

O presente trabalho é dividido em capítulos, sendo este o Capítulo 1 que contém as considerações iniciais, os objetivos e a justificativa do trabalho. O Capítulo 2 contém o referencial teórico que se refere às aplicações *web* denominadas de interface rica. No Capítulo 3 são apresentadas as ferramentas, as tecnologias e o método utilizados para a análise e o desenvolvimento do sistema. No Capítulo 4 está o resultado obtido com a realização do trabalho que é representado pelos diagramas que foram desenvolvidos e pelo sistema implementado. Esse é apresentado por meio de exemplos de telas e de partes de código. Por fim, está o Capítulo 5 com as considerações finais que são seguidas das referências utilizadas na composição do texto.

2 RICH INTERNET APPLICATION

Tem sido observado um rápido crescimento das aplicações *web* com comportamento sofisticado em termos de interface com o usuário. A sofisticação está relacionada à quantidade e à diversidade dos componentes utilizados para compor essas interfaces e das funcionalidades em termos de interação que elas oferecem, assemelhando-se às aplicações *desktop*. A esse comportamento rico, se comparado às aplicações *web* tradicionais como interface desenvolvida em *HyperText Markup Language* (HTML) baseada em *tags* simples, está associada à redução de tráfego de rede em decorrência da minimização de dados transmitidos entre servidor e cliente e vice-versa (pela atualização apenas de partes da página que sofreram alterações pela interação do usuário) e pela possibilidade de armazenamento e de processamento de dados no cliente.

Como ressaltam Roubi, Erramdani e Mbarki (2015), essas aplicações chamadas *Rich Internet Application* (RIA) juntaram a riqueza de interface das aplicações *desktop* e a navegação em páginas *web*, ou seja, a combinação da simplicidade do paradigma de hipertexto com a flexibilidade da interface *desktop*.

O termo RIA foi proposto em 2002 em um artigo da Macromedia (JEREMY, 2002). Embora a definição de Roubi, Erramdani e Mbarki (2015) seja bem aceita, Casteleyn, Garrigós e Mazón (2014) ressaltam que mais de uma década após a sua proposição ainda não há uma definição padronizada e amplamente aceita para o termo RIA. Para esses autores, as RIAs são qualitativamente caracterizadas como aplicações *web* que visam prover as características e as funcionalidades das aplicações *desktop* tradicionais. As RIAs almejam responsividade, capacidade de interação e interface com o usuário melhoradas para oferecer uma experiência de uso caracterizada como rica. Elas tipicamente dependem massivamente de tecnologias no lado cliente que são combinadas com comunicação assíncrona entre cliente e servidor para alcançar esses objetivos.

Centrado na forma de processamento, Martínez-Ruiz et al. (2006) definiram RIA como uma aplicação *web* que transfere a maior parte da carga de processamento da interface do usuário para o cliente *web*, enquanto a parte predominante dos dados envolvida no controle e na manutenção dos dados de negócio permanece no servidor da aplicação. Esses autores propõem uma

arquitetura padrão para as RIAs, representada na Figura 1, baseada na divisão do processamento de dados entre cliente e servidor.

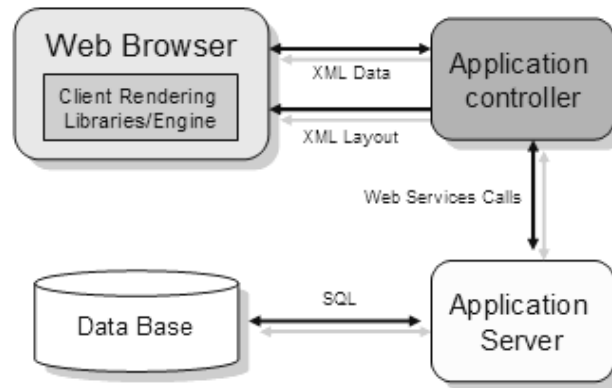


Figura 1 – Arquitetura típica de uma aplicação RIA
Fonte: Martínez-Ruiz (2010, p. 345).

O formato de intercâmbio (transferência) de dados entre cliente e servidor como consta na Figura 1 é o *eXtensible Markup Language* (XML). Ressalta-se que o *JavaScript Object Notation* (JSON) é outro formato que vem sendo amplamente utilizado para esse tipo de tarefa.

3 MATERIAIS E MÉTODO

Este capítulo apresenta as ferramentas e as tecnologias utilizadas para a modelagem e a implementação do sistema desenvolvido como resultado da realização deste trabalho.

3.1 MATERIAIS

O Quadro 1 apresenta as ferramentas, as tecnologias e o padrão conceitual que foram utilizados para modelar e implementar o sistema desenvolvido.

Ferramenta	Versão	Link	Funcionalidade
Visual Studio Community	2017	https://www.visualstudio.com/pt-br/downloads/	<i>Integrated Development Environment</i> (IDE) para desenvolvimento do sistema.
ASP.Net		https://www.asp.net/	<i>Framework</i> de desenvolvimento <i>web</i> .
C#		https://docs.microsoft.com/pt-br/dotnet/articles/csharp/	Linguagem de programação.
MVC		https://www.asp.net/mvc	Padrão arquitetônico para estruturação do código em camadas.
Entity Framework	6.1.3	https://www.nuget.org/packages/EntityFramework	Ferramenta de mapeamento objeto relacional.
SQL Server Express	2016	https://www.microsoft.com/pt-br/sql-server/sql-server-downloads	Banco de dados para armazenar dados no servidor.
Astah Community	7.0.0	http://astah.net/download	Ferramenta para desenvolver os diagramas.

Quadro 1 - Tecnologias e ferramentas utilizadas

A seguir são apresentadas as ferramentas e as tecnologias constantes no Quadro 1. MVC poderia ser mais adequadamente caracterizado como padrão, modelo ou conceito, mas também foi incluído nesse quadro por ser utilizado como modelo conceitual para a organização da codificação da aplicação.

3.1.1 Visual Studio 2017 Community

Visual Studio Community é um ambiente de desenvolvimento, IDE, gratuita e extensível para a criação de sistemas para Android, IOS e Windows, além de sistemas *web* e serviços de nuvem. Os recursos do Xamarin integrados ao Visual

Studio possibilitam o desenvolvimento integrado de sistemas móveis (VISUAL STUDIO, 2017).

Esse ambiente possibilita o desenvolvimento em várias linguagens de programação para *web* como: ASP.NET, Node.js, Python e JavaScript. Além de *frameworks* e estruturas como AngularJS, jQuery, Bootstrap, Django e Backbone.js. O suporte multilíngue abrange C#, Visual Basic, F#, C++, JavaScript, TypeScript, Python e outras (VISUAL STUDIO, 2017).

O Visual Studio também possibilita o gerenciamento do código-fonte em repositórios Git hospedados por qualquer provedor, incluindo o GitHub. A ferramenta Visual Studio Team Services é utilizada para gerenciar o código, os erros (*bugs*) e itens de trabalho do projeto.

