Normalização do Banco de Dados para Desenvolvimento de Jogos

Esta apresentação explora a aplicação das Formas Normais (1FN, 2FN, 3FN) na modelagem de um banco de dados para desenvolvimento de jogos, garantindo integridade e eficiência.

Link da apresentação: https://youtu.be/e3pkJRV-uYw

Agenda

1 Introdução à Normalização

Entendimento do conceito e a importância da normalização em bancos de dados relacionais.

2 Primeira Forma Normal (1FN)

Detalhamento da 1FN, identificação de problemas e soluções.

3 Segunda Forma Normal (2FN)

Exploração da 2FN, dependências parciais e como resolvê-las.

4 Terceira Forma Normal (3FN)

Análise da 3FN, dependências transitivas e sua eliminação.

5 Visão Geral do Modelo Normalizado

Apresentação das entidades e seus atributos após a normalização.

Normalização do Banco de Dados para Desenvolvimento de Jogos

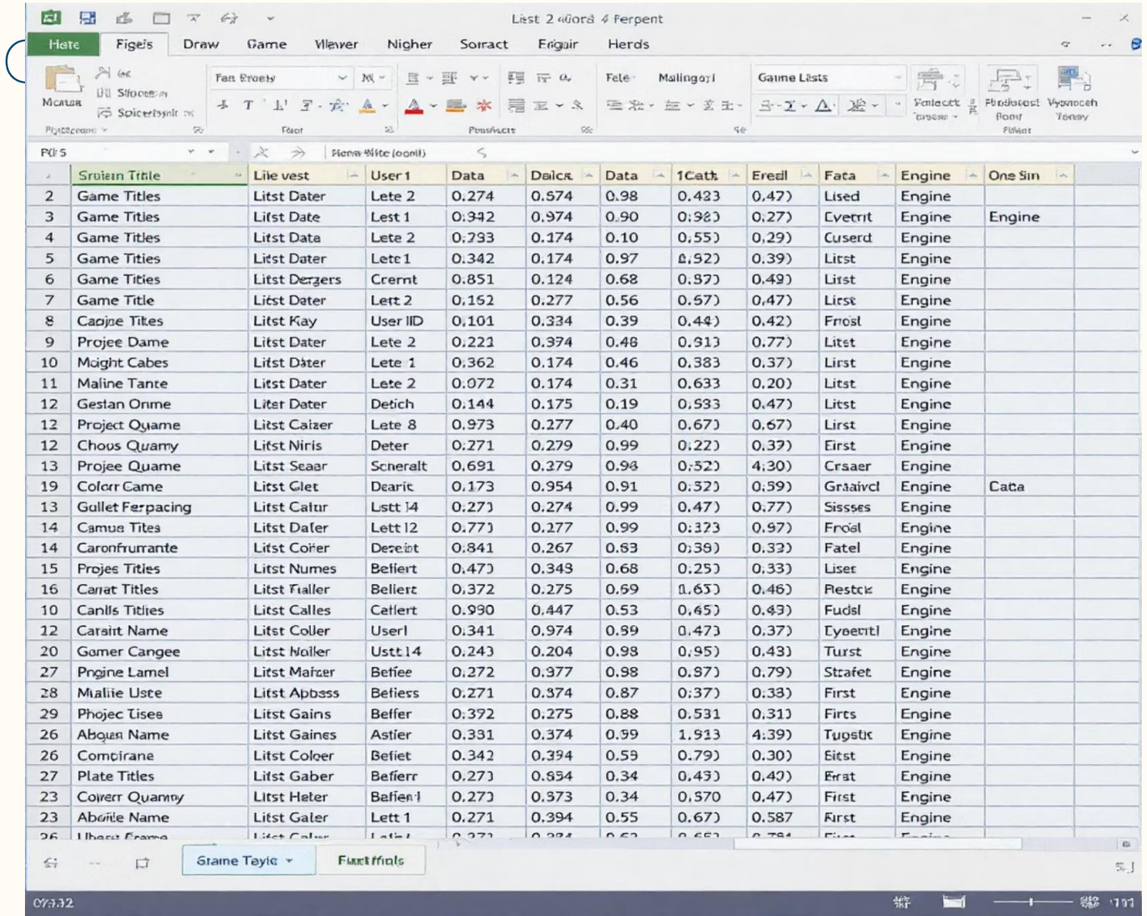
A **normalização** é um processo de organização dos dados em um banco de dados relacional. Seu principal objetivo é eliminar redundâncias e inconsistências, garantindo que cada informação seja armazenada de forma lógica e eficiente.
Ao aplicar as regras de normalização, dividimos grandes tabelas em tabelas menores e bem relacionadas. Isso reduz a duplicidade de dados, facilita a manutenção, melhora a integridade das informações e otimiza consultas e atualizações.

Problema Inicial: Planilhas com Dad

No estágio inicial, os dados de desenvolvimento de jogos estavam dispersos em planilhas, com informações de diversas entidades — Jogos, Projetos, Equipes, Usuários, Engines, Versões, Plataformas e Assets — todas entrelaçadas.

