CENTRO ESTADUAL DE TECNOLOGIA PAULA SOUZA
FACULDADE DE TECNOLOGIA DE MAUÁ

MELISSA DE FREITAS SANTOS

SISTEMA DE GERENCIAMENTO PARA GAME HOUSE

MAUÁ/SP
2023

MELISSA DE FREITAS SANTOS

SISTEMA DE GERENCIAMENTO PARA GAME HOUSE

Monografia apresentada à FATEC Mauá, como parte dos requisitos para obtenção do Título de Tecnólogo em Informática para Negócios.
Orientador: Prof. Me. Ivan Carlos Pavão

MAUÁ/SP

2023

Santos, Melissa de Freitas.
Sistema de gerenciamento para Game House. Santos, Melissa de Freitas

0 p.; 30 cm.

TCC (Trabalho de Conclusão de Curso).
CEETEPS-FATEC Mauá/SP, 0º Sem. 2023.
Orientador: Prof. M. Sc.

Referências: p. 0.
Palavras-chave: Informação, Logística, Sistema, Game House.

MELISSA DE FREITAS SANTOS

SISTEMA DE GERENCIAMENTO PARA GAME HOUSE

Monografia apresentada à FATEC Mauá, como parte dos requisitos para obtenção do Título de Tecnólogo em Informática para Negócios.

Aprovação em: 0 jun. 2023.

_____

Prof. Me. Ivan Carlos Pavão

FATEC Mauá

Orientador

_____

Prof (a) _____

FATEC Mauá

Avaliador(a)

_____

Prof (a) _____

FATEC Mauá

Avaliador(a)

Dedicarei este trabalho, primeiramente, a Oxalá, por ser essencial em minha vida, aos meus pais, pelos incentivos, os quais tornaram possível a conclusão desta etapa em da minha vida e, por fim, aos professores que me auxiliaram nessa jornada de estudo.

“Uma pessoa que nunca cometeu um erro nunca tentou nada de novo”

Albert Einstein

RESUMO

O presente trabalho

Palavras-chave: Informação, Logística, Sistema, Game House.

ABSTRACT

Keywords: Information, Logistics, System, Game House.

Diagramas

Tabelas

Sumário

1. INTRODUÇÃO

Tecnologia se tornou algo essencial na vida das pessoas, facilitando seu dia a dia. Atualmente ela vem sendo inovada em grande velocidade e trazendo otimização nos sistemas. Muitas empresas necessitam de sistemas para auxiliar no gerenciamento e tomada de decisão.

Com a crescente onda de jogos na era digital, muitas pessoas começaram a comprar jogos e consoles, porém muitos se arrependem de suas escolhas por serem caros ou não gostarem. Visto isso, foi criado uma adaptação das antigas Locadoras, chamado Game House, aonde as pessoas podem ir para alugarem jogos e levarem para casa ou até mesmo jogar no local utilizando os consoles da loja, os mesmos que servem para testes de comparação de qualidade, usabilidade, entre outros fatores, de um console para outro e de jogo para outro.

Desta maneira, o presente trabalho tem como objetivo apresentar a prototipagem de um sistema de gerenciamento que facilite a administração de uma Game House, para que o sistema seja efetivo deve-se ser cadastrado os funcionários, os clientes, os produtos, os agendamentos, os pagamentos e em função disso, foi feito um planejamento após análises de requisitos necessários para o desenvolvimento, que serão apresentados ao decorrer do trabalho.

Considerando a grande relevância que um sistema tem para uma empresa, trazendo uma dependência cada vez maior para o fluxo de serviços aplicados pelas entidades empresariais, esse tipo de ferramenta trará valor para a empresa, centralizando os dados, trazendo a agilidade e facilitando na análise da situação da organização, garantindo o melhor desempenho empresarial.

Este trabalho foi redigido por meio de pesquisas bibliográficas e on-line. Através de experiências de profissionais que a autora do trabalho vivenciou, percebe-se a grande necessidade da criação de softwares específicos para o gerenciamento de organizações.

Serão abordados os temas: Game House, tecnologias utilizadas no desenvolvimento, arquitetura Model-View-Controller, desenvolvimento e o manual do sistema. Ao final das abordagens do tema será apresentado, também, o protótipo do projeto proposto pela autora do trabalho.

2. CONCEITUANDO UMA GAME HOUSE

Comentado [mf1]: Conceituar a game house

https://www.legiaodosherois.com.br/2019/locadoras-de-games-no-brasil-mais-que-diversao-um-estilo-de-vida.html

https://www.youtube.com/watch?v=H4ZAEhuZpjA

2.1 Modelo de negócio

Como objeto de estudo e prototipagem, foi escolhido utilizar um empreendimento fictício de uma locadora de games ou game house, e para que o software tivesse regras de negócio baseado a autora criou um plano de negócio resumido que contemplasse algumas exigências.

