Introdução

terça-feira, 8 de fevereiro de 2022 11:27

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 Core e como linguagem o C#.
*OBS: Parágrafo acima pode ser usado na introdução.

Problema + Modelo de negócio

terça-feira, 7 de junho de 2022 15:19

Como objeto de estudo e prototipagem, foi escolhido utilizar um projeto acadêmico feito durante as aulas de ministradas pelo professor que leciona a matéria de Engenharia de Software. O professor propôs aos alunos escolherem um empreendimento fictício, após a escolha deviam criar um pequeno modelo de negócio e a partir do modelo do empreendimento deviam criar alguns requisitos de sistema funcionais e não-funcionais.

Modelo de Negócio

- **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 (OBS: pode ser jogado na plataforma escolhida um jogo por vez), 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:** 1 Xbox 360, 1 Xbox One, 1 PS1, 1 PS2, 1 PS3, 1 PS4, 1 PS5, 3 PC's gamer, 1 Óculos VR para PS4, 1 Nitendo Switch.

- **Atividades-chave:** agendamento do uso da sala de jogos, locação dos jogos, gestão de compra de novos jogos/aparelhos, monitoramento do uso consoles/jogos, tirar dúvidas dos clientes.

- **Fornecedores:** SND Distribuição (vende jogos e itens de informática), ShopB (além de comercializar videogames e **acessórios para games, disponibiliza alguns cursos para quem está mergulhando com tudo nesse universo), MH Games (Loja que fornece preços diferenciados para clientes do varejo e do atacado games)**

- **Estrutura de custo:** energia(R$500,00/mês), água(R$50,00/mês), internet e telefone(R$140,00), salários(R$2400,00), aluguel(R$1000,00).

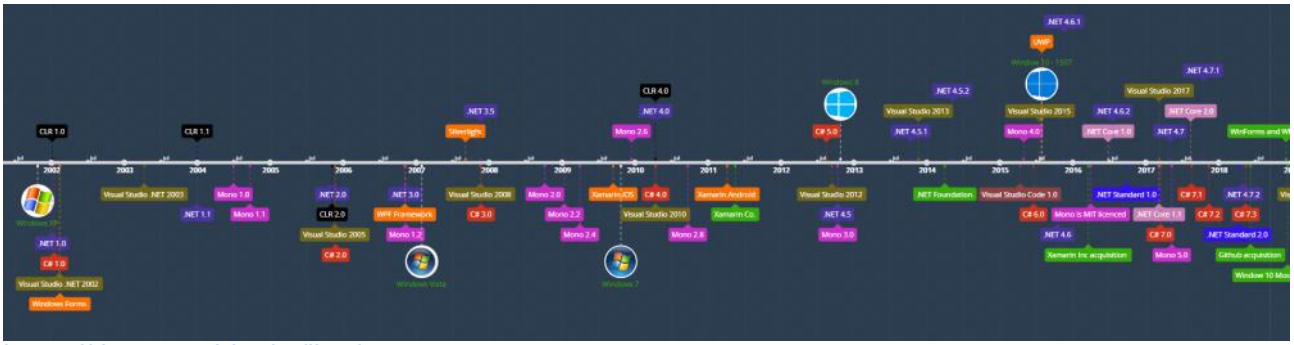
.NET

terça-feira, 7 de junho de 2022 15:17

*Porque criar o .net?

Segundo a documentação oficial da Microsoft, o .NET 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. https://docs.microsoft.com/pt-br/dotnet/framework/get-started/

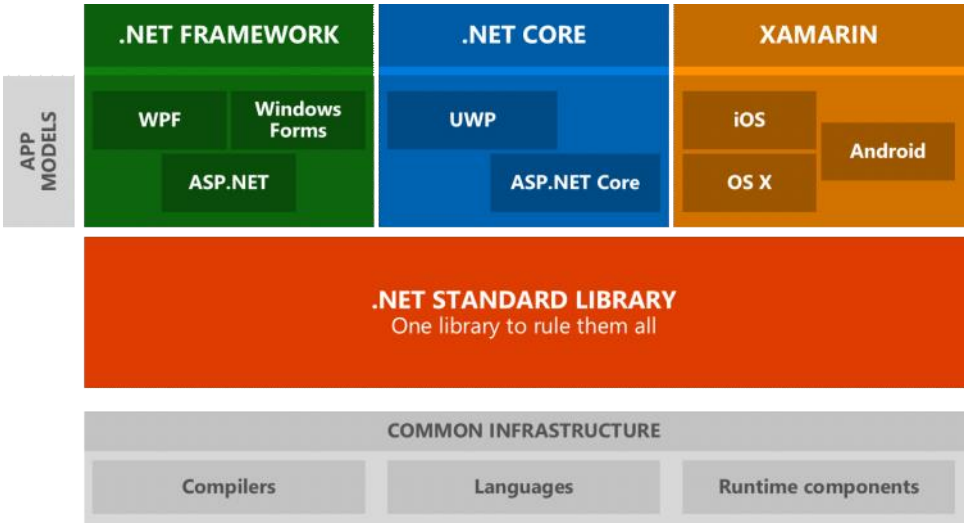
A primeira versão do .NET foi lançada no ano de 2002, sendo chamada de .NET Framework 1.0, ela era compatível apenas ao Sistema Operacional Windows, o que limitava o poder de desenvolvimento, porém a ideia da plataforma era permitir a utilização de várias linguagens numa plataforma única, podendo ser 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.