3.1.2 ASP.NET

ASP.NET é um *framework web* de código fonte aberto utilizado na construção de sistemas *web* e serviços com .NET e ASP.NET criando sites *web* baseados em HTML5, CSS e JavaScript de maneira simples e escalável (ASP.NET, 2017).

ASP.NET é baseado no .NET Framework. Os sistemas podem ser escritos em qualquer linguagem compatível com o *Common Language Runtime* (CLR), incluindo Visual Basic e C#. No desenvolvimento do projeto que é objeto deste trabalho foi utilizada a linguagem C#.

O Visual Studio 2017 Community utilizado no desenvolvimento deste projeto inclui o conjunto básico de recursos de *design* da *web*.

3.1.3 C#

C# (pronunciado "C *sharp*") é uma linguagem de programação que pode ser utilizada para criar uma variedade de sistemas executados no .NET Framework. C# é fortemente tipada e orientada a objeto. C# permite o desenvolvimento rápido de sistemas mantendo a expressividade e a elegância das linguagens de estilo C (MICROSOFT, 2017).

3.1.4 Padrão Arquitetônico MVC

Model-View-Controller (MVC) separa um aplicativo ou sistema em três componentes principais: modelo (*model*), exibição (*view*) e controlador (*controller*), como apresentado na Figura 2. A estrutura ASP.NET MVC oferece uma alternativa ao padrão *web forms* do ASP.NET para criar aplicativos *web*. A estrutura ASP.NET MVC é uma estrutura de apresentação altamente testável que é integrada aos recursos ASP.NET existentes, como páginas mestre e autenticação baseada em associação (MICROSOFT, 2017b).

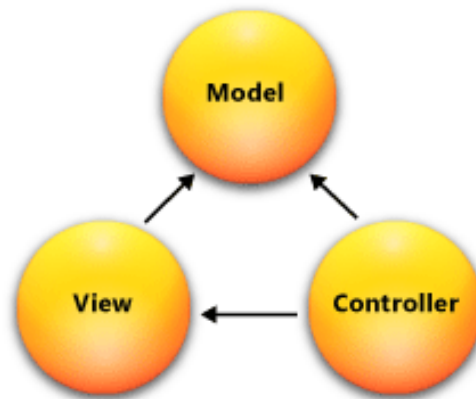


Figura 2 – Padrão MVC
Fonte: Microsoft (2017b, p.1).

A estrutura MVC inclui os seguintes componentes (MICROSOFT, 2017b):

a) **Modelos (*model*)** - os objetos de modelo são as partes do aplicativo que implementam a lógica para o domínio de dados do aplicativo. Muitas vezes, os objetos de modelo recuperam e armazenam o estado do modelo em um banco de dados.

b) **Exibições (*view*)** - as exibições são os componentes que apresentam a interface do usuário do aplicativo. Essa interface é, normalmente, criada a partir dos dados do modelo.

c) **Controladores (*controller*)** - os controladores são os componentes que lidam com a interação do usuário, trabalham com o modelo e selecionam uma exibição de renderização que é apresentada na interface. Em um aplicativo MVC, a exibição só apresenta informações e o controlador manipula e responde à entrada e à interação do usuário.

3.1.5 Entity Framework

O Microsoft Entity Framework é uma ferramenta de mapeamento objeto relacional - *Object Relational Management* (ORM) - que permite trabalhar com classes (entidades) que correspondem a tabelas em um banco de dados, tornando transparente o acesso a estes dados e, principalmente, eliminando a necessidade de escrever código para operações de banco de dados (*select*, *insert*, *update*, *delete*) na aplicação. Com o Entity Framework os dados são manipulados por meio de classes que são mapeadas com as tabelas do banco de dados, sendo assim, o acesso e a manipulação desses dados fica mais simples, pois o desenvolvedor já trabalha naturalmente com objetos, propriedades e coleções (SANTOS, 2012).

A Figura 3 apresenta uma visão geral do funcionamento do Entity Framework.

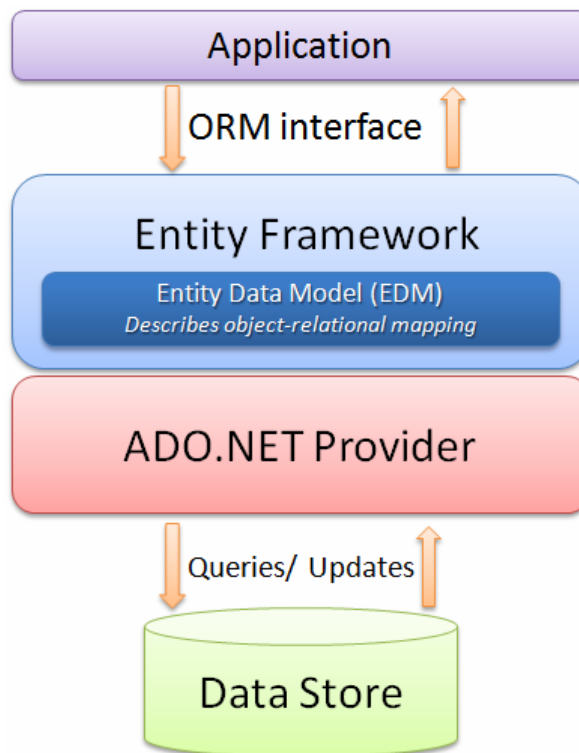


Figura 3 - Diagrama do funcionamento do Entity Framework
Fonte: Santos (2012, p. 1).

A comunicação do Entity Framework com o banco de dados é feita por meio do ADO.Net Provider, que funciona como um *driver* do banco de dados, normalmente desenvolvido pelo próprio fabricante do banco ou em alguns casos por um terceiro. Sendo assim, todos os comandos submetidos pelo Entity Framework

são traduzidos para a linguagem do banco de dados por meio do seu *provider*, gerando os comandos SQL mais adequados a cada operação e, principalmente, comandos que tenham o máximo de desempenho (SANTOS, 2012).

3.1.6 SQL Server Express

O Microsoft SQL Server é um Sistema Gerenciador de Banco de Dados (SGBD) relacional desenvolvido pela Microsoft. O Microsoft SQL Server possui edições diferentes que são destinadas para públicos e cargas de trabalho distintas (variando de pequenas aplicações que armazenam e recuperam dados no mesmo computador, a milhões de usuários e computadores que acessam grandes quantidades de dados a partir da Internet). As suas linguagens de consulta primárias são *Transact-SQL (T-SQL)* e *American National Standards Institute Structured Query Language (ANSI SQL)* (DATABASE DESIGN RESOURCE, 2017).

O Microsoft SQL Server fornece suporte a dados estruturados e não estruturados armazenando dados corporativos com suporte nativo para dados relacionais, XML e dados espaciais. Esse SGBD também possibilita adicionar informações geográficas aos aplicativos de negócios e criar aplicativos com geolocalização. A granularidade dos dados temporais é aumentada com dados do tipo *date* e *time* (DATABASE DESIGN RESOURCE, 2017).

3.1.7 Astah Community

Astah Community é uma ferramenta para modelagem de sistemas com suporte para a *Unified Modeling Language 2 (UML)* (ASTAH, 2017). Além da modelagem de diagramas da UML, a ferramenta permite a impressão e a exportação das imagens dos diagramas.