Tipo de empreendimento	Locadora de games
Nome fantasia do sistema	Violet_Games
Clientes	Jogadores de todas as idades, pessoas que querem testar algo novo ou tem medo de comprar o jogo/videogame e não gostar.
Proposta de valor	Sala de jogos equipada com consoles disponíveis + Jogos compatíveis para esses consoles que podem ser jogados no espaço ou alugados.
Relacionamentos com clientes	O plano mensal permite que o cliente utilize a sala de jogos com prioridade no agendamento de uso, além de poder alugar 3 jogos por vez. Os 10 primeiros clientes que assinarem o plano mensal, terão um desconto de 50% na mensalidade por 6 meses, os aniversariantes ganham 10% de desconto no plano mensal.
Fontes de receita	A sala de jogos pode ser usada por meio de agendamento, qualquer console pode ser usado pelo valor de R$4,00/H, para locação de jogos o valor da diária é R$8,00/por jogo. Plano mensal R$ 49,90 (inclui o uso da sala de jogos + locação de jogos).
Recursos principais	Xbox 360, Xbox One, PS1, PS2, PS3, PS4, PS5, 3 PC's gamer, Nitendo Switch.
Atividades-chave	Agendamento do uso da sala de jogos, locação dos jogos, monitoramento do uso consoles/jogos.

Tabela 1 - Plano de negócio (Própria autora)

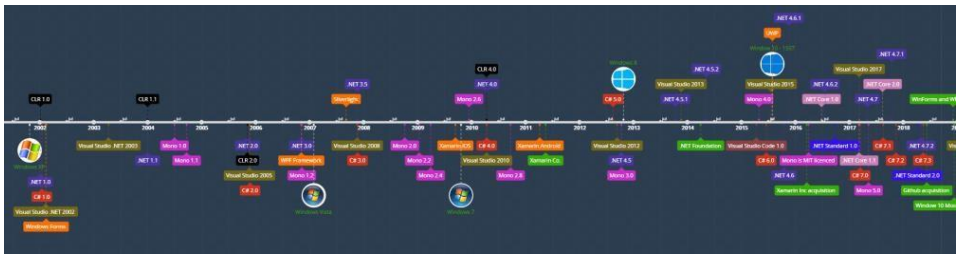
3. TECNOLOGIAS UTILIZADAS NO DESENVOLVIMENTO

3.1 C#, .NET e ASP.NET

Para a criação de sistemas, escolher a linguagem e as tecnologias a serem utilizadas é um passo fundamental, normalmente, a escolha da linguagem é determinada a partir da plataforma, da natureza e da cultura da empresa, para esse projeto a plataforma principal escolhida foi o .NET 5 e como linguagem o C#.

Segundo a documentação oficial da Microsoft, o .NET (DotNet) é um sistema de execução virtual chamado Common Language Runtime (CLR) e um conjunto de classes. O CLR é a implementação da Microsoft da CLI (Common Language Infrastructure), um padrão internacional. A CLI é a base para a criação de ambientes de execução e desenvolvimento nos quais as linguagens e bibliotecas funcionam em conjunto diretamente.

Comentado [mf2]: GEWARREN. **Introdução ao .NET Framework - .NET Framework**. Disponível em: <https://docs.microsoft.com/pt-br/dotnet/framework/get-started/>. Acesso em: 10 mar. 2022.



https://time.graphics/pt/line/291016

A primeira versão do .NET foi lançada no ano de 2002, sendo chamada de .NET Framework 1.0, ela era compatível apenas ao Sistema Operacional Windows, o que limitava o poder de desenvolvimento, porém a ideia da plataforma era permitir a utilização de várias linguagens numa plataforma única, podendo ser utilizado: VB.NET, C++, J# e F# além de outras linguagens de programação. Durante anos foram lançadas várias versões do .NET Framework, assim a framework foi sendo atualizada e melhorada.

Comentado [mf3]: Contextualizar a criação do c#

https://www.alura.com.br/apostila-csharp-orientacao-objetos/o-que-e-c-e-net

https://coodesh.com/blog/dicionario/o-que-e-dotnet/#:~:text=Origem%20e%20varia%C3%A7%C3%B5es,carreira%20de%20Back%2Dend%20C%23%20.

Para esse projeto dentro das versões do .Net, foi selecionado a versão 5, essa versão https://balta.io/blog/dotnet

Segundo a documentação oficial da Microsoft, o C# (CSharp) é uma linguagem de programação orientada a objetos e orientada a componentes, criada pela Microsoft e lançada no

Comentado [mf4]: ERIKDIETRICH. **O histórico da linguagem C# – Guia do C#**. Disponível em: <https://docs.microsoft.com/pt-br/dotnet/csharp/whats-new/csharp-version-history>. Acesso em: 10 mar. 2022.

mesmo mês do lançamento do .NET 1.0, tendo como base a familia da linguagem de programação C, além de ter sido influênciada por outras linguagens de programação, como Pascal e Java. Foi desenvolvido por Anders Hejlsberg, um engenheiro de software dinamârques e uma equipe de programadores que trabalhavam para a Microsoft. A linguagem foi desenvolvida no intuito de flexibilizar o desenvolvimento de aplicativos e possibilitar a criação de soluções executáveis sobre a plataforma .NET, assim o desenvolvedor não cria soluções para um dispositivo de aplicativos, e sim para a plataforma .NET.