Essa abordagem levava a:

- Atributos **não atômicos**: Múltiplos valores (ex: várias plataformas) em uma única célula.
- Mistura de entidades: Diferentes tipos de informações (projeto e usuário) na mesma estrutura.
- Redundância e inconsistência: Dados repetidos e difíceis de manter atualizados.



Primeira Forma Normal (1FN)

A 1FN exige que todos os atributos em uma tabela sejam atômicos e que não existam grupos repetitivos de colunas. Ou seja, cada célula deve conter apenas um valor.

Problema Identificado

Campos com múltiplos valores, como uma lista de plataformas ou equipes associadas a um único projeto dentro de uma única célula.

Solução Aplicada

Separação em tabelas distintas, garantindo que cada atributo seja atômico. Por exemplo, plataformas são agora uma entidade separada.

Exemplo

Criação de tabelas para **Jogo**, **Projeto**, **Usuário**, **Equipe**, **Engine**, **Plataforma**, **Asset**. Relacionamentos um-para-muitos são estabelecidos com chaves estrangeiras.

Segunda Forma Normal (2FN)

A 2FN baseia-se na 1FN e exige que atributos não-chave em uma tabela com chave composta dependam inteiramente de toda a chave composta, não apenas de parte dela.

Problema Identificado

Tabelas com chaves compostas (ex: Projeto + Usuário) onde atributos não-chave dependiam apenas parcialmente da chave. Por exemplo, o nome do usuário dependia somente do ID do usuário, não da combinação completa.

Solução Aplicada

Criação de entidades independentes e tabelas associativas. Entidades como **Usuário** e **Jogo** são totalmente separadas, com chaves primárias únicas.

Exemplo

Introdução de tabelas como **EquipeProjeto** (ID_usuario, ID_projeto, função) e **JogoPlataforma** (ID_jogo, ID_plataforma, data_lançamento) para gerenciar relacionamentos muitos-para-muitos.

Terceira Forma Normal (3FN)

A 3FN baseia-se na 2FN e proíbe dependências transitivas, ou seja, atributos não-chave que dependem de outros atributos não-chave, em vez da chave primária.

Problema Identificado

Dependências transitivas, como o nome da engine dependendo do ID da engine (que é um FK) em uma tabela de jogo, em vez de depender diretamente da PK do jogo.

Solução Aplicada

Criação de entidades normalizadas para cada conceito. Cada atributo não-chave deve depender **apenas** da chave primária da tabela.

Exemplo

Tabelas dedicadas para **Engine** (ID, nome, versão) e **Plataforma** (ID, nome). Remoção de campos derivados e dados duplicados, como 'data de lançamento' repetida.

Modelo de Dados Normalizado: Entidades Chave

O processo de normalização resultou em um modelo de dados limpo e eficiente, eliminando redundâncias e melhorando a integridade dos dados. As principais entidades são:

Usuário

ID, nome, e-mail, função/perfil

Jogo

ID, título, gênero, data_criação

Projeto

ID, status, data_início, ID_jogo

Engine

ID, nome, versão

Plataforma

ID, nome (ex: Steam, PS5, Android)