O Astah Community é uma ferramenta gratuita para o usuário final, possuindo uma versão paga com mais recursos, como a criação de diagramas a partir de códigos das linguagens Java, C e C++ e também a exportação de código a partir dos diagramas para essas linguagens (ASTAH, 2017).

3.2 MÉTODO

A seguir são apresentadas as etapas de realização do trabalho como levantamento de requisitos, análise e implementação:

a) Levantamento de requisitos – Para definir o problema foi realizada análise de sistemas semelhantes e pesquisa em artigos *online*. Essa análise de sistemas semelhantes auxiliou a definir os requisitos pretendidos e as funcionalidades básicas do sistema. Os requisitos foram organizados em casos de uso e modelados na forma de um diagrama.

b) Análise e projeto – Para realizar a análise foram utilizados conceitos e a metodologia do paradigma da orientação a objetos. Com base nos requisitos do sistema, organizados como casos de uso, foram elaborados os diagramas de entidades e relacionamentos do banco de dados e o diagrama de classes.

c) Implementação – A codificação do sistema foi realizada utilizando tecnologias e conceitos constantes no Quadro 1. Não foram realizados testes formais ou baseados em planos de testes. Os testes realizados tiveram o objetivo de identificar erros de codificação e verificação de atendimento às funcionalidades pretendidas e definidas para o sistema. Os testes foram realizados paralelamente à implementação.

4 RESULTADOS

Com resultado deste trabalho foi realizada a modelagem e a implementação de um sistema para divulgação de eventos.

4.1 ESCOPO DO SISTEMA

O sistema possibilitará o cadastro de eventos que ficarão disponíveis para consulta e a busca de eventos, por meio da utilização de filtros como locais e datas. O sistema permitirá a inclusão de imagens e de comentários relacionados a um evento cadastrado.

Eventos cadastrados poderão ser compartilhados por meio do Facebook e o usuário poderá deixar registrado o seu interesse em participar de um referido evento ou indicar a confirmação de participação em eventos cadastrados.

4.2 MODELAGEM DO SISTEMA

A Figura 4 representa o diagrama de caso de uso, contendo as principais funcionalidades do sistema.

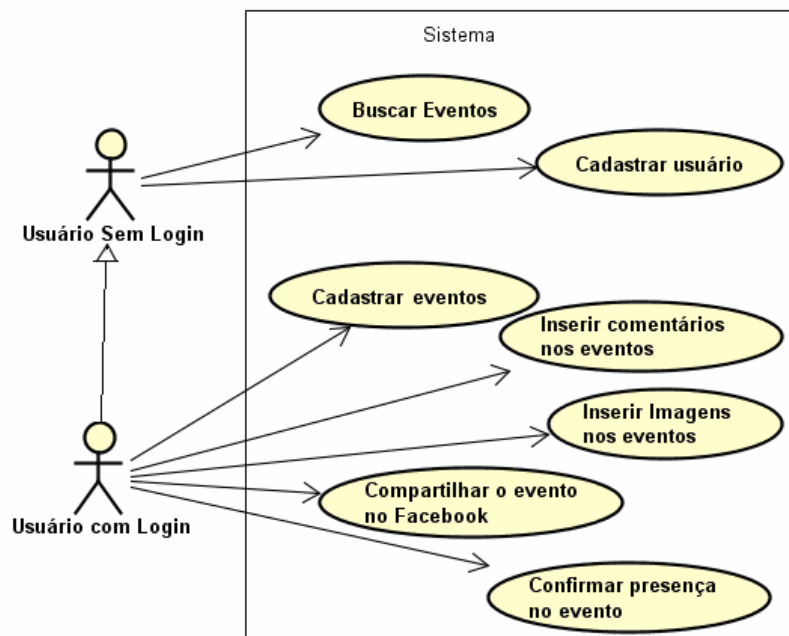


Figura 4 - Diagrama de Caso de Uso

No diagrama de caso de uso, apresentado na Figura 4, estão as principais funcionalidades que o usuário poderá executar no sistema, como cadastrar, buscar e compartilhar eventos. O sistema terá usuários com dois perfis distintos ou permissões de acesso: usuário com login e usuário sem login. O usuário com login herda as permissões do usuário sem login que são busca evento e cadastrar usuário. Essa última só faz sentido se o usuário ainda não possui cadastro. E é o próprio usuário quem faz o seu cadastro no sistema. Uma vez autenticado, o usuário terá acesso a todas as funcionalidades oferecidas pelo sistema.

A Figura 5 apresenta o diagrama de entidades e relacionamentos contendo as tabelas criadas no banco de dados do sistema. A tabela “USUARIO” armazenará os dados dos usuários do sistema. Em “EVENTO” estarão os dados dos eventos cadastrados no site. A tabela “IMAGEM” armazenará todas as imagens relacionadas aos eventos cadastrados. Um evento poderá ter diversas imagens relacionadas. A tabela “CIDADE” conterà o cadastro de cidades utilizadas no cadastro de eventos. E “COMENTARIOEVENTO” armazenará todos os comentários dos eventos e o usuário que cadastrou.