*Add uma referência sobre C#

https://www.devmedia.com.br/a-evolucao-da-linguagem-de-programacao-csharp/28639

Para facilitar a codificação de sistemas baseados em .NET Framework, foi lançado o Visual Studio, pela Microsoft. Esta ferramenta é uma IDE(Integrated Development Environment) que é utilizado na edição de código e na compilação, com modelos de projetos de exemplos, designers e assistente de códigos.

De acordo com os desenvolvedores Fabrício Sanchez e Márcio Fábio, conforme o cenário web transformou-se rápida e largamente nos anos posteriores (mudança alavancada especialmente pelos negócios, que igualmente passaram a ser realizados neste ambiente), fez-se necessária a estruturação de uma nova tecnologia para desenvolvimento web incorporado da plataforma .NET, que atendesse às novas demandas de mercado e que pudesse endereçar os problemas mencionados anteriormente. Surgiu logo, o ASP.NET MVC.

https://blog.betrybe.com/framework-de-programacao/asp-net-o-que-e/

https://www.treinaweb.com.br/blog/o-que-e-o-asp-net-core

https://acervolima.com/introducao-ao-asp-net/

http://www.linhadecodigo.com.br/artigo/1602/consideracoes-iniciais-sobre-o-aspnet-mvc-framework.aspx

https://www.devmedia.com.br/net-brasil-asp-net-e-html-5-revista-net-magazine-90/22575

3.2 SQL Server

O SQL Server é um Sistema de Gerenciamento de Banco de Dados (SGBD) da empresa Microsoft, foi criado em parceria com a Sybase, em 1988, inicialmente como um complemento do Sistema Operacional Windows NT, e logo depois passou a ser refinado e vendido separadamente. A parceria da Microsoft com a Sybase terminou em 1994, e o desenvolvimento do programa continuou a ser feito pela Microsoft.

Comentado [mf5]: Add uma referência sobre C#

Comentado [mf6]: SANCHEZ, F.; MÁRCIO FÁBIO ALTHMANN. **Desenvolvimento web com ASP.NET MVC**. [s.l.] Editora Casa do Código, 2013.

Comentado [mf7]: Pesquisar mais sobre o asp.net

Comentado [mf8]: **Microsoft SQL Server**. Disponível em: <https://pt.wikipedia.org/wiki/Microsoft_SQL_Server>. Acesso em: 23 feb. 2023.

Esse banco de dados é classificado como um Banco de Dados Relacional (RDBMS, na sigla em inglês), segundo o site da Oracle, um banco de dados relacional é um tipo de banco de dados que armazena e fornece acesso a pontos de dados relacionados entre si. Bancos de dados relacionais são baseados no modelo relacional, uma maneira intuitiva e direta de representar dados em tabelas. Em um banco de dados relacional, cada linha na tabela é um registro com uma ID exclusiva chamada chave. As colunas da tabela contêm atributos dos dados e cada registro geralmente tem um valor para cada atributo, facilitando o estabelecimento das relações entre os pontos de dados.

A principal linguagem de consulta do SQL Server é a Transact-SQL, que utiliza a implementação do padrão ANSI/ISO o Structured Query Language (SQL). O SQL se origina de um modelo de dado relacional chamado de SEQUEL ou *Linguagem de Consulta em Inglês Estruturado (em inglês: Strutucred English Query Language)* que foi desenvolvido nos anos 70, com base nas pesquisas sobre arranjo de dados realizadas pelo cientista inglês Edgar Frank Codd que na época era pesquisador da IBM.

Atualmente, esse SGBD continua a ser um dos mais utilizados, ele possui versões gratuitas e pagas. As versões posteriores ao SQL Server 2008 R2 impressionaram desenvolvedores pela simplicidade, diminuindo o tempo para a criação do banco de dados, além de permitirem ao desenvolvedor usar uma linguagem de programação gerenciada, como C# ou VB.NET, para o endereçamento das consultas, ao invés de usar declarações SQL. Outra vantagem são as consultas transparentes orientadas ao conjunto, escritas em .NET.

3.3 Entity Framework

3.4 HTML, CSS e JavaScript

3.5 Bootstrap

4. ARQUITETURA MODEL-VIEW-CONTROLLER (MVC)

Na fase de projeto, o tópico fundamental é a escolha e o desenho da arquitetura da aplicação, Martin Fowler considera que a arquitetura de uma aplicação possui dois objetivos essenciais: decompor esse sistema em suas partes principais, em alto nível, e representar um modelo geral de forma estável, ou seja, sem grande tendência a alterações. Após muitas pesquisas sobre arquiteturas existentes e as mais utilizadas no mercado de desenvolvimento, foi avaliado que a arquitetura mais compatível para a criação do sistema proposto, seria a Arquitetura Model-View-Controller.

