

**Ecossistema Ânima de Educação**

Projeto avaliativo de Modelagem de Software

***Meio de garantir que todos os jovens e uma substancial proporção dos adultos, homens e mulheres estejam alfabetizados e tenham adquirido o conhecimento básico de matemática.***

**Alice Martins Bahiense Bezerra Bauler**

**Erick Barros Ferreira Gomes**

**Marcela Tourinho Machado Barreto**

**Marcilio Batista de Oliveira Filho**

**Marcus Vinicius dos Santos**

**Maria Clara Daltro Arruda**

**Rodson de Souza Silva Junior**

**Orientadores: Wellington Lacerda e Ana Cleyge**

**13/06/2024**

**Salvador - BA**

**2024**

Sumário (Rascunho)

1.Introdução ..................................................................................................

1.1 Contexto e Objetivos ...................................................................  
 1.2 Relevância do Estudo ..................................................................

3. Processo de desenvolvimento do Software ..............................................   
 3.1 Modelo de processo de desenvolvimento....................................   
 3.2 Embasamento histórico ..............................................................

4.Engenharia de requisitos ...........................................................................

4.1 Viabilidade do sistema computacional ...........................................

4.2 Levantamentos dos requisitos ........................................................

4.3 Análise e negociação de requisitos ...............................................

4.4 Especificar os requisitos do sistema computacional .....................

4.5 Validações dos requisitos ..............................................................

5. Arquitetura de software ..............................................................................

5.1 Tipo de arquitetura de software .....................................................

5.2 Embasamentos histórico ...............................................................

6. Conclusão ...................................................................................................  
 6.1 Resultados e Discussão ...............................................................  
 6.2 Resultados Quantitativos .............................................................  
 6.3 Contribuições e Limitações ...........................................................

7. Referencias ................................................................................................

Introdução

**1.1 Contexto e Objetivos**

O objetivo de desenvolvimento da ONU escolhido pela equipe foi o 4.6: "Meio de garantir que todos os jovens e uma substancial proporção dos adultos, homens e mulheres estejam alfabetizados e tenham adquirido o conhecimento básico de matemática. ". Este assunto e de extrema relevância para a sociedade e representa um desafio significativo em muitas partes do mundo, impactando negativamente o desenvolvimento socioeconômico e a qualidade de vida desses indivíduos.

Por conceito, o analfabetismo se refere à falta de habilidade de identificar, compreender, interpretar e criar enunciados a partir de materiais escritos. Contudo, o analfabetismo abrange diversas facetas como: O Analfabetismo Funcional, trata-se da incapacidade de compreender ideias explícitas e implícitas em um texto e emitir um julgamento crítico sobre elas; O Analfabetismo Digital, está relacionada à falta de conhecimentos necessários para interagir com tecnologias modernas, como a Internet, celulares e computadores; E O analfabetismo matemático que se refere à falta de habilidades básicas em matemática, que vão além da simples leitura e escrita de números até a aplicação logica em problemas operacionais e cotidianos.

Esta questão é complexa e diversificada, sendo influenciada por vários fatores, como a pobreza, a falta de infraestruturas educativas, a discriminação de gênero, os conflitos armados e a ausência de programas eficazes de alfabetização. Superar essa invisibilidade é essencial para promover um letramento mais amplo e igualitário e qualidade, fundamental para enfrentar os desafios do mundo atual. Em resumo, a garantia de alfabetização e conhecimento básico de matemática requer esforços conjuntos de governos, instituições educacionais e sociedade como um todo. Um investimento essencial para o progresso individual e coletivo.

Sob essa ótica, o objetivo deste artigo é apresentar o desenvolvimento de um recurso que pode auxiliar que todos os jovens e uma proporção significativa dos adultos estejam alfabetizados e possuam conhecimentos satisfatórios em matemática. O objetivo proposto de erradicar o analfabetismo até a virada do milênio não foi alcançado. Consequentemente, é necessário continuar trabalhando em sua concretização e analisar os fatores que fizeram e os que não o fizeram, a fim de criar recursos para progredir gradualmente e por fim alcançar esse objetivo.