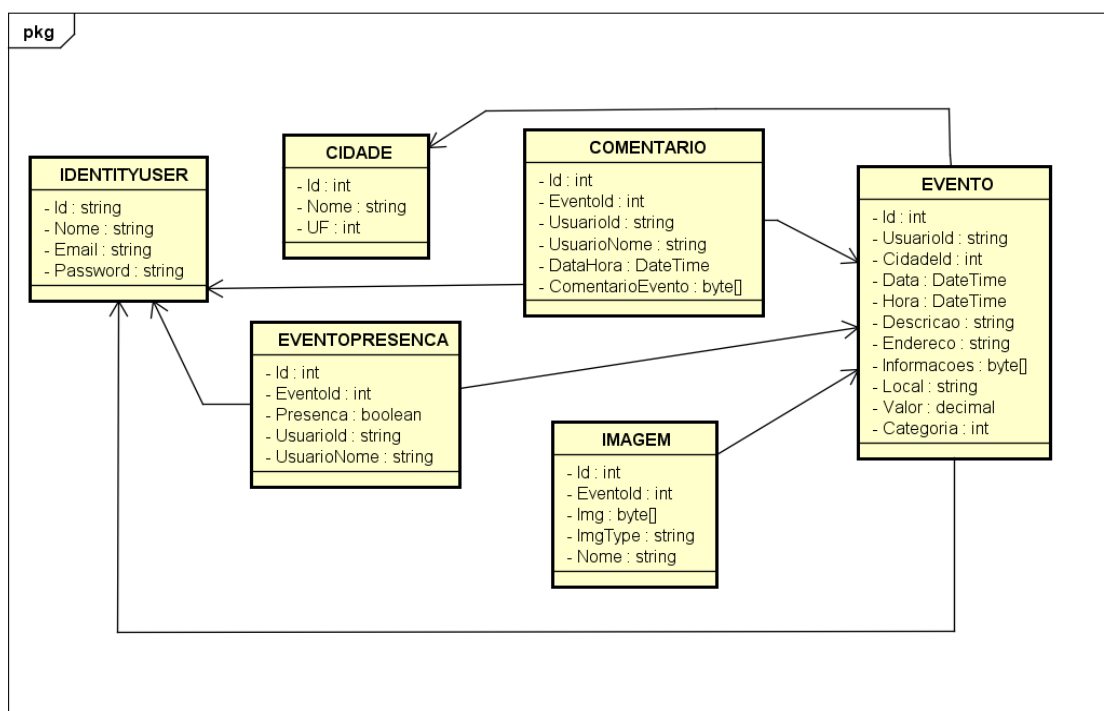


Figura 5 - Diagrama de Entidade e Relacionamento

Na Figura 6 é apresentado o diagrama de classes. Nesse diagrama, a classe de usuários (*ApplicationUser*) que herda atributos da classe *IdentityUser*, padrão do ASP.NET *Identity*, que é um sistema de associação que permite adicionar funcionalidades de *login* ao aplicativo, armazenando as informações da conta do usuário para validar suas credenciais no sistema. A classe de eventos armazenará dados do evento, como data, hora, descrição, endereço, local, valor, informações, categoria que é uma classe do tipo Enum, uma referência para o usuário que criou o evento, outra para a cidade. As demais classes, *EventoPresenca*, *Comentario* e *Imagem* estão relacionadas com o *Evento*.

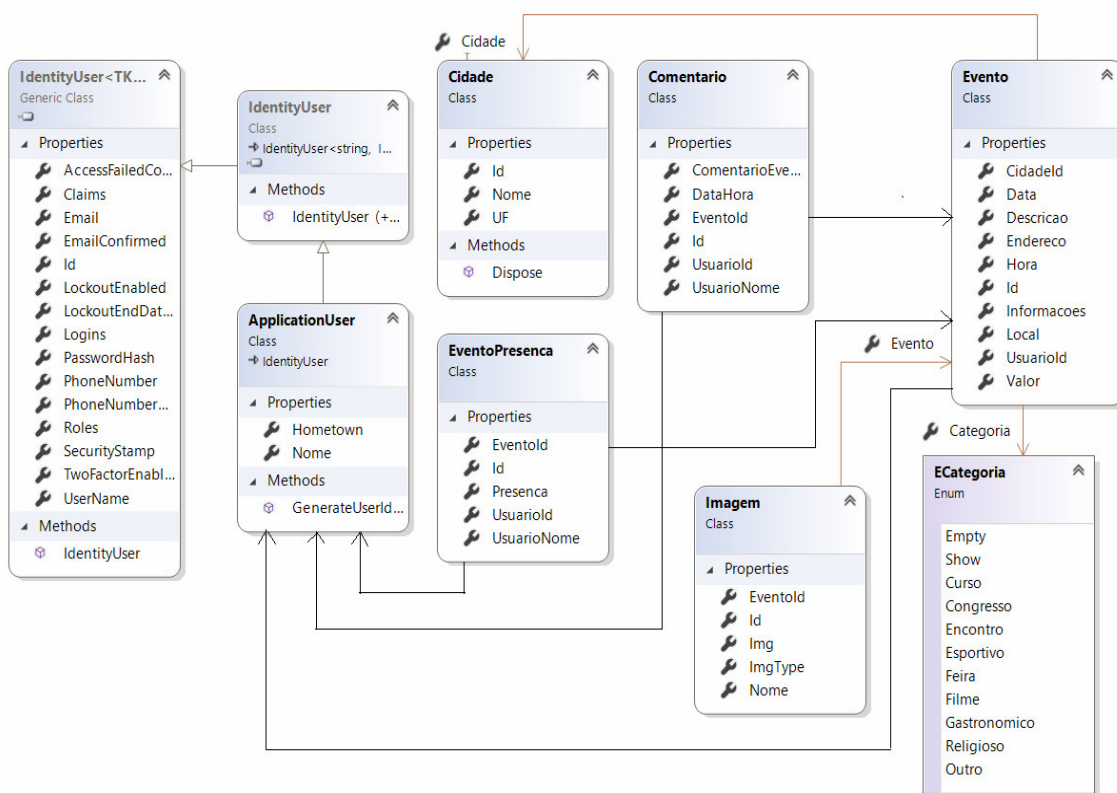


Figura 6 - Diagrama de Classes

4.3 APRESENTAÇÃO DO SISTEMA

O leiaute do sistema é composto por duas partes: um menu superior fixo, que contém a logo do site e os itens de navegação, e a parte central que contém o conteúdo da página que está sendo navegada.

A Figura 7 apresenta as duas partes nas quais as páginas foram organizadas. Na parte central, da Figura 7, está sendo exibida a página inicial do sistema.

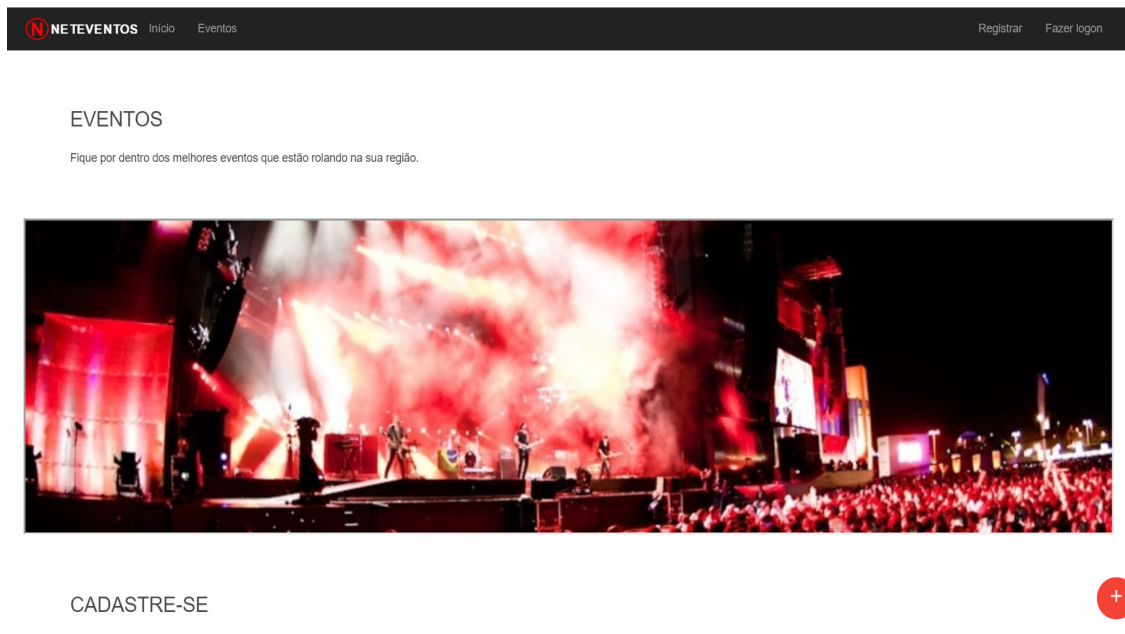


Figura 7 – Página inicial

Na página inicial existe também um menu, no lado inferior direito, que permite navegar dentro da própria página, como é mostrado na Figura 8.