Asset

ID, tipo (imagem, áudio, script), nome, ID_projeto

UsuárioExterno

ID, nome, contato, usa_engine_proprietária

Modelo de Dados Normalizado: Tabelas Associativas

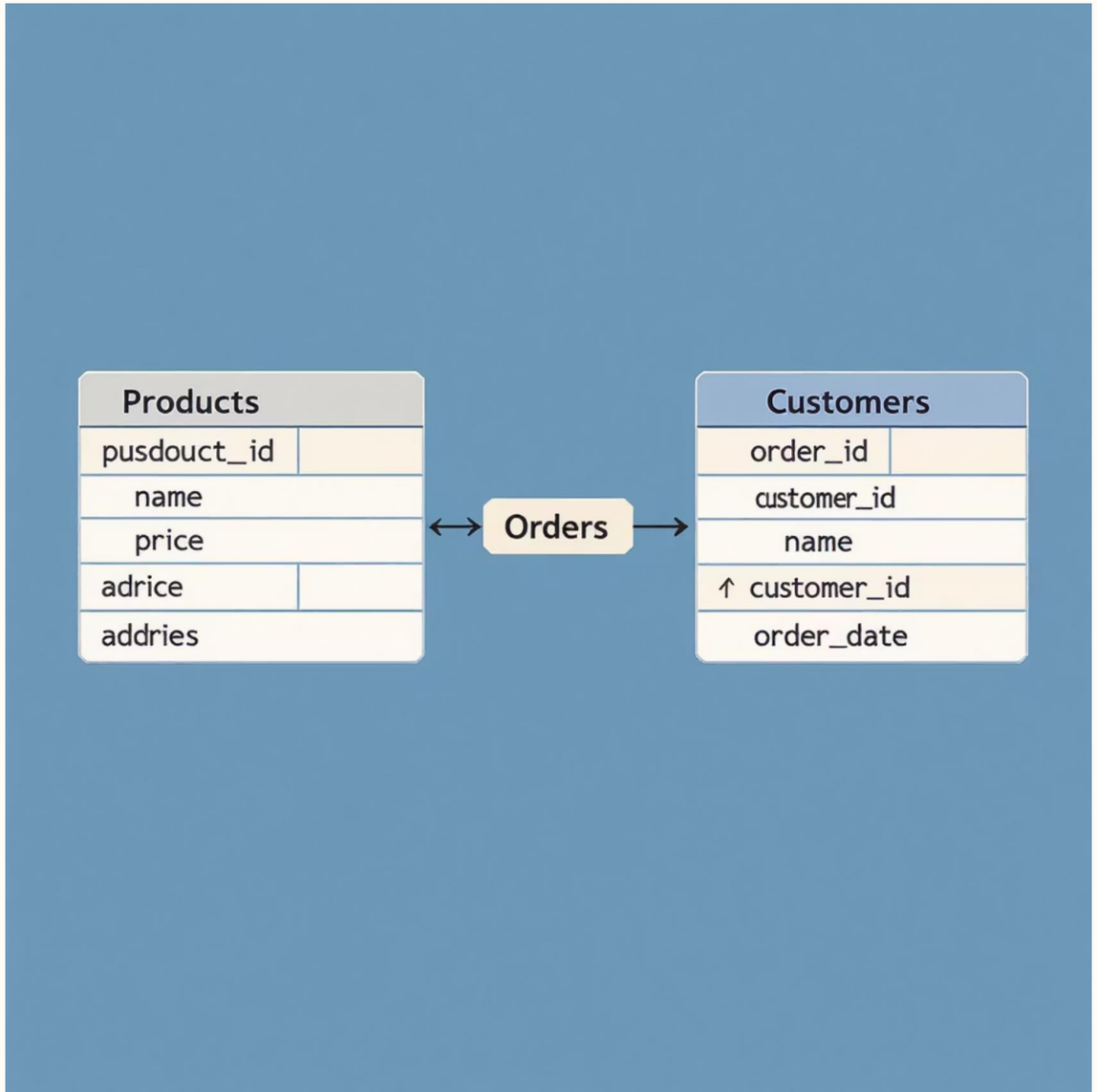
Para gerenciar relacionamentos muitos-para-muitos e adicionar atributos específicos a essas associações, foram criadas as seguintes tabelas:

EquipeProjeto

Registra a participação de usuários em projetos. Contém: ID_usuario, ID_projeto, função e progresso.

JogoPlataforma

Associa jogos às plataformas em que são lançados. Contém: ID_jogo, ID_plataforma e data_lançamento.



Próximos Passos e Benefícios

Com o banco de dados normalizado, podemos esperar melhorias significativas na eficiência e confiabilidade do sistema.

- **Integridade de Dados Aprimorada:** Redução drástica de inconsistências e anomalias.
- **Menos Redundância:** Armazenamento de dados otimizado, economizando espaço e facilitando a manutenção.
- **Consultas Mais Rápidas:** Estrutura organizada permite buscas e junções de dados mais eficientes.
- **Manutenção Simplificada:** Atualizações e modificações de dados são mais fáceis e menos propensas a erros.
- **Escalabilidade Aumentada:** O modelo pode crescer e se adaptar a novas necessidades sem grandes reestruturações.

Este modelo serve como base robusta para o desenvolvimento de sistemas mais complexos e eficientes para a gestão de jogos.

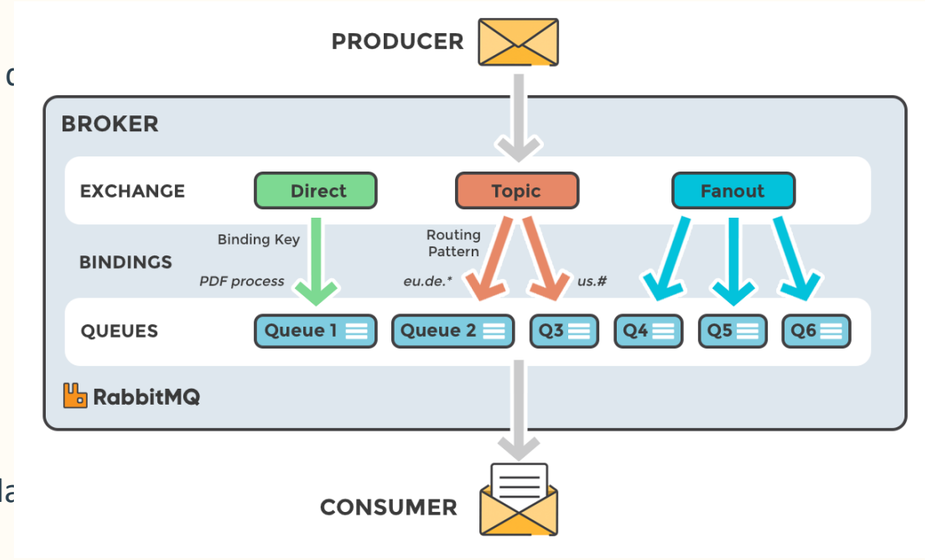
Mensageria no desenvolvimento dos jogos: Papel do Rabb

•**O que é mensageria?**
Técnica que permite a comunicação assíncrona entre diferentes partes de um sistema, usando filas de mensagens para enviar informações entre serviços, aplicações ou módulo

•**Por que usar mensageria em jogos?**
•Processamento de eventos em tempo real (ex: notificações de conquistas, chat entre jogadores, ranking d
•Desacoplamento entre sistemas (ex: separar lógica de pagamentos do servidor de partidas)
•Escalabilidade e resiliência (falhas não travam o sistema principal)

•**O que é RabbitMQ?**
•Um dos sistemas de mensageria mais usados no mundo.
•Baseado no protocolo AMQP.
•Permite criar filas, exchanges e roteamentos flexíveis de mensagens.