O Model-View-Controller conhecido como MVC, foi desenvolvido na década de 70, pelo cientista da computação norueguês e professor emérito da Universidade de Oslo, Trygve Mikkjel Heyerdahl Reenskaug enquanto trabalhava na Xerox PARC. Utilizando a plataforma de desenvolvimento ASP.NET MVC da Microsoft, esse modelo contribuiu para a redução do acoplamento entre classes, auxiliando no reuso.

O MVC consiste na partição do código do software em três camadas funcionais para serem independentes, criando assim uma facilidade na manutenção do código e sua reutilização em outros projetos. As três camadas são nomeadas de Model (Modelo), View (Visualização) e Controller(Controlador).

Comentado [mf9]: O que é um banco de dados relacional? Disponível em: <https://www.oracle.com/br/database/what-is-a-relational-database/>. Acesso em: 23 feb. 2023.

Comentado [mf10]: KLINE, K.; HUNT, B.; KLINE, D. **SQL in a Nutshell.** [s.l.] "O'Reilly Media, Inc.", 2004.

Comentado [mf11]: FOWLER, Martin. (**2006) Padrões de Arquitetura de Aplicações Corporativas**. Bookman, São Paulo.

Comentado [mf12]: BURBECK, Steve. **Applications Programming in Smalltalk-80(TM): How to use Model-View-Controller (MVC)**. Disponível em: <https://folk.universitetetioslo.no/trygver/themes/mvc/mvc-index.html>. Acesso em: 10 mar. 2022.

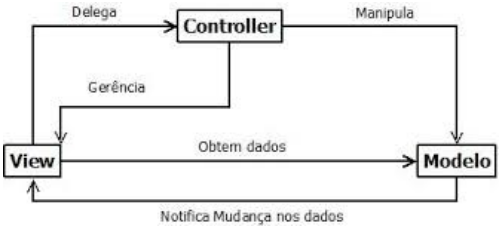


Figura 1 - Diagrama do modelo MVC

Conceito do padrão MVC: (ASP.NET Core MVC: Aplicações modernas em conjunto com o Entity Framework)

1. O controlador (Controller), é responsável pelas interpretações das entradas do mouse ou do teclado enviadas pelo usuário, assim ele mapeia essas ações do usuário em comandos que são enviados para o modelo (Model) e/ou para a janela de visualização (View) para executar a modificação apropriada;
2. O modelo (Model), faz o gerenciamento de um ou mais elementos de dados, respondendo a perguntas sobre o seu estado, e respondendo a instruções para alterar de estado. O modelo sabe o que o aplicativo quer executar e é a principal estrutura computacional da arquitetura, pois é ele quem modela o problema a ser resolvido;
3. Por fim, a visão (View) faz o gerenciamento do espaço retangular do display e é responsável por exibir as informações para o usuário através de uma combinação de gráficos e textos. A visão não sabe nada sobre o que a aplicação está atualmente fazendo, pois tudo que ela efetivamente faz é receber instruções do controle e informações do modelo e logo exibi-las. A visão igualmente se comunica de volta com o modelo e com o controlador para reportar o seu estado.

Existem vários tipos de Arquiteturas, a MVC por ser uma arquitetura que divide o projeto em três camadas é confundido em várias ocasiões com a Arquitetura de Aplicação de Três Camadas, embora ambas tenham o mesmo primórdio em separarem a aplicação em três camadas há fatores que as diferenciam.

Segundo [Brown *et al.*], a arquitetura Três Camadas possui três camadas principais, sendo elas: Apresentação, Domínio e fonte de dados. As três camadas dessa arquitetura estão sendo representadas em resumo na tabela abaixo.

Tabela 1.1 Três Camadas Principais

Camada	Responsabilidades
Apresentação	Fornecimento de serviços, exibição de informações (p. ex., em Windows ou HTML, tratamento de solicitações do usuário (cliques com o *mouse*, pressionamento de teclas), requisições HTTP, chamadas em linhas de comando, API em lotes)
Domínio	Lógica que é o real propósito do sistema
Fonte de Dados	Comunicação com os bancos de dados, sistemas de mensagens, gerenciadores de transações, outros pacotes

Tabela 2 - Três camadas principais [Brown et al.]

Segundo Leonel Sanches da Silva (Arquiteto de Soluções da Twilio), para entender a diferença, é necessário entender do ponto de vista de um Model. Um Model é uma classe que define não apenas os elementos de dados, mas quais valores eles podem receber, como são validados e

Comentado [mf13]: Arrumar referência

Comentado [mf14]: ARAÚJO, Everton Coimbra. **ASP.NET Core MVC: Aplicações modernas em conjunto com o Entity Framework.** São Paulo: Casa do Código, 2018.