**1.2 Relevância do Estudo**

Em 2022, a taxa de analfabetismo no Brasil era de 7% para a população com 15 anos ou mais. No entanto, houve uma evolução positiva, já que em 2010, a porcentagem de analfabetos era maior, chegando a 9,6%1. Essa melhoria reflete os esforços contínuos para promover a educação e a alfabetização no país. Vale ressaltar que o Nordeste continua sendo a região com a maior proporção de analfabetos, com o dobro da média nacional. Essas foram algumas estatísticas divulgadas pelo Instituto Brasileiro de Geografia e Estatística - IBGE.

A partir da análise do sistema brasileiro de educação em que ( dado ) Mesmo quando há acesso à educação, a qualidade do ensino varia significativamente. Professores mal remunerados, falta de treinamento adequado e currículos desatualizados podem prejudicar a aprendizagem efetiva. Desigualdades socioeconômicas afetam o acesso à educação. Mesmo quando as pessoas são alfabetizadas, o analfabetismo funcional ainda é um problema. Algumas culturas e normas sociais desvalorizam a educação, especialmente para mulheres. Alguns indivíduos simplesmente não têm interesse em aprender ou não veem o valor da educação formal. Livros didáticos, materiais de ensino e acesso à internet são essenciais para a aprendizagem. A falta desses recursos dificulta o processo educacional. A matemática é frequentemente considerada difícil e abstrata. Ensinar conceitos matemáticos de maneira acessível e prática é um desafio constante.

A modelagem de software pode desempenhar um papel significativo na promoção da alfabetização universal, os softwares educacionais podem ser uma ferramenta valiosa para a resolução desse problema. Através da modelagem, é possível criar ambientes de aprendizado personalizados, considerando o ritmo de cada estudante e suas habilidades específicas. A modelagem permite a criação de interfaces interativas e visuais que tornam o processo de alfabetização mais instigante para os alunos. Elementos como animações, jogos e simulações podem ajudar os alunos a compreenderem conceitos de forma mais eficaz. Além disso, essa abordagem lúdica torna o processo de alfabetização mais divertido e motivador para os alunos. Softwares podem ser projetados para atender a diferentes perfis de alunos, incluindo aqueles com necessidades especiais. Recursos como leitores de tela, legendas e opções de contraste beneficiam a inclusão. A alfabetização digital é essencial na sociedade atual. Através do uso de softwares, os alunos também desenvolvem habilidades tecnológicas, preparando-se para o mundo digital e contornando o analfabetismo digital. Softwares podem facilitar a colaboração entre alunos, permitindo que eles compartilhem textos, ideias e projetos. Isso promove a construção coletiva do conhecimento. Softwares educacionais bem projetados despertam a curiosidade dos alunos. Eles podem explorar diferentes temas, pesquisar, criar e aprender de maneira autônoma. Em resumo, a modelagem de software oferece oportunidades valiosas para aprimorar a alfabetização, tornando-a mais acessível, envolvente e eficaz.

Desenvolvimento do Software

**3.1 Modelo de processo de desenvolvimento**

O Modelo Incremental foi escolhido pela equipe por ser uma metodologia ágil e iterativa que possibilita a entrega de resultados em etapas e em um curto espaço de tempo, além de permitir realizar e alterar o projeto, se necessário. Este modelo combina elementos do Modelo em Cascata aplicados de maneira iterativa, ou seja, de forma que o progresso aconteça através de sucessivos refinamentos, melhorados a cada iteração. /\*REF: (PRESSMAN, 2006). Coloquei aqui porque não sei onde colocar \*/.