•**Exemplo prático em jogos:**
•Cada vez que um jogador termina uma partida, o servidor publica uma mensagem com o resultado na fila
•Outro serviço lê essa fila, processa a mensagem e atualiza o ranking no banco de dados normalizado.



Alan Beltrão Costa
Danielly Reis Dos Santos
Eduardo Figueiredo
Guilherme Melo de Lima
Letícia Moreira Guimarães
Victor Hugo Batista Tavares

GAMECOREDW

1. Cenário

GameCoreDW é uma empresa de médio porte focada no desenvolvimento e distribuição de jogos digitais. A empresa atua tanto na criação de jogos quanto na oferta de uma engine de desenvolvimento própria, que é utilizada por estúdios parceiros e desenvolvedores independentes.

As equipes são compostas por desenvolvedores de diferentes perfis: programadores, artistas, designers e engenheiros de som. Cada equipe trabalha em um ou mais projetos simultaneamente. Um desenvolvedor pode participar de vários projetos, e o progresso dessas participações precisa ser registrado e consultado com facilidade.

A GameCoreDW deseja implantar um novo sistema de gerenciamento centralizado, com foco em organizar e acompanhar:

- Seus jogos desenvolvidos
- As equipes de desenvolvimento são compostas por programadores, designers, roteiristas.
- Os projetos em andamento
- As versões das engines utilizadas nos jogos,
- A publicação e distribuição dos jogos em plataformas como Steam, consoles, mobile;
- O relacionamento com desenvolvedores externos que utilizam sua engine.

2. Visão Geral do Sistema

Descrição do problema ou necessidade que o sistema irá resolver

Atualmente, os dados dos jogos, projetos, desenvolvedores e assets (como imagens, áudios, modelos 3D e scripts) estão espalhados em planilhas e documentos manuais, dificultando o acompanhamento, a rastreabilidade e a colaboração entre equipes.

Além disso, a empresa trabalha com diferentes Engines (como Unity, Unreal Engine, Godot, Construc 3), e precisa manter o controle de quais projetos usam qual engine, em qual versão.

Objetivos do sistema

deseja implantar um novo sistema de gerenciamento centralizado, com foco em organizar e acompanhar:

- Seus jogos desenvolvidos,
- As equipes de desenvolvimento compostas por programadores, designers, roteiristas,
- Os projetos em andamento,
- As versões das engines utilizadas nos jogos,
- A publicação e distribuição dos jogos em plataformas como Steam, consoles, mobile,
- O relacionamento com desenvolvedores externos que utilizam sua engine.

Usuários do sistema e seus perfis

programadores, artistas, designers e engenheiros de som.

Contexto de uso dos sistemas

A GameCoreDW é uma empresa dedicada ao desenvolvimento de jogos digitais e à criação de uma engine própria para uso interno e por parceiros. Com o crescimento da empresa e a diversificação de seus projetos, tornou-se necessário implementar um sistema de gerenciamento centralizado para organizar informações críticas do ciclo de desenvolvimento. O sistema será utilizado principalmente por meio de uma interface web, com níveis de acesso diferenciados. Ele precisa permitir o cadastro, atualização, consulta e geração de relatórios sobre as entidades e seus relacionamentos, além de fornecer mecanismos para rastrear o histórico de versões dos jogos e das engines.

3. Requisitos Funcionais

Descrição detalhada das funcionalidades que o sistema deve ter casos de uso e exemplos de como as funcionalidades serão utilizadas informações sobre os dados que serão processados pelo sistema regras de negócio

Cadastro e Gerenciamento dos Jogos

Deve permitir o cadastro, edição e exclusão de jogos.

Login dos Usuários

Deve fazer o controle de acessos dos usuários permitindo funções para cada tipo de usuário.

Cadastro de Usuários

Deve permitir o cadastro de novos usuários desenvolvedores e jogadores.

Controle de Equipe

Deve permitir vincular um ou mais desenvolvedores a diferentes projetos.

Cadastro do Projeto (Andamento do desenvolvimento)

Deve registrar os projetos em andamento e associá-los aos respectivos jogos.

Deve armazenar e gerenciar os assets utilizados em cada projeto (imagens, áudios, modelos 3D, scripts etc.).

Deve registrar o progresso individual dos desenvolvedores em cada projeto.

Cadastro de Engines e Versões

Deve permitir registrar e consultar as versões das engines utilizadas em cada jogo.