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

*Evolução do .NET

Seguindo a Evolução do .NET em 2016, foi lançado o .NET CORE que permitia o desenvolvimento em vários sistemas operacionais como



https://docs.microsoft.com/pt-br/dotnet/standard/library-guidance/cross-platform-targeting

C#

terça-feira, 14 de junho de 2022 21:23

O C# é uma linguagem de programação orientada a objetos e orientada a componentes, criada pela Microsoft e lançada no 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.
https://docs.microsoft.com/pt-br/dotnet/csharp/whats-new/csharp-version-history

*Add uma referência sobre C#

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.

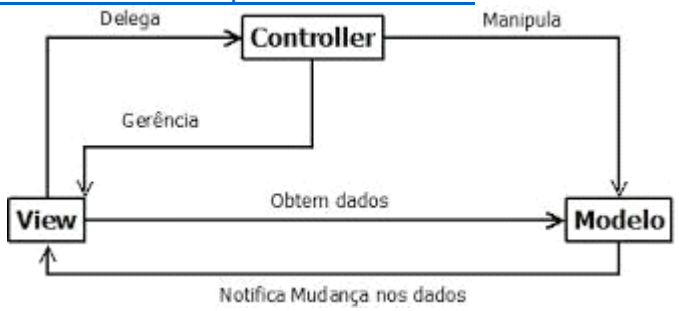
Arquitetura Model-View-Controller(MVC)

terça-feira, 7 de junho de 2022 15:18

Na fase de projeto, o ponto crucial é a escolha e o desenho da arquitetura da aplicação, por esse motivo, 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 paradigma contribuiu para a diminuição do acoplamento entre classes, auxiliando no reuso.

O MVC consiste na divisão 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).
https://conic-semesp.org.br/anais/files/2013/trabalho-1000014483.pdf
https://www.devmedia.com.br/introducao-ao-padrao-mvc/29308



Padrão MVC:

1. A priori, 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 efetuar a alteração apropriada;
2. Por sua vez, 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 mudar de estado. O modelo sabe o que o aplicativo quer fazer 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 da área retangular do display e é responsável por apresentar 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 realmente faz é receber instruções do controle e informações do modelo e então exibi-las. A visão também se comunica de volta com o modelo e com o controlador para reportar o seu estado.

Requisitos do sistema + Diagramas

terça-feira, 7 de junho de 2022 15:20

Requisitos em sua definição consiste na identificação documentada de uma propriedade ou comportamento que um produto deve atender. São a base para capturar e comunicar necessidades, gerenciar expectativas, priorizar e atribuir trabalho, verificar e validar o sistema (aceitação) e gerenciar o escopo 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>

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

- Requisitos Funcionais

Código	Nome Requisito	Descrição	Ator
RF-001	**Cadastramento de funcionários**	**Esta funcionalidade, deverá permitir o usuário (gerente) a realizar o cadastramento de funcionários, as ações que estarão disponíveis serão: criar, remover, alterar e consultar o cadastro de funcionários.**	**Gerente**
RF-002	**Cadastramento de clientes**	**Esta funcionalidade, deverá permitir o usuário (funcionário) a realizar 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 Gerente**
RF-003	**Cadastramento de produtos**	**Esta funcionalidade, deverá permitir o usuário (funcionário) a realizar o cadastramento de produtos, as ações que estarão disponíveis serão: criar, remover, alterar e consultar o cadastro de produtos.**	**Funcionário Gerente**
RF-004	**Pagamento**	**Esta funcionalidade, deverá permitir o usuário (funcionário) cobrar 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 Gerente**
RF-005	**Realizar o controle do pagamento do plano mensal**	**Esta funcionalidade, deverá permitir o usuário (funcionário) realizar 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.**	**Funcionário (apenas consultar) Gerente (consultar e alterar)**
RF-006	**Realizar agendamento de console**	**Esta funcionalidade, deverá permitir o usuário (funcionário) a realizar 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 Gerente**
RF-007	**Realizar o monitoram**	**Esta funcionalidade, deverá permitir o usuário (funcionário) a realizar o monitoramento do**	**Funcionário Gerente**

	ento do uso dos consoles	uso dos consoles (tempo de uso).	
RF-008	Realizar a locação de jogos	Esta funcionalidade, deverá permitir o usuário (funcionário) a realizar 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 Gerente
RF-009	Venda de produtos	Esta funcionalidade, deverá permitir o usuário (funcionário) a efetuar vendas de produtos: criar, remover, alterar e consultar pedidos.	Funcionário Gerente

- Requisitos Não-Funcionais

Código	Nome 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