O modelo incremental se utiliza de uma abordagem flexível, na qual se produz pequenas partes que podem fornecer feedback para as suas sucessoras, possibilitando a construção de um modelo baseado nas nossas experiências empíricas do projeto com intuito de fornecer um software capaz de ajudar a reduzir o Analfabetismo global. O projeto em questão não é de larga escala e complexidade, adequando-se a nossa proposta fornecida por se tratar de um aplicativo em versão Beta.

O processo caso seja necessário alterar algo nessa implementação, é desenvolvido um novo incremento e o resultado é novamente apresentado. Cada aprimoramento é lançado como uma versão. Novas versões são criadas até que o sistema fique completo e adequado, para então, ser lançada a versão final. Os clientes podem estabelecer as prioridades das partes do sistema a serem desenvolvidas, especificando as mais úteis primeiro. Após o cliente estabelecer as funcionalidades necessárias, são criados os estágios (incrementos) de entrega, onde cada estágio fornece um conjunto de funcionalidades do sistema.

**3.2 Embasamento histórico**

Este modelo é abordado em um dos livros de Barry Boehm (Software Engineering Economics), um dos pioneiros da engenha de software. Uma das aplicações do modelo incremental pode ser observada em projetos realizados pela NASA, como o projeto Apollo. Os diferentes setores exigem diferentes abordagens para cada um, mas especificamente no lançamento de versões do projeto, se observa a utilização do modelo, para se obter feedbacks construtivos e melhoras conforme novos incrementos são realizados.

Engenharia de requisitos

**4.1 Viabilidade do sistema computacional**

**4.1.1 Faça uma análise preliminar.**

Neste ponto, você precisa delinear seu plano e é sempre uma boa ideia concentrar-se em quaisquer vantagens distintas que seu projeto, produto ou serviço tenha. Em seguida, você precisa identificar se há ou não algum risco associado a ele. Por exemplo, se for muito caro ou se houver muitos concorrentes. Identifique e analise os pontos fortes e fracos e certifique-se de fazer as modificações necessárias em seu plano ao identificar problemas.

**4.1. 2 Escopo do projeto.**

Esta seção analisa a oportunidade de negócios. Pense nela como a definição do motivo pelo qual esse projeto deve existir. Você se perguntará: "Para quem é esse projeto?", "Há necessidade dessa ideia?", "Quem são meus concorrentes?", etc.

**4.1.3 Requisitos do projeto.**

Esta seção deve analisar e descrever o que você precisará para concluir o seu projeto; materiais, recursos, orçamento e habilidades necessárias para a conclusão do projeto.

**4.1.4 Abordagem.**

É aqui que você detalha como seu projeto abordará seu caso de negócios ou sua proposta. Examine as alternativas e descreva os diferentes cursos de ação. Ao analisar essas rotas, você determinará se a sua proposta é viável e prática ou não.

**4.1.5 Faça uma pesquisa de mercado.**

Essa etapa é fundamental para o seu estudo porque lhe dá uma visão clara da receita que você pode esperar do projeto. Considere dados geográficos, demográficos, concorrentes e o valor do mercado.

**4.1.6 Avaliação do projeto.**

É aqui que você analisa o custo e a receita projetada do projeto. Mas lembre-se de que você também precisa levar em consideração o dinheiro que gastará ao realizar o projeto: honorários de mão de obra, fluxos de renda e qualquer outro recurso necessário. Depois que o custo total tiver sido calculado, você o avaliará por meio de uma análise de custo-benefício.

**4.1.7 Revise as informações.**

Neste ponto, você reexaminará as etapas anteriores e determinará se há algo que precisa ser ajustado. Em seguida, você deve apresentar suas descobertas e revisar o estudo de viabilidade com todas as partes interessadas.

**4.1.8 Tome uma decisão.**

Agora você tem informações suficientes para tomar uma decisão. É aqui que você determina se o seu projeto é viável ou não. Alguns aspectos que você deve considerar antes de tomar a decisão final são: se a ideia vale o esforço e o dinheiro total e se está alinhada com as metas estratégicas de longo prazo da sua empresa.