Controle de Fornecimento

Deve manter o histórico de publicação dos jogos, indicando as plataformas e datas de lançamento

Controle de Qualidade

Deve permitir o gerenciamento do relacionamento com desenvolvedores externos que utilizam a engine proprietária.

4. Requisitos Não Funcionais

Desempenho (velocidade, tempo de resposta)
Segurança (autenticação, autorização, proteção de dados)
Usabilidade (facilidade de uso, interface intuitiva)
Confiabilidade (disponibilidade, tolerância a falhas)
Manutenibilidade (facilidade de atualização e correção)
Portabilidade (compatibilidade com diferentes plataformas)

Desempenho

Deve carregar imagens no máximo em 1 segundo.
Deve carregar áudios no máximo em 2 segundos.
Deve carregar vídeos no máximo em 3 segundos.
Deve estar disponível 24 horas por dia, 7 dias por semana.

Segurança

Deve permitir acesso multiusuário com controle de permissões por perfil.
Deve exigir que cada usuário faça seu próprio login para utilizar o sistema.
Deve permitir que somente usuários administradores podem cadastrar novos itens.

Usabilidade

Deve ser compatível como banco de imagens e vídeo.
Deve possuir interface intuitiva.

Confiabilidade

Deve fazer o salvamento automático de jogos em desenvolvimento.

Manutenibilidade

Deve possuir um campo para feedback de usuários/ testers e jogadores.
Deve guardar o histórico de versões do jogo em desenvolvimento.
Deve registrar logs de acesso e alterações feitas por usuários no banco de dados.

Portabilidade

Deve funcionar em todos os navegadores.
Deve ser adaptado para dispositivos móveis.

5. Critérios de Aceitação

Definição de como será avaliado se o sistema atende aos requisitos indicadores de sucesso do sistema.

Funcionalidades Essenciais Implementadas:

Sucesso: Todas as funcionalidades listadas nos Requisitos Funcionais (Cadastro de Jogos, Login, Gestão de Equipes, Projetos, Engines, Publicação, Controle de Qualidade) foram implementadas e funcionam conforme o esperado.

Indicador: 100% dos casos de uso críticos executados com sucesso em testes de aceitação do usuário (UAT).

Melhora na Rastreabilidade e Colaboração:

Sucesso: Redução significativa do tempo gasto na busca de informações e aumento da comunicação entre as equipes.

Indicador: Pesquisas de satisfação da equipe mostram um aumento de 80% na percepção de organização e colaboração. Tempo médio para encontrar informações sobre um projeto/asset reduzido em 50%.

Desempenho da Aplicação:

Sucesso: O sistema atende aos tempos de carregamento especificados para imagens, áudios e vídeos.

Indicador: Testes de desempenho automatizados confirmam que o carregamento de imagens está abaixo de 1s, áudios abaixo de 2s e vídeos abaixo de 3s em 95% das requisições.

Segurança e Acesso Controlado:

Sucesso: O sistema implementa efetivamente o controle de acesso por perfil e exige autenticação para todos os usuários.

Indicador: Auditoria de segurança não encontra vulnerabilidades críticas relacionadas ao controle de acesso. Tentativas de acesso não autorizadas são bloqueadas.

Usabilidade e Experiência do Usuário (UX):

Sucesso: A interface é intuitiva e fácil de usar para todos os perfis de usuário, resultando em menor necessidade de treinamento.

Indicador: Usuários conseguem completar tarefas essenciais sem assistência em 90% das vezes em testes de usabilidade. Taxa de erro dos usuários menor que 5%.

Confiabilidade e Disponibilidade:

Sucesso: O sistema demonstra alta disponibilidade e capacidade de recuperação, com o salvamento automático funcionando como esperado.

Indicador: Uptime do sistema de 99,9% durante o período de teste. Nenhum dado é perdido devido a falhas inesperadas em 100% dos testes de interrupção simulada.

Manutenibilidade e Evolução:

Sucesso: O sistema é facilmente atualizável e corrigível, com um bom registro de feedback e histórico de versões.

Indicador: O tempo médio para resolução de bugs críticos é de no máximo 24 horas. O sistema de feedback registra e processa 95% das sugestões e erros reportados.

Portabilidade:

Sucesso: O sistema funciona perfeitamente em diferentes navegadores e dispositivos móveis.

Indicador: Testes de compatibilidade com navegadores e dispositivos móveis não identificam problemas críticos de layout ou funcionalidade.

6. Requisitos Adicionais

Quaisquer outros requisitos relevantes que não se encaixam nas categorias anteriores. Restrições com orçamento, prazo, tecnologias.

7. Anexos

Ferramentas e Técnicas:

Entrevistas: Conversas com os usuários para entender suas necessidades e expectativas.

Questionários: Formulários com perguntas estruturadas para coletar informações de muitos usuários.

Workshops: Reuniões com os usuários para discutir e validar os requisitos.

Prototipagem: Criação de versões simplificadas do sistema para testar e validar as funcionalidades.

Casos de uso: Descrição detalhada de como os usuários interagem com o sistema.

Histórias de usuário: Descrição dos requisitos em termos de cenários de uso.