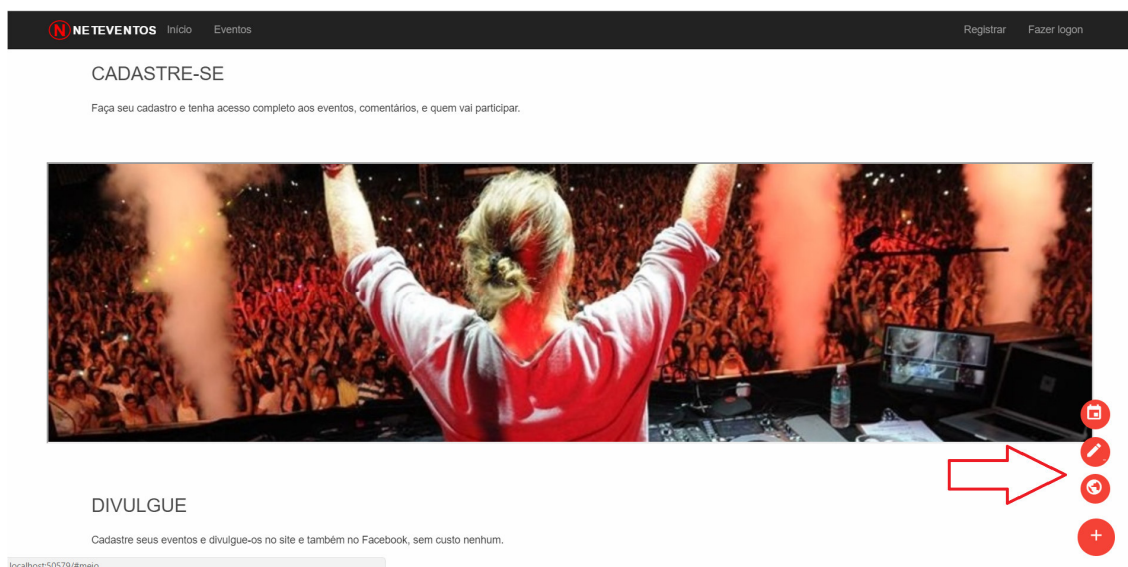


Figura 8 – Menu da página inicial

Clicando no item “Eventos” do menu superior é exibida uma lista de eventos com um menu de filtros no lado esquerdo, possibilitando a busca por descrição,

cidade, categoria e período ou data de realização do evento categoria e período ou data de realização do evento, como pode ser observado na Figura 9.

Buscar por evento

Cidade: PR Pato Branco

Categoria:

Período de: 01/05/2017 até: 12/07/2017

Evento	Data	Hora	Local	Categoria	Cidade
Festa Junina	17/06/2017	14:00	Praça	Show, música e festa	Pato Branco
Pato x Marreco	11/06/2017	19:00	Ginásio Arrudão	Esportivo	Pato Branco
Show ACDC	23/05/2017	00:00	Pinheiros	Show, música e festa	Pato Branco

Página 1 de 5

© 2017 - Net Eventos

Figura 9 – Lista de eventos

Quando selecionado um evento na lista são exibidas informações mais detalhadas sobre o evento, como local, data, valor de ingresso, número de confirmados e de interessados, imagens, comentários e, também a opção de compartilhar no Facebook, conforme apresentado na Figura 10.

testando

Informações

Local - teste
rua x n123 - Pato Branco - PR

Data - terça-feira, 23 de maio de 2017, 20:30h

Valor da entrada - R\$ 44.00

Categoria - Show, música e festa

Compartilhar com seus amigos - Compartilhar 2

9 Confirmados - 1 Interessados

Imagens

Comentários

Usuário	Comentário
augusto	ahaaa
01/01/1900 00:00	soakso soakso soakso
01/01/1900 00:00	aaaaaaaaaaaaaaaaaaaa

Figura 10 – Informações do evento

O usuário pode fazer seu cadastro informando seu nome, *email* e senha e efetuar o *login* no sistema, utilizando os itens do menu superior “Registrar” e “Fazer Logon”. Na Figura 11 é exibida a página de *login*.

Figura 11 – Login do usuário

Depois que o usuário efetuar seu *login* é habilitado um novo item no menu superior, chamado “Meus Eventos”. Ao ser selecionado, esse menu lista todos os eventos que o próprio usuário cadastrou e são apresentadas opções para cadastrar novos eventos e alterar e excluir eventos já cadastrados, como mostra a Figura 12.

Evento	Data	Hora	Local	Categoria	Cidade
Show ACDC	23/05/2017	00:00	Pinheiros	Show, música e festa	Pato Branco

Figura 12 – Meus eventos

Quando selecionada a opção “Criar novo evento” é exibido um formulário com os campos referentes ao cadastro de evento e as validações de obrigatoriedade de

preenchimento de determinados campos. A Figura 13 apresenta a tela de cadastro de evento.

The screenshot shows the 'Novo evento' registration page. At the top, there is a navigation bar with 'NETEVENOTOS', 'Início', 'Eventos', and 'Meus Eventos'. The main heading is 'Novo evento' with a sub-heading 'Evento'. The form is divided into several sections:

- Form Fields:**
 - Descrição:** A text input field with a red error message below it: 'O campo descrição é obrigatório'.
 - Categoria:** A dropdown menu currently showing 'Show, música e festa'.
 - Local:** A text input field with a red error message below it: 'O campo local é obrigatório'.
 - Valor:** A text input field.
 - Endereço:** A text input field.
 - Cidade:** Two dropdown menus, the first showing 'PF' and the second showing 'Pato Branco'.
 - Data:** A date picker showing '23/05/2017'.
 - Hora:** A time picker showing '00:00'.
- Informações:** A large empty rectangular box on the right side of the form.
- Imagens:** A section with a 'Nome' input field, an 'Imagem' section with an 'Escolher arquivo' button and the text 'Nenhum arq... lecionado', and an 'Adicionar' button.
- Buttons:** A 'Salvar' button is located at the bottom right of the form area. A 'Voltar' link is at the bottom left.

Figura 13 – Cadastro de evento

Além das informações apresentadas na visualização dos dados de um evento cadastrado, quando o usuário está logado no sistema, são habilitadas as opções de confirmar presença e visualizar as pessoas que já confirmaram presença e, também, incluir comentários nos eventos (Figura 14).

The screenshot shows the event details page for 'Show ACDC'. A modal window titled 'Confirmados' is open, displaying a list of confirmed attendees:

- augusto
- raquel

Below the list, it shows 'Total de confirmados - 1'. A 'Fechar' button is located at the bottom right of the modal. The background shows the event details page with a sidebar on the left containing various event information and a 'Confirmar presença' button.

Figura 14 – Confirmados no evento

4.4 IMPLEMENTAÇÃO DO SISTEMA

Como apresentado na Seção 4.3, o leiaute do sistema é composto por um menu superior fixo que contém a logo do site e os itens de navegação e na parte central está o conteúdo da página que está sendo navegada.