Aplicativos educativos podem desempenhar um papel crucial na promoção da alfabetização digital. Algumas maneiras pelas quais aplicativos podem ajudar a combater o analfabetismo digital:

1. **Personalização do Aprendizado**:
   * Aplicativos podem adaptar o conteúdo com base no nível de conhecimento do usuário. Isso permite que cada pessoa aprenda no seu próprio ritmo e de acordo com suas necessidades específicas.
2. **Aprendizado Lúdico**:
   * Muitos aplicativos usam elementos de gamificação para tornar o aprendizado divertido. Isso motiva os usuários a explorarem e aprender, especialmente crianças.
3. **Foco em Habilidades Essenciais**:
   * Aplicativos podem ensinar habilidades específicas, como busca na internet, avaliação de conteúdo e uso de ferramentas digitais. Essas habilidades são fundamentais para a alfabetização digital.
4. **Acesso a Recursos Digitais**:
   * Alguns aplicativos oferecem acesso a bibliotecas digitais, livros eletrônicos e materiais educacionais. Isso amplia o acesso à informação e à leitura.
5. **Inclusão de Libras e Outros Idiomas**:
   * Alguns aplicativos incluem recursos de Libras (Língua Brasileira de Sinais) ou oferecem suporte em diferentes idiomas. Isso promove a inclusão e a diversidade.

Aqui estão alguns aplicativos específicos que podem ajudar na alfabetização digital:

* **EduEdu**: Personalizado para crianças de 5 a 9 anos, o EduEdu oferece exercícios de língua portuguesa adaptados ao nível de conhecimento de cada aluno.
* **GraphoGame**: Desenvolvido em colaboração com cientistas e professores, o GraphoGame ajuda no desenvolvimento das habilidades de leitura e ortografia. Ele oferece experiências escolares e domésticas para crianças até 12 anos.
* **Duolingo**: Além de ensinar idiomas, o Duolingo também oferece um aplicativo para alfabetização infantil, com lições simples e gamificadas.

**4.2 Levantamentos dos requisitos**

O levantamento de requisitos é o processo de coletar informações sobre o que o sistema deve fazer. Isso envolve interagir com os stakeholders (usuários, clientes, especialistas) para entender suas necessidades e expectativas. Técnicas comuns de levantamento de requisitos incluem entrevistas, questionários, workshops, observação e análise de documentos existentes.

**4.3 Análise e negociação de requisitos**

A análise de requisitos envolve examinar os requisitos coletados para entender sua viabilidade técnica e funcional. Isso inclui identificar inconsistências, ambiguidades e conflitos entre diferentes requisitos. A negociação de requisitos ocorre quando há divergências entre as partes interessadas. O objetivo é chegar a um consenso sobre os requisitos, considerando restrições, prioridades e recursos disponíveis.

**4.4 Especificação de Requisitos**

A especificação de requisitos é a documentação formal dos requisitos coletados e analisados. Ela descreve detalhadamente o que o sistema deve fazer. A especificação pode incluir:

* + Requisitos Funcionais: Descrição das funcionalidades do sistema.
  + Requisitos Não Funcionais: Aspectos como desempenho, segurança, usabilidade e confiabilidade.
  + Casos de Uso: Cenários de interação entre usuários e sistema.
  + Diagramas de Fluxo: Representações gráficas de processos.
  + Protótipos: Modelos visuais do sistema.
  + Matriz de Rastreabilidade: Relaciona requisitos a componentes do sistema.

**4.5 Validação dos requisitos**  
 A validação de requisitos, por outro lado, concentra-se em assegurar que o software desenvolvido atenda às necessidades reais dos stakeholders. Ela verifica se os requisitos escritos estão de acordo com as solicitações das partes interessadas. A validação ajuda a detectar erros de um estágio inicial do desenvolvimento do produto, evitando retrabalho excessivo posteriormente no ciclo de vida do sistema. Ela garante precisão e clareza nos dados, mitigando defeitos nos requisitos coletados. A validação verifica se os requisitos descrevem os objetivos e funções do sistema pretendido.