Importância do Levantamento de Requisitos:

Redução de riscos: Identificar problemas e necessidades desde o início do projeto evita retrabalho e atrasos.

Melhora da qualidade: Um sistema que atende às necessidades dos usuários tem maior chance de sucesso.

Aumento da satisfação do cliente: Clientes satisfeitos com o sistema são mais propensos a utilizá-lo e recomendá-lo.

Redução de custos: Ao evitar erros e retrabalho, o levantamento de requisitos contribui para a redução de custos do projeto.

Observações:

É importante adaptar o formulário e as técnicas de coleta de requisitos ao contexto de cada projeto. O processo de levantamento de requisitos deve ser iterativo e envolver os usuários em todas as etapas. É fundamental validar os requisitos com os usuários para garantir que eles atendam às suas necessidades.

Alan Beltrão Costa
Danielly Reis Dos Santos
Eduardo Figueiredo
Guilherme Melo de Lima
Letícia Moreira Guimarães
Victor Hugo Batista Tavares

GAMECOREDW

1. Cenário

GameCoreDW é uma empresa de médio porte focada no desenvolvimento e distribuição de jogos digitais. A empresa atua tanto na criação de jogos quanto na oferta de uma engine de desenvolvimento própria, que é utilizada por estúdios parceiros e desenvolvedores independentes.

As equipes são compostas por desenvolvedores de diferentes perfis: programadores, artistas, designers e engenheiros de som. Cada equipe trabalha em um ou mais projetos simultaneamente. Um desenvolvedor pode participar de vários projetos, e o progresso dessas participações precisa ser registrado e consultado com facilidade.

A GameCoreDW deseja implantar um novo sistema de gerenciamento centralizado, com foco em organizar e acompanhar:

- Seus jogos desenvolvidos
- As equipes de desenvolvimento são compostas por programadores, designers, roteiristas.
- Os projetos em andamento
- As versões das engines utilizadas nos jogos,
- A publicação e distribuição dos jogos em plataformas como Steam, consoles, mobile;
- O relacionamento com desenvolvedores externos que utilizam sua engine.

2. Visão Geral do Sistema

Descrição do problema ou necessidade que o sistema irá resolver

Atualmente, os dados dos jogos, projetos, desenvolvedores e assets (como imagens, áudios, modelos 3D e scripts) estão espalhados em planilhas e documentos manuais, dificultando o acompanhamento, a rastreabilidade e a colaboração entre equipes.

Além disso, a empresa trabalha com diferentes Engines (como Unity, Unreal Engine, Godot, Construc 3), e precisa manter o controle de quais projetos usam qual engine, em qual versão.

Objetivos do sistema

deseja implantar um novo sistema de gerenciamento centralizado, com foco em organizar e acompanhar:

- Seus jogos desenvolvidos,
- As equipes de desenvolvimento compostas por programadores, designers, roteiristas,
- Os projetos em andamento,
- As versões das engines utilizadas nos jogos,
- A publicação e distribuição dos jogos em plataformas como Steam, consoles, mobile,
- O relacionamento com desenvolvedores externos que utilizam sua engine.

Usuários do sistema e seus perfis

programadores, artistas, designers e engenheiros de som.

Contexto de uso dos sistemas

A GameCoreDW é uma empresa dedicada ao desenvolvimento de jogos digitais e à criação de uma engine própria para uso interno e por parceiros. Com o crescimento da empresa e a diversificação de seus projetos, tornou -se necessário implementar um sistema de gerenciamento centralizado para organizar informações críticas do ciclo de desenvolvimento. O sistema será utilizado principalmente por meio de uma interface web, com níveis de acesso diferenciados. Ele precisa permitir o cadastro, atualização, consulta e geração de relatórios sobre as entidades e seus relacionamentos, além de fornecer mecanismos para rastrear o histórico de versões dos jogos e das engines.

3. Requisitos Funcionais

Descrição detalhada das funcionalidades que o sistema deve ter casos de uso e exemplos de como as funcionalidades serão utilizadas informações sobre os dados que serão processados pelo sistema regras de negócio

Cadastro e Gerenciamento dos Jogos

Deve permitir o cadastro, edição e exclusão de jogos.

Login dos Usuários

Deve fazer o controle de acessos dos usuários permitindo funções para cada tipo de usuário.

Cadastro de Usuários

Deve permitir o cadastro de novos usuários desenvolvedores e jogadores.

Controle de Equipe

Deve permitir vincular um ou mais desenvolvedores a diferentes projetos.

Cadastro do Projeto (Andamento do desenvolvimento)

Deve registrar os projetos em andamento e associá-los aos respectivos jogos.

Deve armazenar e gerenciar os assets utilizados em cada projeto (imagens, áudios, modelos 3D, scripts etc.).

Deve registrar o progresso individual dos desenvolvedores em cada projeto.

Cadastro de Engines e Versões

Deve permitir registrar e consultar as versões das engines utilizadas em cada jogo.

Controle de Fornecimento

Deve manter o histórico de publicação dos jogos, indicando as plataformas e datas de lançamento

Controle de Qualidade

Deve permitir o gerenciamento do relacionamento com desenvolvedores externos que utilizam a engine proprietária.