Comentado [mf15]: [Brown *et al.*] Brown et al. **Enterprise Java Programming with IBM Websphere.** Addison-Wesley, 2001.

Comentado [mf16]: **c# - 3 camadas vs MVC**. Disponível em: <https://pt.stackoverflow.com/questions/33352/3-camadas-vs-mvc>. Acesso em: 20 jan. 2023.

as relações de um Model com outro, coisa que não existe no Modelo de 3 Camadas. Na arquitetura de 3 Camadas, é necessário colocar validações, relações e características de cada entidade na camada de dados, ou na camada de negócio.

A responsabilidade de um Controller é a de harmonizar e arbitrar as relações entre Models. É ele que comanda a criação, modificação, exclusão e seleção dos dados da aplicação. Além disso, é ele que recebe a requisição e decide o que deve ser retornado como apresentação, como por exemplo, o formato dos dados (HTML, JSON, e assim por diante).

Há abordagens que procuram colocar uma camada extra para trabalhar juntamente com o Controller, sob a alegação de que não é responsabilidade do Controller de cuidar de regras de negócio. Isto não é verdadeiro, se for considerado como aspecto algo que ele é responsável por fazer, no caso, a harmonização de dados entre Models.

Comentado [mf17]: Add mais sobre MVC

5. DESENVOLVIMENTO

5.1 Requisitos do sistema

Requisitos em sua descrição consiste na identificação documentada de uma característica ou comportamento que um produto deve atender. São a princípio, para capturar e transmitir necessidades, gerenciar expectativas, priorizar e atribuir trabalho, verificar e validar o sistema (aceitação) e gerenciar o objetivo do projeto.

> *Requisitos definem o que um sistema deve fazer e sob quais restrições. Requisitos relacionados com a primeira parte dessa definição — o que um sistema deve fazer, ou seja, suas funcionalidades — são chamados de Requisitos Funcionais. Já os requisitos relacionados com a segunda parte — sob que restrições — são chamados de Requisitos Não-Funcionais.* De <https://engsoftmoderna.info/cap3.html>

Comentado [mf18]: VALENTE, M. T. **Cap. 3: Requisitos – Engenharia de Software Moderna**. Disponível em: <https://engsoftmoderna.info/cap3.html>.

Para esse projeto foram identificados e documentados os principais requisitos Funcionais e Não-Funcionais:

Comentado [mf19]: Comentar a referencia

- **Requisitos Funcionais**

Código	Requisito	Descrição	Ator
RF-001	Cadastramento de funcionários	Deverá permitir o usuário a realizar o cadastramento de funcionários, as ações que estarão disponíveis serão: criar, remover, alterar e consultar o cadastro de funcionários.	Apenas o Gerente (administrador)
RF-002	Cadastramento de clientes	Deverá permitir o usuário realize o cadastramento de clientes, separando os clientes que tem plano mensal e os que vão apenas utilizar o serviço sem plano mensal, as ações que estarão disponíveis serão: criar, remover, alterar e consultar o cadastro de clientes.	Funcionário (padrão) e Gerente (administrador)

RF-003	Cadastramento de produtos	Deverá permitir o usuário realize o cadastramento de produtos, as ações que estarão disponíveis serão: criar, remover, alterar e consultar o cadastro de produtos.	Funcionário (padrão) e Gerente (administrador)
RF-004	Pagamento	Deverá permitir o usuário cobre o pagamento do cliente, dando a opção de pagamento (cartão de crédito e débito), em caso de pagamento em dinheiro, informar qual será o valor do troco se necessário.	Funcionário (padrão) e Gerente (administrador)
RF-005	Realizar o controle do pagamento do plano mensal	Deverá permitir o usuário realize o controle do pagamento do plano mensal, vinculado ao cadastro do cliente deve haver uma tag que indica se o pagamento está em dia ou está pendente. As ações que estarão disponíveis serão: alterar e consultar.	Apenas o Gerente (administrador)
RF-006	Realizar agendamento de console	Deverá permitir o usuário realize o agendamento do uso do console, as ações que estarão disponíveis serão: criar, remover, alterar e consultar reservas. Cada reserva, deverá ter um cliente e um console em respectivo período.	Funcionário (padrão) e Gerente (administrador)
RF-007	Realizar o monitoramento do uso dos consoles	Deverá permitir o usuário realize o monitoramento do uso dos consoles (tempo de uso).	Funcionário (padrão) e Gerente (administrador)
RF-008	Realizar a locação de jogos	Deverá permitir o usuário realize a locação de jogos disponíveis, com a regra de que clientes com plano mensal podem alugar até 3 jogos por vez, clientes sem plano podem alugar 1 jogo por vez, ou seja, deve estar vinculado com o cadastro do cliente. As ações que estarão disponíveis serão: criar, remover, alterar e consultar.	Funcionário (padrão) e Gerente (administrador)
RF-009	Venda de produtos	Deverá permitir o usuário efetue vendas de produtos: criar, remover, alterar e consultar pedidos.	Funcionário (padrão) e Gerente (administrador)