**4.6 Verificação de Requisitos:**

A verificação de requisitos é o processo de confirmar que os requisitos do sistema estão bem documentados e atendem aos critérios de qualidade para requisitos. Em outras palavras, a verificação verifica se os requisitos foram especificados corretamente e se contêm todos os elementos necessários de requisitos bem escritos.

Essa etapa é crítica no desenvolvimento do sistema, pois ajuda a garantir que o produto atenda aos seus objetivos e funções conforme pretendido. Antes do projeto, os requisitos devem ser verificados e aprovados para evitar retrabalho posterior.

A verificação de requisitos é necessária para evitar problemas durante o desenvolvimento, teste e implantação do sistema. Requisitos ausentes, incorretos ou inconsistentes podem levar a resultados insatisfatórios e custos excessivos.

Arquitetura de software

**5.1 tipo de arquitetura de software**

A arquitetura de software é fundamental para o desenvolvimento de sistemas escaláveis, seguros e de fácil manutenção. Existem vários tipos de arquitetura, cada um com suas características específicas. Vou apresentar alguns dos principais modelos:

**5.1.1 Arquitetura em Camadas (Layered Architecture):** Nesse modelo, o sistema é dividido em camadas distintas, cada uma responsável por uma funcionalidade específica. Por exemplo, temos a camada de apresentação, a camada de negócios e a camada de dados. Essa abordagem facilita a manutenção e a evolução do software.

**5.1.2 Arquitetura Cliente-Servidor:** Aqui, o sistema é separado em dois processos: o cliente, que solicita informações, e o servidor, que mantém os dados. É comum em aplicativos e sistemas que envolvem comunicação entre dispositivos.

**5.1.3 Arquitetura Orientada a Serviços (SOA):** Nesse modelo, o sistema é integrado com outros sistemas por meio de serviços que expõem funcionalidades específicas, como APIs e web services. Isso permite a reutilização de componentes e a flexibilidade na composição de soluções.

**5.1.4 Arquitetura em Micro serviços:** Aqui, o sistema é fragmentado em serviços independentes e autônomos que se comunicam entre si. Cada micro serviço é responsável por uma parte específica da aplicação, o que facilita a escalabilidade e a manutenção.

**5.1.5 Opções de arquitetura para um aplicativo móvel:**

**5.1.6 Aplicativos Móveis Nativos:**

**5.1.6.1** Esses aplicativos são desenvolvidos especificamente para uma plataforma (iOS ou Android) e são escritos em linguagens nativas, como Swift (para iOS) ou Kotlin (para Android).

**5.1.6.2** Vantagens: Alto desempenho, acesso total aos recursos do dispositivo e melhor experiência do usuário.

**5.1.6.3** Desvantagens: Requer desenvolvimento separado para cada plataforma.

**5.1.7 Aplicativos da Web para Dispositivos Móveis:**

**5.1.7.1** Esses aplicativos são acessados por meio de um navegador em dispositivos móveis.

**5.1.7.2** Vantagens: Desenvolvimento único para várias plataformas, atualizações instantâneas e menor custo.

**5.1.7.3** Desvantagens: Menor desempenho em comparação com aplicativos nativos.

**5.1.8 Aplicativos Móveis Híbridos:**

**5.1.8.1** Esses aplicativos combinam elementos de aplicativos nativos e da web.

**5.1.8.2** Vantagens: Desenvolvimento único para várias plataformas, acesso a recursos do dispositivo e atualizações rápidas.

**5.1.8.3** Desvantagens: Desempenho inferior aos aplicativos nativos.

**5.1.9 Model-View-ViewModel (MVVM):**

**5.1.9.1** Um padrão de arquitetura popular para aplicativos móveis que separa a UI e as camadas de lógica de negócios em componentes separados: Model (dados), View (apresentação) e ViewModel (lógica de apresentação).