4. Requisitos Não Funcionais

Desempenho (velocidade, tempo de resposta)
Segurança (autenticação, autorização, proteção de dados)
Usabilidade (facilidade de uso, interface intuitiva)
Confiabilidade (disponibilidade, tolerância a falhas)
Manutenibilidade (facilidade de atualização e correção)
Portabilidade (compatibilidade com diferentes plataformas)

Desempenho

Deve carregar imagens no máximo em 1 segundo.
Deve carregar áudios no máximo em 2 segundos.
Deve carregar vídeos no máximo em 3 segundos.
Deve estar disponível 24 horas por dia, 7 dias por semana.

Segurança

Deve permitir acesso multiusuário com controle de permissões por perfil.
Deve exigir que cada usuário faça seu próprio login para utilizar o sistema.
Deve permitir que somente usuários administradores podem cadastrar novos itens.

Usabilidade

Deve ser compatível como banco de imagens e vídeo.
Deve possuir interface intuitiva.

Confiabilidade

Deve fazer o salvamento automático de jogos em desenvolvimento.

Manutenibilidade

Deve possuir um campo para feedback de usuários/ testers e jogadores.
Deve guardar o histórico de versões do jogo em desenvolvimento.
Deve registrar logs de acesso e alterações feitas por usuários no banco de dados.

Portabilidade

Deve funcionar em todos os navegadores.
Deve ser adaptado para dispositivos móveis.

5. Critérios de Aceitação

Definição de como será avaliado se o sistema atende aos requisitos indicadores de sucesso do sistema.

Funcionalidades Essenciais Implementadas:

Sucesso: Todas as funcionalidades listadas nos Requisitos Funcionais (Cadastro de Jogos, Login, Gestão de Equipes, Projetos, Engines, Publicação, Controle de Qualidade) foram implementadas e funcionam conforme o esperado.

Indicador: 100% dos casos de uso críticos executados com sucesso em testes de aceitação do usuário (UAT).

Melhora na Rastreabilidade e Colaboração:

Sucesso: Redução significativa do tempo gasto na busca de informações e aumento da comunicação entre as equipes.

Indicador: Pesquisas de satisfação da equipe mostram um aumento de 80% na percepção de organização e colaboração. Tempo médio para encontrar informações sobre um projeto/asset reduzido em 50%.

Desempenho da Aplicação:

Sucesso: O sistema atende aos tempos de carregamento especificados para imagens, áudios e vídeos.

Indicador: Testes de desempenho automatizados confirmam que o carregamento de imagens está abaixo de 1s, áudios abaixo de 2s e vídeos abaixo de 3s em 95% das requisições.

Segurança e Acesso Controlado:

Sucesso: O sistema implementa efetivamente o controle de acesso por perfil e exige autenticação para todos os usuários.

Indicador: Auditoria de segurança não encontra vulnerabilidades críticas relacionadas ao controle de acesso. Tentativas de acesso não autorizadas são bloqueadas.

Usabilidade e Experiência do Usuário (UX):

Sucesso: A interface é intuitiva e fácil de usar para todos os perfis de usuário, resultando em menor necessidade de treinamento.

Indicador: Usuários conseguem completar tarefas essenciais sem assistência em 90% das vezes em testes de usabilidade. Taxa de erro dos usuários menor que 5%.

Confiabilidade e Disponibilidade:

Sucesso: O sistema demonstra alta disponibilidade e capacidade de recuperação, com o salvamento automático funcionando como esperado.

Indicador: Uptime do sistema de 99,9% durante o período de teste. Nenhum dado é perdido devido a falhas inesperadas em 100% dos testes de interrupção simulada.

Manutenibilidade e Evolução:

Sucesso: O sistema é facilmente atualizável e corrigível, com um bom registro de feedback e histórico de versões.

Indicador: O tempo médio para resolução de bugs críticos é de no máximo 24 horas. O sistema de feedback registra e processa 95% das sugestões e erros reportados.

Portabilidade:

Sucesso: O sistema funciona perfeitamente em diferentes navegadores e dispositivos móveis.

Indicador: Testes de compatibilidade com navegadores e dispositivos móveis não identificam problemas críticos de layout ou funcionalidade.

6. Requisitos Adicionais

Quaisquer outros requisitos relevantes que não se encaixam nas categorias anteriores. Restrições com orçamento, prazo, tecnologias.

7. Diagrama de Entidade - Relacionamento

Buscando atender os requisitos levantados foi elaboroado o seguinte diagrama.

