

**PROJETO MASTERSCORE****Requisitos da disciplina Modelagem de Software e Arquitetura de Sistemas****INTEGRANTES DO PROJETO e RA'S**

Julia Gomes Basilio	-	25028157
Julia de Souza Silva	-	25028077
Maria Eduarda Barberino Olo	-	25028088
Leonardo Cruz Lamari	-	25028194

São Paulo

2025

## Sumário

<b>1 INTRODUÇÃO .....</b>	<b>3</b>
<b>2. DOCUMENTO DE ABERTURA DO PROJETOS.....</b>	<b>4</b>
2.1 – Project Charter.....	4
2.2 – Histórias do Usuário .....	5
<b>3. DESIGN SPRINT – Ideação e prototipação do desafio.....</b>	<b>6</b>
3.1 Desafio .....	6
3.2 Entender Mapear.....	6
3.3 Ideação – desenho da solução (trilha do usuário) .....	6
3.4 Prototipagem.....	6
<b>4.REQUISITOS DE SISTEMA .....</b>	<b>6</b>
4.1 REQUISITOS FUNCIONAIS DE SOFTWARE .....	6
4.2 REQUISITOS NÃO FUNCIONAIS DE SOFTWARE .....	7
<b>5. CASOS DE USO .....</b>	<b>8</b>
<b>6. DIAGRAMA DE CLASSE.....</b>	<b>8</b>
<b>7. ARQUITETURA DO SISTEMA .....</b>	<b>8</b>
<b>8. REFERÊNCIAS BIBLIOGRÁFICAS .....</b>	<b>8</b>

## 1 INTRODUÇÃO

O projeto Master Score foi desenvolvido por uma equipe de quatro integrantes, cada um contribuindo e ajudando a desenvolver partes diferentes do projeto para a construção do jogo. Todos os membros foram responsáveis por discutir e elaborar a solução proposta, definindo toda a lógica do jogo, o sistema de pontuação e o funcionamento do acúmulo de moedas e resgate de cupons. Cada membro contribuiu de forma específica para o MasterScore e contou com a ajuda de outros integrantes na divisão de tarefas e desenvolvimento do projeto. Um dos integrantes focou principalmente na programação, outro na parte de design e interface, enquanto outros cuidaram do banco de dados, do acúmulo de pontos e do funcionamento geral do jogo. Essa divisão de tarefas permitiu que cada aspecto do projeto fosse desenvolvido com atenção, e a colaboração constante garantiu que todas as partes se encaixassem com sucesso dentro do jogo.

## MasterScore

**Nome da Instituição:** MasterScore

**Objetivo da Aplicação:**

ADS 1 O objetivo do desafio desse projeto corresponde ao desenvolvimento de um jogo educativo e interativo, inspirado no famoso jogo *Perguntados*, que visa tornar o aprendizado mais dinâmico e envolvente.

O aplicativo tem como objetivo principal testar o conhecimento do usuário nas seguintes categorias: Matemática Financeira, História cultura e a opção de MIX, onde misturamos todas as categorias em um só. Nele utilizamos elementos de gamificação — como acúmulo de pontos, progressão de níveis e recompensas — para aumentar o engajamento dos usuários.

## Desafio:

O desafio está em desenvolver um sistema educativo que una diversão e aprendizado, permitindo que o usuário jogue e aprenda ao mesmo tempo. O sistema deve oferecer perguntas em diferentes categorias e permitir que o jogador troque seus pontos por dicas dentro de uma “lojinha” do jogo. Essas dicas servem para dar pistas ao usuário de onde encontrar os cupons de desconto de maneira divertida e intuitiva.

**Personas a serem atendidas:**

- **Jogador/Usuário Final:** pessoa com interesse em aprender de forma dinâmica, utilizando o celular como ferramenta de estudo e ganhando recompensas em troca
- **Administrador do Jogo:** responsável por cadastrar novas perguntas e acompanhar métricas de uso.

#### Recursos de Referência:

<https://kahoot.com>

<https://triviacrack.com>

## 2. DOCUMENTO DE ABERTURA DO PROJETO

### 2.1 – Project Charter

#### Prefácio:

Este documento tem como finalidade apresentar o projeto MasterScore, desenvolvido pelos alunos do curso de Análise e desenvolvimento de sistemas da FECAP, sob a orientação do professor Aimar Lopes, na disciplina de Modelagem e Arquitetura de Software.

#### Histórico de Versões:

Versões	Data	Descrição
1.0	10/10/2025	Criação inicial do documento e estruturação do Design Sprint e requisitos.

#### Introdução:

O sistema **MasterScore** foi criado para suprir a necessidade de plataformas educativas mais atrativas e acessíveis.

Ele funciona como um aplicativo de perguntas e respostas, onde os usuários acumulam pontos e com eles são compradas dicas que levam à cupons que podem ser utilizados pelo usuário ao realizar compras em outros sites e aplicativos.



O sistema contribui para o desenvolvimento cognitivo e promove o aprendizado ativo, alinhando-se aos objetivos educacionais e de engajamento digital.

**Glossário:**

**Gamificação:** Aplicação de elementos e mecânicas de jogos (como pontos, rankings e recompensas) em ambientes não lúdicos, com o objetivo de aumentar o engajamento e a motivação dos usuários.

**Quiz:** Jogo de perguntas e respostas, geralmente de múltipla escolha, utilizado para testar conhecimentos sobre determinado tema.