(Considerar fatores como escalabilidade, manutenção, tempo de desenvolvimento e experiência do usuário ao escolher a arquitetura mais adequada para o seu aplicativo móvel)

**5.2 Embasamentos histórico?**

**6.1 Framework?**

**6.2 Documentação dos entregáveis/sprints?**

Conclusão

**7.1 Resultados e Discussão**

**7.2 Resultados Quantitativos**

**7.3 Contribuições e Limitações**

Referencias

PRESSMAN, Roger. S. Engenharia de Software, 6ª Edição. McGrawHill, Nova York, EUA, 2006

SOMMERVILLE, Ian. Engenharia de Software, 9ª Edição. Pearson. São Paulo, Brasil, 2011

<https://www.linkedin.com/pulse/apollo-program-masterclass-incremental-iterative-large-thomas-meloche/>

<https://appel.nasa.gov/2018/11/27/spotlight-on-lessons-learned-aligning-system-development-models-with-insight-approaches/>

OS DIAGRAMAS

1. Diagrama de caso de uso

Atores:

Usuário: Representa os alunos que usam o aplicativo

Administrador: Representa os administradores que gerenciam o sistema

Casos de Uso:

Realizar lições: O usuário pode acessar lições de Português e matemática, completar exercícios e aprender novas palavras e frases

Acompanhar progresso: O usuário pode verificar seu progresso, como pontuação, nível de fluência e conquistas

Configurar perfil: O usuário pode personalizar seu perfil, adicionar informações pessoais e definir preferências

Comunicar-se com outros usuários: Os usuários podem interagir com outros alunos, trocar mensagens e participar de fóruns

Gerenciar assinatura: Os usuários com assinatura premium podem gerenciar sua conta, pagamentos e benefícios

Relacionamentos:

O ator Usuário está associado a todos os casos de uso mencionados acima

O ator Administrador pode gerenciar usuários, conteúdo e problemas técnicos

2. Descrição geral do Diagrama de classes:

Classe “Usuário”:

Representa os usuários do aplicativo

Atributos: nome de usuário, senha, nível de fluência, conquistas e histórico de atividades

Métodos para realizar lições: acompanhar o progresso e interagir com outros usuários

Classe “Curso”:

Representa as trilhas oferecidos pelo Aplicativo

Atributos: nome do curso, nível, conteúdo e requisitos

Métodos: acessar lições específicas do curso/ trilha

Classe “Lição”:

Representa uma lição individual dentro de um curso

Atributos: tributos como título, descrição, dificuldade e pontuação

Métodos: realizar a lição e verificar respostas

Classe “Exercício”:

Representa os exercícios dentro de uma lição

Atributos: tipo de exercício (Desenhar, preenchimento de lacunas, escolha múltipla etc.) e texto da pergunta

Métodos: avaliar respostas dos usuários

Classe “Conquista”:

Representa as conquistas desbloqueadas pelos usuários

Atributos: nome da conquista, descrição e pontuação

Métodos: verificar se um usuário atingiu determinada conquista

Classe “Administração”:

Representa funções administrativas do Aplicativo

Métodos: gerenciar cursos, usuários e conteúdo

3.Diagrama de sequência da realização de uma lição:

O usuário inicia a lição selecionando um tópico (por exemplo, “vogais”)

O sistema carrega o conteúdo da lição

O usuário visualiza a primeira pergunta

O usuário responde à pergunta

O sistema verifica a resposta e fornece feedback (correto ou incorreto)

O usuário avança para a próxima pergunta

O processo se repete até que todas as perguntas da lição sejam concluídas

4.fluxo de interações do usuário durante uma lição:

Início da Lição:

O usuário seleciona uma lição específica no aplicativo

O sistema carrega o conteúdo da lição, incluindo exercícios e tópicos relevantes

Apresentação do Exercício:

O usuário visualiza a primeira pergunta ou exercício

O exercício pode ser uma tradução, preenchimento de lacunas, escolha múltipla ou outro tipo

Resposta do Usuário:

O usuário responde à pergunta:

O sistema registra a resposta e a avalia

Feedback do Sistema:

O sistema fornece feedback imediato sobre a resposta do usuário

Se a resposta estiver correta, o usuário avança para a próxima pergunta

Se a resposta estiver incorreta, o sistema oferece dicas ou explicações para ajudar o usuário a aprender

Progresso e Pontuação:

O sistema atualiza o progresso do usuário na lição

A pontuação é calculada com base nas respostas corretas e na rapidez das respostas

Iteração com Diferentes Exercícios:

O usuário continua respondendo a perguntas e exercícios até concluir a lição

Cada exercício pode abordar diferentes aspectos, como vocabulário, gramática ou pronúncia

(Precisa de uma versão para matemática)

Conclusão da Lição:

Após concluir todos os exercícios, o usuário recebe uma pontuação geral

O sistema registra o progresso e atualiza o nível de fluência do usuário

5.Modelo de banco de dados (entidades e relações)

Usuários:

Armazenar informações sobre os usuários, como nome, e-mail, senha (criptografada), pontuação, nível, conquistas e

Relacionamento com outras entidades: Pode estar relacionado a tabelas de “conquistas”, “lições concluídas” etc

Lições:

Cada lição pode ter um título, descrição, conteúdo (texto, imagens, vídeos) e nível de dificuldade.

Relacionamento com outras entidades: Pode estar relacionado a tabelas de “usuários” (para rastrear o progresso do usuário) e “conquistas” (para desbloquear conquistas após a conclusão de uma lição)

Conquistas:

Armazene informações sobre as conquistas que os usuários podem ganhar, como nome, descrição e pontuação associada

Relacionamento com outras entidades: Pode estar relacionado a tabelas de “usuários” (para rastrear as conquistas de cada usuário)

Atividades:

Registre as atividades dos usuários, como conclusão de lições, respostas corretas/erradas em quizzes, etc

Relacionamento com outras entidades: Pode estar relacionado a tabelas de “usuários” e “lições”

Quizzes:

Se o aplicativo incluir quizzes, você pode criar uma tabela para armazenar perguntas, opções de resposta e respostas corretas

Relacionamento com outras entidades: Pode estar relacionado a tabelas de “lições” (cada lição pode ter um quiz associado)

Trilhas:

O aplicativo fornece duas trilhas, você pode criar uma tabela para armazenar informações sobre os avancos tanto na area Matematica quanto na area de Linguagem

Relacionamento com outras entidades: Pode estar relacionado a tabelas de “usuários” (preferências de Trilha) e “lições” (conteúdo direcionado)

MODELAGEM DE BANCO DE DADOS

# **Fundamentos**

Certos conceitos são utilizados na modelagem conceitual para descrever os papéis de cada componente do domínio analisado. Dentre eles, podemos citar:

* **Instância**: denomina uma ocorrência, um registro que existe ou passou a existir de uma entidade;
* **Entidade**: define qualquer coisa que seja identificável, singular e tenha existência bem delimitada, podendo ser classificada entre “fraca” e “forte” de acordo com sua cardinalidade e representada por um retângulo no DER;
* **Atributo**: é usado para referir-se à cada característica possuída pelas instâncias de uma entidade ou de um relacionamento e sua representação no DER se dá em forma de balões — embora seja desencorajada por poluí-lo facilmente;
* **Relacionamento**: descreve um evento significativo que ocorre entre instâncias de duas entidades e é representado por um losango, sendo considerado e representado a partir da modelagem lógica como uma entidade no DER quando tem cardinalidade N:N (surpresa!);
* **Cardinalidade**: conceito usado para dizer quantas vezes uma instância de uma entidade pode se relacionar com instâncias de outra entidade, também referenciado como “grau de relacionamento”; e
* **Condicionalidade**: responsável por comunicar se a existência de uma instância de uma entidade está condicionada à existência de uma instância de outra entidade, não sendo muito representada no DER.