Tabela 3 - Requisitos Funcionais (Própria autora)

- **Requisitos Não-Funcionais**

Código	Requisito	Descrição

RNF-001	Restrições de Hardware	Requisito mínimo de hardware: processador intel i3, amd ryzen 3(ou sucessores) baseado em x64, memória ram 8GB, com placa de rede.
RNF-002	Restrições de software	O software do cliente, deverá executar no browser, com servidor de banco de dados (SQL-Server) na linguagem C#.
RNF-003	Usabilidade	Facilidade de navegação.
RNF-004	Restrição de acesso	Ter mecanismos de controle de acesso para apenas pessoas autorizadas terem acesso

Tabela 4 - Requisitos Não- Funcionais (Própria autora)

Assim, os seguintes recursos deverão ser implementados:

Area administrativa da aplicação: área de acesso restrito, onde os usuários precisarão realizar o processo de autenticação para ter o devido acesso. Além do mecanismo de login, dentro da área administrativa serão implementados todos os cadastros (Funcionários, usuários, clientes, produtos, agendamentos e planos de clientes);

Gerenciamento de Funcionários: acoplada a área administrativa será criado uma subárea para CRUD (Create, Read, Update e Delete) de novos funcionários que poderá ser acessado apenas pelo administrador do sistema;

Gerenciamento de usuários: também acoplada a area administrativa, será criada uma subárea para CRUD de usuários que poderá ser acessado apenas pelo administrador do sistema, limitando as ações do usuário no sistema;

Gerenciamento de produtos: acoplada a área administrativa será criado uma subárea para CRUD (Create, Read, Update e Delete) de produtos, onde o administrador e o funcionário poderão gerenciar o estoque de produtos.

Gerenciamento de Agendamentos: acoplada a área administrativa será criado uma subárea para CRUD (Create, Read, Update e Delete) de Agendamentos, onde o administrador e o funcionário poderão gerenciar o agendamento de locação de jogos e de consoles.

Gerenciamento de plano de clientes: acoplada a área administrativa será criado uma subárea para CRUD (Create, Read, Update e Delete) de plano de cliente.