Para facilitar o desenvolvimento foi criada uma página principal, chamada “_Layout.cshtml”, como pode ser observado na Listagem 1, contendo a estrutura principal do HTML. Nessa página foi declarada a referência para o CSS, o JQuery e o Bootstrap na *tag header*, que será utilizada em todo o sistema. Na *tag body* foi criado o menu que estará visível em todas as páginas e uma *tag div* que será utilizada para renderizar todas as outras páginas nesta seção.

```
<!DOCTYPE html>
<html lang="pt">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <meta charset="utf-8" />
  <title>@ViewBag.Title - NETEVENTOS</title>
  <link href="~/favicon.ico" rel="shortcut icon" type="image/x-icon" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet"
href="https://fonts.googleapis.com/icon?family=Material+Icons">

  <meta property="og:url" name="metaUrl">
  <meta property="og:type" content="website">
  <meta property="og:title" content="NETEVENTOS">
  <meta property="og:description" content="descrição teste">
  <meta property="og:image" content="~/Image/logo.svg" />

  @Styles.Render("~/Content/css")
  @Scripts.Render("~/bundles/modernizr")
  @Scripts.Render("~/bundles/jquery")
  @Scripts.Render("~/bundles/bootstrap")
  @RenderSection("Scripts", required: false)
</head>
<body>
  <div class="navbar navbar-inverse navbar-fixed-top">
    <div class="container" style="max-width: 95%">
      <div class="navbar-header">
        <button type="button" class="navbar-toggle" data-toggle="collapse"
data-target=".navbar-collapse">
          <span class="icon-bar"></span>
          <span class="icon-bar"></span>
          <span class="icon-bar"></span>
        </button>
        <a href='@Url.Action("Index", "Home")'>
          <img src='@Url.Content("~/Image/logo.svg")' style="height: 48px;
width: 147px; padding-top: 10px; padding-bottom: 8px;" />
        </a>
      </div>
      <div class="navbar-collapse collapse">
        <ul class="nav navbar-nav">
          <li>@Html.ActionLink("Início", "Index", "Home", new { area = "" },
null)</li>
```

```

    }, null)</li>
        <li>@Html.ActionLink("Eventos", "Index", "Evento", new { area = ""
            @if (Request.IsAuthenticated)
            {
                <li>@Html.ActionLink("Meus Eventos", "Index", "MeusEventos",
new { area = "" }, null)</li>
            }
        </ul>
        @Html.Partial("_LoginPartial")
    </div>
</div>
<div class="container body-content" style="max-width: 95%">
    @RenderBody()
    @RenderSection("SPAViews", required: false)
    <hr />
    <footer>
        <p>&copy; @DateTime.Now.Year - Net Eventos</p>
    </footer>
</div>
</body>
</html>

```

Listagem 1 – _Layout.cshtml

A página inicial do sistema foi desenvolvida na “_Home.cshtml” e é apresentada na Listagem 2. Nessa página foram criadas *sections* que contém textos e imagens que causam um efeito 3D, quando é rolada a barra de rolagem da página. Isso ocorre porque as imagens rolam em velocidade menor que a velocidade de rolagem da página. Também foi desenvolvido um menu inferior no lado esquerdo, quando o usuário passar o *mouse* sobre esse menu são apresentadas opções de navegação dentro da própria página.

```

<main>
    <nav>
        <div class="menu"><i class="material-icons">&#xE145;</i></div>
        <div class="nav-itens">
            <a href="#topo" class="tooltip2" data-tooltip="Topo">
                <i class="material-icons">&#xE878;</i>
                <label class="text-tooltip2"></label>
            </a>
            <a href="#meio" class="tooltip2" data-tooltip="Meio">
                <i class="material-icons">&#xE150;</i>
                <label class="text-tooltip2"></label>
            </a>
            <a href="#fim" class="tooltip2" data-tooltip="Fim">
                <i class="material-icons">&#xE80B;</i>
                <label class="text-tooltip2"></label>
            </a>
        </div>
    </nav>

    <section>
        <div id="topo" class="section_info">
            <div class="row container">
                <h2 class="header">Eventos</h2>
                <p>Fique por dentro dos melhores eventos que estão rolando na sua

```

```

região.</p>
    </div>
  </div>

  <div class="parallax-container" style="border-style: inset;">
    <div class="parallax"></div>
  </div>
</section>

<section>
  <div id="meio" class="section_info">
    <div class="row container">
      <h2 class="header">Cadastre-se</h2>
      <p>Faça seu cadastro e tenha acesso completo aos eventos, comentários,
e quem vai participar.</p>
    </div>
  </div>

  <div class="parallax-container" style="border-style: inset;">
    <div class="parallax"></div>
  </div>
</section>

<section>
  <div id="fim" class="section_info">
    <div class="row container">
      <h2 class="header">Divulgue</h2>
      <p>Cadastre seus eventos e divulgue-os no site e também no Facebook,
sem custo nenhum.</p>
    </div>
  </div>

  <div class="parallax-container" style="border-style: inset;">
    <div class="parallax"></div>
  </div>
</section>
</main>

```

Listagem 2 – _Home.cshtml

Para criar os efeitos na página “_Home.cshml” foi implementado um arquivo JavaScript, “script.js”, apresentado na Listagem 3, que utiliza os seletores do JQuery para buscar os elementos na página e vincular as funções desejadas para eles. A função responsável pelos efeitos nas imagens é a *parallax*. Essa função é declarada dentro do arquivo “parallax.js” e é disponibilizada pela *Material Design* (linguagem visual da *Google*).

```

$(document).ready(function () {

  $(''.parallax').parallax();

  var page = $('html, body');
  $('a.tooltip2').click(function () {
    page.animate({
      scrollTop: $($.attr(this, 'href')).offset().top
    }, 1000);
    return false;
  });
});

```



```

$("nav").mouseover(function () {
    $(".nav-itens").addClass("nav-show");
});
$("nav").mouseout(function () {
    $(".nav-itens").removeClass("nav-show");
});
});