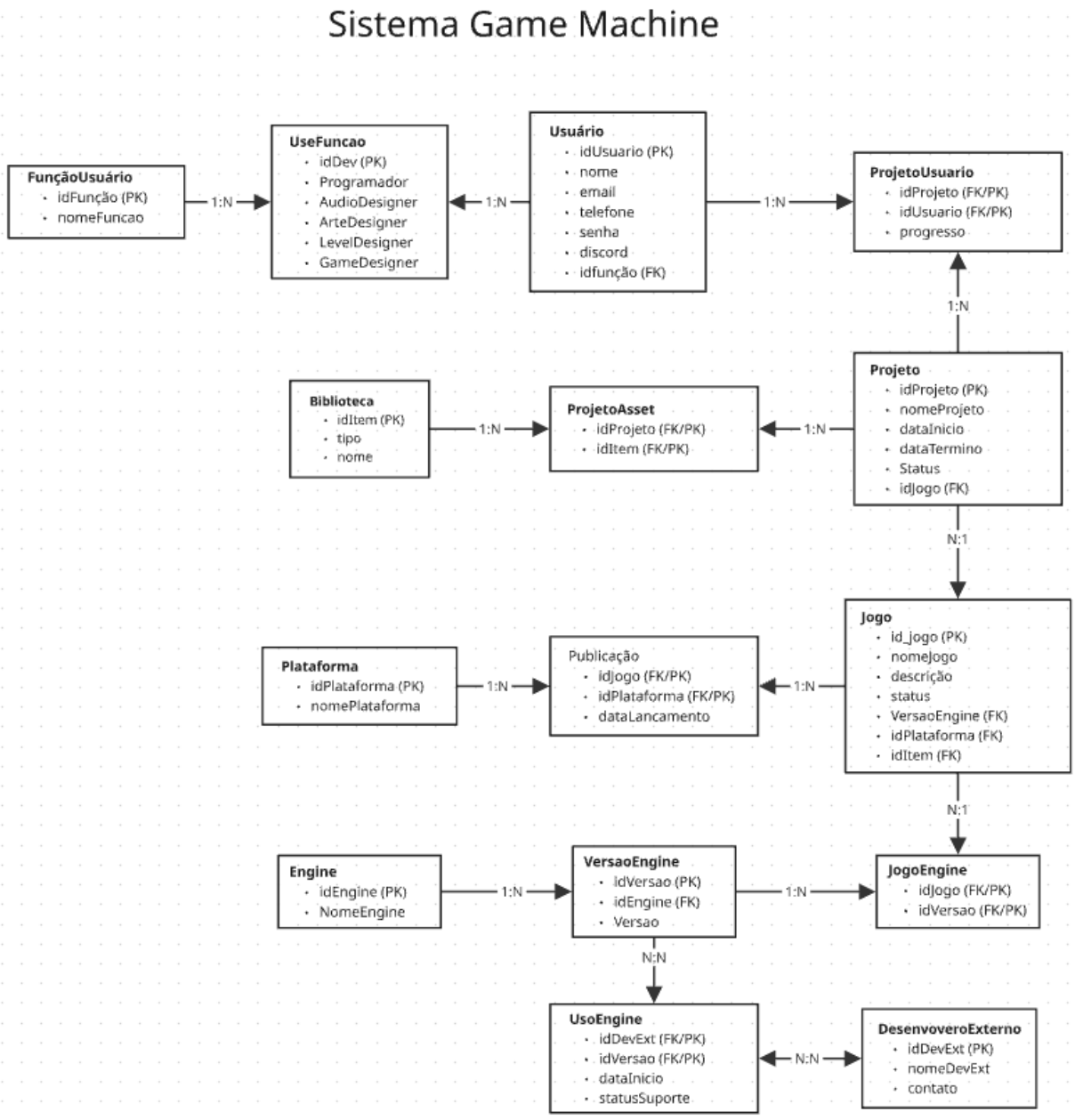


Figura 1 - Diagrama ER do sistema Game Machine

RESUMO DE RELACIONAMENTOS (cardinalidade simplificada)

Conversas com os usuários para entender suas necessidades

Entidade A	Relacionamento	Entidade B	Cardinalidade
Usuário	usa	FunçãoUsuario	N:N
Usuário	participa	Projeto	N:N (com progresso)
Projeto	utiliza	Biblioteca (Assets)	N:N
Projeto	pertence	Jogo	N:1
Jogo	roda em	Plataforma	N:1
Jogo	usa	VersaoEngine	N:N
Jogo	é publicado em	Plataforma	N:N
Engine	possui	VersaoEngine	1:N
VersaoEngine	é usada por	Jogo	N:N
VersaoEngine	é usada por	DesenvolvedorExterno	N:N

Tabela 1 - Resumo de Relacionamentos

8. Anexo B

Ferramentas e Técnicas:

Entrevistas: Conversas com os usuários para entender suas necessidades e expectativas.

Questionários: Formulários com perguntas estruturadas para coletar informações de muitos usuários.

Workshops: Reuniões com os usuários para discutir e validar os requisitos.

Prototipagem: Criação de versões simplificadas do sistema para testar e validar as funcionalidades.

Casos de uso: Descrição detalhada de como os usuários interagem com o sistema.

Histórias de usuário: Descrição dos requisitos em termos de cenários de uso.

Observações:

É importante adaptar o formulário e as técnicas de coleta de requisitos ao contexto de cada projeto. O processo de levantamento de requisitos deve ser iterativo e envolver os usuários em todas as etapas. É fundamental validar os requisitos com os usuários para garantir que eles atendam às suas necessidades.