5.2 Diagramas de Caso

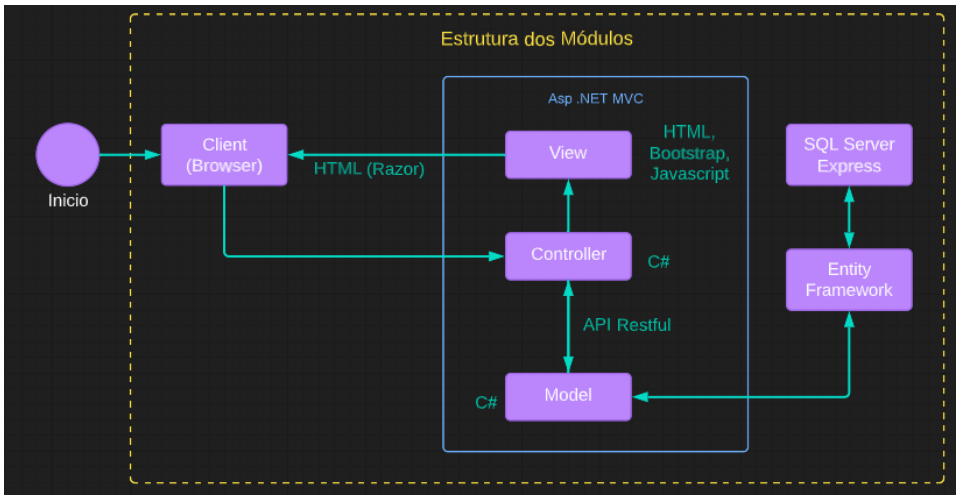


Figura 2 – Estrutura dos Módulos

- **Cadastros**

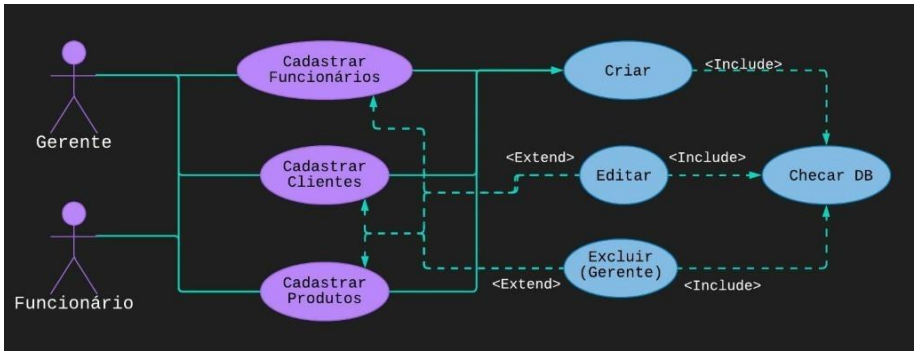


Figura 3 - Diagrama de caso: Cadastros

- **Alugar jogos**

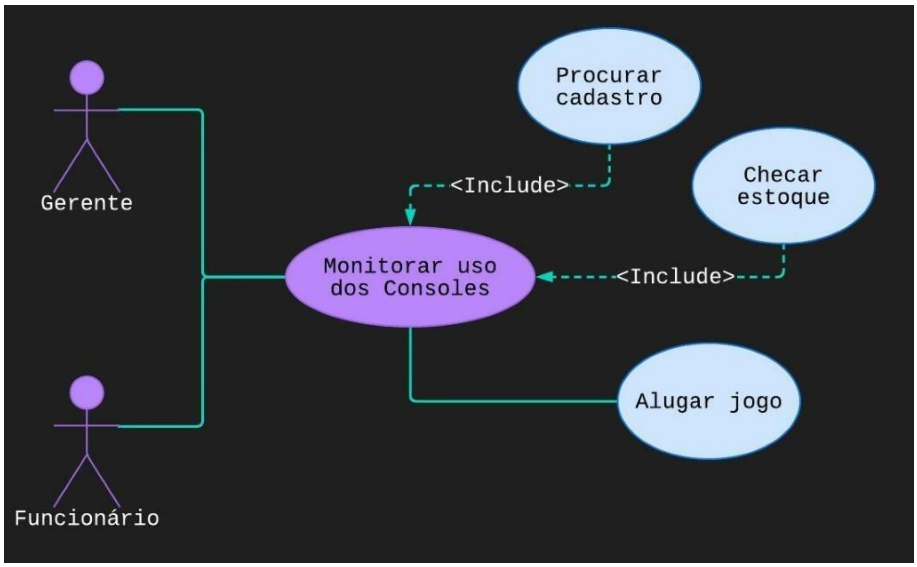


Figura 4 - Diagrama de caso: Locação de Jogos

- **Monitoramento de consoles**

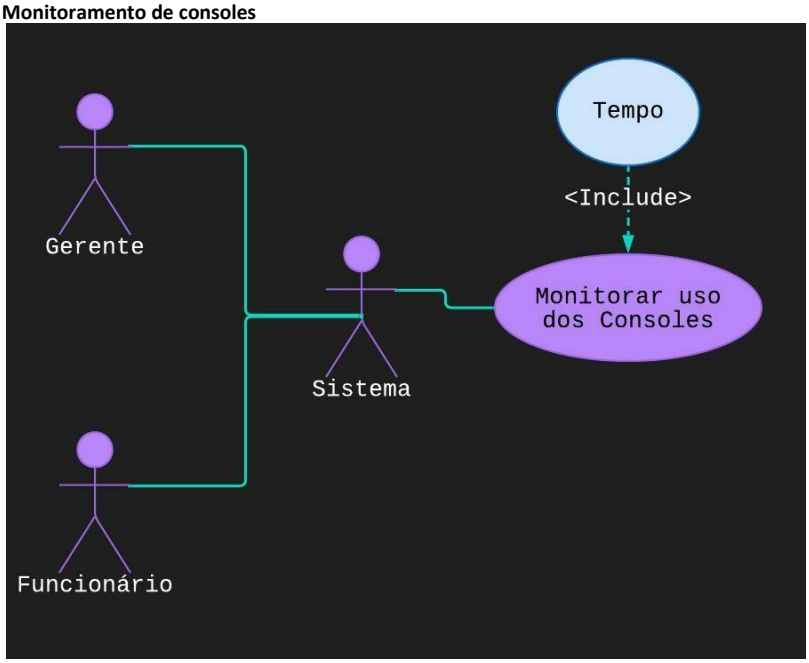


Figura 5 - Diagrama de caso: Monitoramento do Consoles

- **Pagamentos**

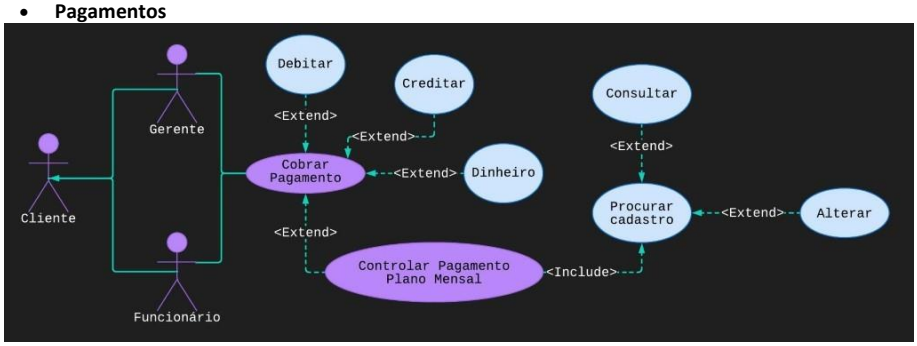


Figura 6 - Diagrama de caso: Pagamentos

- **Venda de produtos**

5.3 Diagrama de Sequência

5.4 Diagrama de Classe

5.5 Diagrama Entidade Relacional

Comentado [mf20]: Pesquisar

Segundo Fabrício Sanchez e Márcio Fábio, o Diagrama Entidade-Relacionamento (ou simplesmente DER) é uma metodologia que permite criar, e posteriormente exibir, de forma gráfica e simplificada—nível de abstração mais alto—uma estrutura mais complexa de regras e agrupamento de dados em uma aplicação.

Comentado [mf21]: SANCHEZ, F.; MÁRCIO FÁBIO ALTHMANN. **Desenvolvimento web com ASP.NET MVC**. [s.l.] Editora Casa do Código, 2013.

Através da representação gráfica clara possibilitada pelos DER's, é possível compreender a lógica de dados e, até mesmo, algum comportamento da aplicação. Além disso, os diagramas expressam os relacionamentos entre as entidades importantes para o sistema, o que facilita o entendimento da situação problema e seguinte implementação, tanto para desenvolvedores quanto para administradores de bancos de dados. Trata-se do "mapa" criado pelos analistas de requisitos e passado para os desenvolvedores, arquitetos de software e administradores de bancos de dados.

6. MANUAL DO SISTEMA

6.1 Tela de Login

6.2 Tela do Dashboard

6.3 Tela do Caixa

6.4 Tela de Produtos

6.5 Tela de Consoles

6.6 Tela de Clientes

6.7 Tela de Funcionários

6.8 Tela de Configuração

7. CONCLUSÃO

8. REFERÊNCIAS

Comentado [mf22]: Colocar print da tela com fonte: próprio autor

Comentado [mf23]: Colocar o print da tela do usuário standard e da tela de admin, situando a diferencia.