```

Listagem 3 – Script.js

Na Listagem 4 está a página “Evento.cshtml”. Nessa página foi implementada uma tabela paginada que exibe a lista de eventos e os filtros dos eventos que o usuário poderá definir, como: descrição, cidade, categoria e período. Para cada filtro selecionado é realizada uma requisição no servidor para atualizar a lista dos eventos.

```

@using PagedList.Mvc;
@model PagedList.IPagedList<NetEventos.Models.Evento>
@{
    ViewBag.Title = "Eventos";
}
<h2>Eventos</h2>
<div class="form-horizontal">
    <fieldset>
        <div class="col-md-2" style="padding-right: 0px;padding-left: 0px;">
            @using (Html.BeginForm("Index", "Evento", FormMethod.Get, new { @class =
"form-horizontal" })) {
                <input type="text" class="form-control col-md-2" style="width: 83%"
name="searchString" placeholder="Buscar por evento">
                <button type="submit" id="btnBuscar" class="btn btn-primary"><span
class="glyphicon glyphicon-search" aria-hidden="true"></span></button>
                <label for="estadoId" style="display: block;margin-top: 5%;margin-
bottom: 0%;">Cidade:</label>
                @Html.DropDownList("estadoId", null, htmlAttributes: new { @class =
"form-control col-md-2", @style = "width: 28%" })
                @Html.DropDownList("cidadeId", ViewBag.cidadeId as
IEnumerable<SelectListItem>, htmlAttributes: new { @class = "form-control col-md-2",
@style = "width: 72%" })
                <label for="categoriaId" style="display: block;margin-top: 20%;margin-
bottom: 0%;">Categoria:</label>
                @Html.DropDownList("categoriaId", null, htmlAttributes: new { @class =
"form-control col-md-2", @style = "width: 100%" })
                <label for="dataInicio" style="display: block;margin-top: 20%;margin-
bottom: 0%;">Período:</label>
                <hr style="margin-bottom: 5%; margin-top: 0%" />
                <label for="dataInicio" style="display: block;margin-top: 0%;margin-
bottom: 0%;margin-left: 10px;">de:</label>
                @Html.Editor("dataInicio", new { htmlAttributes = new { @class =
"form-control col-md-2", @style = "width: 95%;margin-left: 10px;", @type = "date",
@value = "dataInicio" } })
                <label for="dataFim" style="display: block;margin-top: 15%;margin-
bottom: 0%;margin-left: 10px;">até:</label>
                @Html.Editor("dataFim", new { htmlAttributes = new { @class = "form-
control col-md-2", @style = "width: 95%;margin-left: 10px;", @type = "date", @value =
"dataFim" } })
            }
        </div>
        <div class="col-md-10">
            <table class="table" style="background-color: whitesmoke;">
                <tr><th width="25%"> @Html.DisplayName("Evento") </th>

```

```

        <th width="8%"> @Html.DisplayName("Data") </th>
        <th width="5%"> @Html.DisplayName("Hora") </th>
        <th width="20%"> @Html.DisplayName("Local") </th>
        <th width="15%"> @Html.DisplayName("Categoria") </th>
        <th> @Html.DisplayName("Cidade") </th></tr>
        @foreach (var item in Model) {
            <tr><td> @Html.DisplayFor(modelItem => item.Descricao) </td>
                <td> @Html.DisplayFor(modelItem => item.Data) </td>
                <td> @Html.DisplayFor(modelItem => item.Hora) </td>
                <td> @Html.DisplayFor(modelItem => item.Local) </td>
                <td>
                    @if (item.Categoria != null) {
                        string cat =
@NetEventos.Helpers.EnumHelper.GetDescription(item.Categoria);
                        <text> @Html.DisplayName(cat) </text>
                    }
                </td>
            <td> @Html.DisplayFor(modelItem => item.Cidade.Nome) </td>
            <td> @Html.ActionLink("Ver evento", "Details", new { id = item.Id }) </td></tr>
        } </table>
        Página @(Model.PageCount < Model.PageNumber ? 0 : Model.PageNumber) de
@Model.PageCount
        @Html.PagedListPager(Model, page => Url.Action("Index", new {
            page, currentFilter = ViewBag.CurrentFilter,
            estadoId = ViewBag.estadoId.SelectedId,
            cidadeId = ViewBag.cidadeId.SelectedId,
            categoriaId = ViewBag.categoriaId.SelectedId,
            dataInicio = ViewBag.dataInicio, dataFim = ViewBag.dataFim}))
    </div>
</fieldset>
</div>

```

Listagem 4 – Evento.cshtml

O método “*Index*”, exibido na Listagem 5, é responsável por retornar a partir do banco de dados uma lista de eventos, filtrando conforme os dados passados por parâmetro. Esse método foi criado dentro do *Controller* “EventoController.cs”, classe que controla os eventos e é utilizada para listar, cadastrar, alterar e excluir eventos.

```

public ActionResult Index(string searchString = "", string currentFilter = "", string
cidadeId = "", string estadoId = "", string categoriaId = "", string dataInicio = "",
string dataFim = "", int? page = 1)
{
    #region UF Cidade
    int ufId = !string.IsNullOrEmpty(estadoId) ? Convert.ToInt32(estadoId) : 18;
    EUF uf = (EUF)ufId;
    int cidade_id = (cidadeId != "" ? Convert.ToInt32(cidadeId) :
_db.Cidades.Where(x => x.UF == uf).Select(x => x.Id).FirstOrDefault());
    ViewBag.cidadeId = new SelectList(_db.Cidades.Where(x => x.UF == uf), "Id",
"Nome", cidade_id);
    var estados = from EUF d in Enum.GetValues(typeof(EUF))
        select new { Id = (int)d, Name = EnumHelper.GetDescription(d) };
    ViewBag.estadoId = new SelectList(estados, "Id", "Name", ufId);
    #endregion UF Cidade
    #region Categoria
    int catId = !string.IsNullOrEmpty(categoriaId) ? Convert.ToInt32(categoriaId):0;
    ECategoria categoria = (ECategoria)catId;
    var categorias = from ECategoria d in Enum.GetValues(typeof(ECategoria))
        select new {Id = (int)d, Name = EnumHelper.GetDescription(d) };
    ViewBag.categoriaId = new SelectList(categorias, "Id", "Name", catId);

```

```

#endregion Categoria
#region Período
ViewBag.dataInicio = dataInicio != "" ?
Convert.ToDateTime(dataInicio).ToString("yyyy-MM-dd") : DateTime.Now.ToString("yyyy-
MM-dd");
ViewBag.dataFim = dataFim != "" ? Convert.ToDateTime(dataFim).ToString("yyyy-MM-
dd") : DateTime.Now.AddMonths(1).ToString("yyyy-MM-dd");
#endregion Período

if (!string.IsNullOrEmpty(searchString))
{
    page = 1;
}
else
{
    searchString = currentFilter;
}
var eventos = _db.Eventos
    .Include(e => e.Cidade);
#region Filtros
if (!string.IsNullOrEmpty(searchString))
{
    eventos = eventos.Where(x =>
x.Descricao.ToUpper().Contains(searchString.ToUpper()));
}
eventos = eventos.Where(x => x.CidadeId == cidade_id);
if (categoria != ECategoria.Empty)
{
    eventos = eventos.Where(x => x.Categoria == categoria);
}
DateTime dtIni = dataInicio != "" ?
Convert.ToDateTime(Convert.ToDateTime(dataInicio).ToString("yyyy-MM-dd 00:00:00"))
    : Convert.ToDateTime((DateTime.Now).ToString("yyyy-MM-dd 00:00:00"));
DateTime dtFim = dataFim != "" ?
Convert.ToDateTime(Convert.ToDateTime(dataFim).ToString("yyyy-MM-dd 23:59:59"))
    : Convert.ToDateTime((DateTime.Now.AddMonths(1)).ToString("yyyy-MM-dd
23:59:59"));
eventos = eventos.Where(x => x.Data >= dtIni && x.Data <= dtFim);
#endregion Filtros
int pageNumber = (page ?? 1);
var pagedEventos = eventos
    .OrderByDescending(x => x.Data)
    .ThenByDescending(x => x.Hora)
    .ToPagedList(pageNumber, _pageSize);
return View(pagedEventos);
}