## **Relacionamentos entre instâncias de entidades**

São comumente efetivados com estas 3 cardinalidades:

1. **“Um” para “Um” (1:1)**: uma instância de uma entidade *pode ou deve se relacionar com* uma e apenas uma instância de outra entidade;
2. **“Um” para “Muitos” (1:N)**: uma instância de uma entidade *pode ou deve se relacionar com* uma ou mais instâncias de outra entidade — é com toda certeza a cardinalidade mais encontrada em bancos de dados; e
3. **“Muitos” para “Muitos” (N:N)**: uma ou mais instâncias de uma entidade *podem ou devem se relacionar com* uma ou mais instâncias de outra entidade — esse tipo de relacionamento transforma-se em uma entidade a partir da modelagem lógica.

## **Condição de relacionamentos**

Podem ser regidos sob 2 condicionalidades (exemplificadas abaixo):

1. **Parcial**: quando instâncias de uma entidade Telefone ***podem*** estar relacionadas com instâncias de Pessoa em uma agenda, ou seja, onde o cadastro de números de telefone sem nome é permitido; e
2. **Total**: quando instâncias de uma entidade Carro ***devem*** estar relacionadas com instâncias de Cliente para entrar em uma oficina, ou seja, onde o cadastro de carros sem dono é proibido.
3. **Entidades de uma relação 1:N**

Duas categorias são usadas para defini-las:

1. **“Fortes”**: para entidades do lado 1 da relação; e
2. **“Fracas”**: para entidades do lado N.

Entidades “fracas” são capazes de nos dizer quando participam de uma relação 1:N pois, após a modelagem lógica, sempre terão um atributo especial que fará menção às entidades “fortes”. Estas últimas, contudo, não permitem o mesmo feito.

## **Boas práticas para nomeação de componentes do MER**

* É recomendado nomear entidades e atributos com substantivos ou adjetivos e relacionamentos com verbos, ambos sem espaços e caracteres especiais (como letras acentuadas e símbolos);
* É proibido nomes que comecem com números.

**Modelo de banco de dados (entidades** e **relações)**

**Usuários**:

* + Armazenar informações sobre os usuários, como nome, e-mail, senha (criptografada), pontuação, nível, conquistas e
  + Relacionamento com outras entidades: Pode estar relacionado a tabelas de “conquistas”, “lições concluídas” etc.

**Lições**:

* + Cada lição pode ter um título, descrição, conteúdo (texto, imagens, vídeos) e nível de dificuldade.
  + Relacionamento com outras entidades: Pode estar relacionado a tabelas de “usuários” (para rastrear o progresso do usuário) e “conquistas” (para desbloquear conquistas após a conclusão de uma lição).

**Conquistas**:

* + Armazene informações sobre as conquistas que os usuários podem ganhar, como nome, descrição e pontuação associada.
  + Relacionamento com outras entidades: Pode estar relacionado a tabelas de “usuários” (para rastrear as conquistas de cada usuário).

**Atividades**:

* + Registre as atividades dos usuários, como conclusão de lições, respostas corretas/erradas em quizzes, etc.
  + Relacionamento com outras entidades: Pode estar relacionado a tabelas de “usuários” e “lições”.

**Quizzes**:

* + Se o aplicativo incluir quizzes, você pode criar uma tabela para armazenar perguntas, opções de resposta e respostas corretas.
  + Relacionamento com outras entidades: Pode estar relacionado a tabelas de “lições” (cada lição pode ter um quiz associado).

**Idiomas**:

* + Se o aplicativo suportar vários idiomas, você pode criar uma tabela para armazenar informações sobre os idiomas disponíveis.
  + Relacionamento com outras entidades: Pode estar relacionado a tabelas de “usuários” (preferências de idioma) e “lições” (conteúdo traduzido).