GEWARREN. **Introdução ao .NET Framework - .NET Framework**. Disponível em: <https://docs.microsoft.com/pt-br/dotnet/framework/get-started/>. Acesso em: 10 mar. 2022.

ERIKDIETRICH. **O histórico da linguagem C# – Guia do C#**. Disponível em: <https://docs.microsoft.com/pt-br/dotnet/csharp/whats-new/csharp-version-history>. Acesso em: 10 mar. 2022.

Microsoft SQL Server. Disponível em: <https://pt.wikipedia.org/wiki/Microsoft_SQL_Server>. Acesso em: 23 feb. 2023.

KLINE, K.; HUNT, B.; KLINE, D. **SQL in a Nutshell.** [s.l.] "O'Reilly Media, Inc.", 2004.

O que é um banco de dados relacional? Disponível em: <https://www.oracle.com/br/database/what-is-a-relational-database/>. Acesso em: 23 feb. 2023.

c# - 3 camadas vs MVC. Disponível em: <https://pt.stackoverflow.com/questions/33352/3-camadas-vs-mvc>. Acesso em: 20 jan. 2023.

ARAÚJO, Everton Coimbra. **ASP.NET Core MVC: Aplicações modernas em conjunto com o Entity Framework.** São Paulo: Casa do Código, 2018.

SANCHEZ, F.; MÁRCIO FÁBIO ALTHMANN. **Desenvolvimento web com ASP.NET MVC**. [s.l.] Editora Casa do Código, 2013.

[Brown *et al.*] Brown et al. **Enterprise Java Programming with IBM Websphere.** Addison-Wesley, 2001.

FOWLER, Martin. **(2006) Padrões de Arquitetura de Aplicações Corporativas**. Bookman, São Paulo.

BURBECK, Steve. **Applications Programming in Smalltalk-80(TM): How to use Model-View-Controller (MVC)**. Disponível em: <https://folk.universitetetioslo.no/trygver/themes/mvc/mvc-index.html>. Acesso em: 10 mar. 2022.

VALENTE, M. T. **Cap. 3: Requisitos – Engenharia de Software Moderna**. Disponível em: <https://engsoftmoderna.info/cap3.html>.

AJCVICKERS. **Visão geral do Entity Framework Core – EF Core**. Disponível em: <https://learn.microsoft.com/pt-br/ef/core/>. Acesso em: 10 mar. 2022.

O que é arquitetura de três camadas (tiers). Disponível em: <https://www.ibm.com/br-pt/cloud/learn/three-tier-architecture#toc-outras-arq-uMx8DOlM>. Acesso em: 20 mar. 2022.

FREEMAN, Eric; FREEMAN, Elisabeth. **Use a cabeça! padrões de projeto: Design Patterns**. 2. ed. São Paulo: Alta Books, 2007.