```

Listagem 5 – EventoController.cs (Método Index)

Os eventos, apresentados Listagem 6, são buscados no banco de dados e retornam uma lista de objetos do tipo evento que é definida na classe “Evento.cs”. Essa classe possui todas as propriedades que o evento terá e suas validações são realizadas por *dataannotations*, *namespace* que fornece classes de atributos que são usados para definir metadados para controles de dados ASP.NET MVC e ASP.NET.

```

using NetEventos.Enumerators;
using System;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace NetEventos.Models
{
    [Table("Evento")]
    public class Evento
    {
        [Key]
        public virtual int Id { get; set; }
        [Display(Name = "Descrição")]
        [MaxLength(100)]
        [Required(ErrorMessage = "O campo descrição é obrigatório", AllowEmptyStrings
= false)]
        public virtual string Descricao { get; set; }
        [Display(Name = "Data")]
        [DisplayFormat(ApplyFormatInEditMode = true, DataFormatString =
"{0:dd/MM/yyyy}")]
        [DataType(DataType.Date)]
        [Required(ErrorMessage = "O campo data é obrigatório", AllowEmptyStrings =
false)]
        public virtual DateTime Data { get; set; }
        [Display(Name = "Hora")]
        [DisplayFormat(ApplyFormatInEditMode = true, DataFormatString = "{0:HH:mm}")]
        [DataType(DataType.Time)]
        [Required(ErrorMessage = "O campo hora é obrigatório", AllowEmptyStrings =
false)]
        public virtual DateTime Hora { get; set; }
        [Display(Name = "Local")]
        [MaxLength(100)]
        [Required(ErrorMessage = "O campo local é obrigatório", AllowEmptyStrings =
false)]
        public virtual string Local { get; set; }
        [Display(Name = "Valor")]
        public virtual decimal Valor { get; set; }
        [Display(Name = "Categoria")]
        [Required(ErrorMessage = "O campo categoria é obrigatório", AllowEmptyStrings
= false)]
        public virtual ECategoria Categoria { get; set; }
        [Display(Name = "Informações")]
        public virtual byte[] Informacoes { get; set; }
        public string UsuarioId { get; set; }
        [Display(Name = "Cidade")]
        [Required(ErrorMessage = "O campo cidade é obrigatório", AllowEmptyStrings =
false)]
        public int CidadeId { get; set; }
        [ForeignKey("CidadeId")]
        public virtual Cidade Cidade { get; set; }
        [Display(Name = "Endereço")]
        public virtual string Endereco { get; set; }
    }
}

```

Listagem 6 – Evento.cs

5 CONCLUSÃO

O sistema desenvolvido possibilita que usuários, geralmente jovens, possam, utilizando sua conta (*login*), possam publicar novos eventos e encontrar eventos que serão realizados. Embora a ideia tenha sido baseada na divulgação de eventos sociais e culturais, o sistema pode ser utilizado para a divulgação dos mais diversos tipos de eventos como esportivos e beneficentes.

As informações sobre os eventos são armazenadas no servidor desenvolvido com a linguagem C#, utilizando o banco de dados SQL Server. A aplicação é para o ambiente *web*, mas como foi desenvolvido com interface responsiva, possibilita que seja acessada por dispositivos com telas de diferentes tamanhos.

As tecnologias utilizadas possuem recursos que facilitam o desenvolvimento de aplicações *web*. Alguns desses recursos são: ASP.NET *Identity*, que gera toda estrutura de gerenciamento de usuários, o *Entity Framework*, que facilita o acesso e persistência de dados no banco e o *Bootstrap*, que possibilita desenvolver a interface do sistema atendendo padrões de usabilidade e, assim, proporcionando melhor interatividade e facilidade de uso. Não foram encontradas dificuldades com as tecnologias durante o desenvolvimento.

REFERÊNCIAS

ASP.NET. **Asp.Net**. Disponível em: <<https://www.asp.net/>>. Acesso em: 03 mar. 2017.

ASTAH. **Astah Community**. Disponível em <<http://astah.net/editions/community>>. Acesso em: 03 mar. 2017.

BOCCHINI, Bruno. **Pesquisa mostra que 58% da população brasileira usam a internet**. 2016. Disponível em: <<http://agenciabrasil.ebc.com.br/pesquisa-e-inovacao/noticia/2016-09/pesquisa-mostra-que-58-da-populacao-brasileira-usam-internet>>. Acesso em: 02 mar. 2017.

CASTELEYN, Sven; GARRIGÓS, Irene; MAZÓN, Jose-Norberto. **Ten years of rich internet applications: a systematic mapping study, and beyond**. ACM Transactions on the Web, Vol. 8, No. 3, Article 18, Publication date: June 2014.

DATABASE DESIGN RESOURCE. **SQL Server overview**. Disponível em: <<http://www.databasedesign-resource.com/sql-server-overview.html>>. Acesso em: 03 mar. 2017.

INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA. **PNAD: de 2005 para 2011, número de internautas cresce 143,8% e o de pessoas com celular, 107,2%**. Disponível em: <<http://saladeimprensa.ibge.gov.br/noticias?view=noticia&id=1&busca=1&idnoticia=2382>>. Acesso em: 16 set. 2015.

JEREMY, Allaire. **Macromedia flash MX - a next-generation rich client**. Tech. rep., Macromedia. March. 2002. Disponível em: <<http://www.c2isoft.in/white-papers/richclient.pdf>>. Acesso em: 01 mar. 2017.

MARTÍNEZ-RUIZ, Francisco J. **The triad-based design of rich user interfaces for internet applications**. In: 2nd ACM SIGCHI Symposium on Engineering Interactive Computing Systems, 2010, p. 345-348.

MARTÍNEZ-RUIZ, Francisco J. et al. **A first draft of a model driven method for designing graphical user interfaces of rich internet applications**. In: 4th Latin American Web Congress LA-Web'2006, IEEE Computer Society Press, 2006.

MICROSOFT. **C#**. 2017a. Disponível em: <<https://msdn.microsoft.com/pt-BR/library/kx37x362.aspx>>. Acesso em: 03 mar. 2017.

MICROSOFT. **Visão geral do ASP.NET MVC**. 2017b. Disponível em: <[https://msdn.microsoft.com/pt-br/library/dd381412\(v=vs.108\).aspx](https://msdn.microsoft.com/pt-br/library/dd381412(v=vs.108).aspx)>. Acesso em: 03 mar. 2017.

ROUBI, Sarra; ERRAMDANI, Mohammed; MBARKI, Samir. **A model driven approach to generate graphical user interfaces for rich internet applications using interaction flow modeling language**. In: 15th International Conference on Intelligent Systems DeSign and Applications. IEEE. 2015, p. 272-276/

SANTOS, Carlos dos. **Fundamentos do Entity Framework 4**. 2012.
<<https://msdn.microsoft.com/pt-br/library/jj128157.aspx>>. Acesso em: 03 mar. 2017.

VISUAL STUDIO. **Visual Studio**. Disponível em: <<https://www.visualstudio.com/pt-br/vs/community/>>. Acesso em: 03 mar. 2017.