**MVP (Produto Mínimo Viável):** Primeira versão funcional de um produto, desenvolvida com o mínimo de recursos necessários para validar a ideia e obter feedback dos usuários.

**Interface do Usuário (UI):** Parte visual do aplicativo com a qual o usuário interage diretamente, composta por botões, menus e telas.

**Experiência do Usuário (UX):** Conjunto de percepções e sentimentos do usuário ao interagir com o sistema, relacionado à facilidade de uso e à satisfação geral.

**Backend:** Parte do sistema responsável pelo processamento de dados e regras de negócio, não visível ao usuário final.

**Frontend:** Parte visível e interativa do sistema com a qual o usuário interage (as telas, menus e elementos visuais).

**Banco de Dados:** Estrutura que armazena informações persistentes, como cadastros de usuários, perguntas, pontuações e cupons.

**API (Interface de Programação de Aplicações):** Conjunto de rotinas e padrões que permitem a comunicação entre sistemas diferentes.

**Prototipagem:** Etapa do desenvolvimento onde são criados modelos visuais (telas, fluxos ou simulações) para testar a aparência e a usabilidade de um sistema antes de sua implementação real.

**Definição de Requisitos de Usuário**

- Interface intuitiva e gamificada;

- Feedback imediato após cada resposta;
- Sistema de pontos;
- Cupom de recompensa.

**Requisitos Não Funcionais:**

- Alta usabilidade (navegação fácil e responsiva).
- Baixo tempo de resposta
- Segurança dos dados do jogador

**Arquitetura do sistema:**

O sistema seguirá o padrão **MVC (Model-View-Controller)**, dividido em:

**Model:** manipulação de dados (usuário, perguntas, pontuação e cupons);

**View:** telas de interação do usuário;

**Controller:** controle das regras do jogo e das pontuações.

*Arquitetura do Sistema*

O sistema seguirá o padrão **MVC (Model-View-Controller)**, dividido em:

**Model:** manipulação de dados (usuário, perguntas, pontuação e cupons);

**View:** telas de interação do usuário;

**Controller:** controle das regras do jogo e das pontuações.

**Especificação de Requisitos do Sistema**

O requisito funcional RFS01 define que o sistema deve permitir que o usuário se cadastre no jogo, realizando operações de criação, leitura, atualização e exclusão. O jogador poderá criar sua conta informando nome de usuário e senha, sendo esses dados validados e armazenados pelo sistema. Ao concluir o cadastro, o sistema exibirá uma mensagem de sucesso — “Cadastro realizado com sucesso!”, ou informará o erro “Usuário já cadastrado” caso os dados já existam.

O requisito RFS02 descreve o processo de login, no qual o usuário informa seu nome de usuário e senha. O sistema deve validar as credenciais no banco de dados e liberar o acesso ao jogo, redirecionando o jogador ao menu principal em caso de sucesso, exibindo a mensagem “Login

efetuado com sucesso.” Se as credenciais estiverem incorretas, o sistema mostrará a mensagem “Credenciais inválidas.”

O RFS03 trata da seleção de categoria, permitindo que o jogador escolha o tema das perguntas que deseja responder. O sistema exibirá as opções disponíveis: Matemática Financeira, História, Cultura e Mix, e ao selecionar uma delas, carregará as perguntas correspondentes. Essa etapa garante que o jogador possa direcionar seu aprendizado para o tema de sua preferência.

O RFS04 estabelece que o sistema deve exibir perguntas e alternativas de múltipla escolha referentes à categoria selecionada. Cada pergunta possui quatro alternativas (A, B, C, D), sendo apenas uma correta. As perguntas devem ser apresentadas aleatoriamente, e o sistema aguardará a escolha do jogador.

O RFS05 descreve o sistema de pontuação, responsável por atribuir pontos ao jogador conforme o número de respostas corretas. Cada acerto deve somar pontos ao perfil do jogador, exibindo feedback imediato com as mensagens “Resposta correta! +10 pontos” ou “Errou! Tente novamente.” A pontuação acumulada é armazenada e atualizada no banco de dados, permitindo acompanhar o desempenho de cada usuário.

O RFS06 trata da funcionalidade de troca de pontos por dicas, que permite ao jogador utilizar os pontos acumulados para adquirir dicas sobre onde encontrar os cupons fictícios dentro do jogo. O sistema exibirá as dicas disponíveis e seus respectivos valores em pontos. Ao selecionar uma dica, o sistema verificará o saldo e realizará a troca, exibindo a mensagem “Dica adquirida com sucesso!” ou, caso o saldo seja insuficiente, “Pontos insuficientes.”

#### Especificação de requisitos Não Funcionais

O requisito não funcional RNF01 define que todas as ações do sistema, como login, seleção de categoria, exibição de perguntas e troca de dicas, devem ocorrer em até dois segundos. Essa agilidade é essencial para manter a fluidez da navegação e o interesse do jogador, evitando lentidão que comprometa a experiência de uso.

O RNF02 estabelece que o sistema deve apresentar uma interface intuitiva, limpa e fácil de usar, com botões grandes e menus visuais bem definidos. O design deve seguir boas práticas, garantindo acessibilidade e legibilidade, de forma que até usuários com pouca familiaridade com tecnologia consigam jogar sem dificuldades.

O RNF03 aborda a compatibilidade do sistema, que deve funcionar corretamente em diferentes dispositivos, como smartphones Android, navegadores e desktops. O design responsivo deve permitir que a interface se ajuste automaticamente ao tamanho da tela, mantendo a usabilidade e o layout estáveis em todas as plataformas.

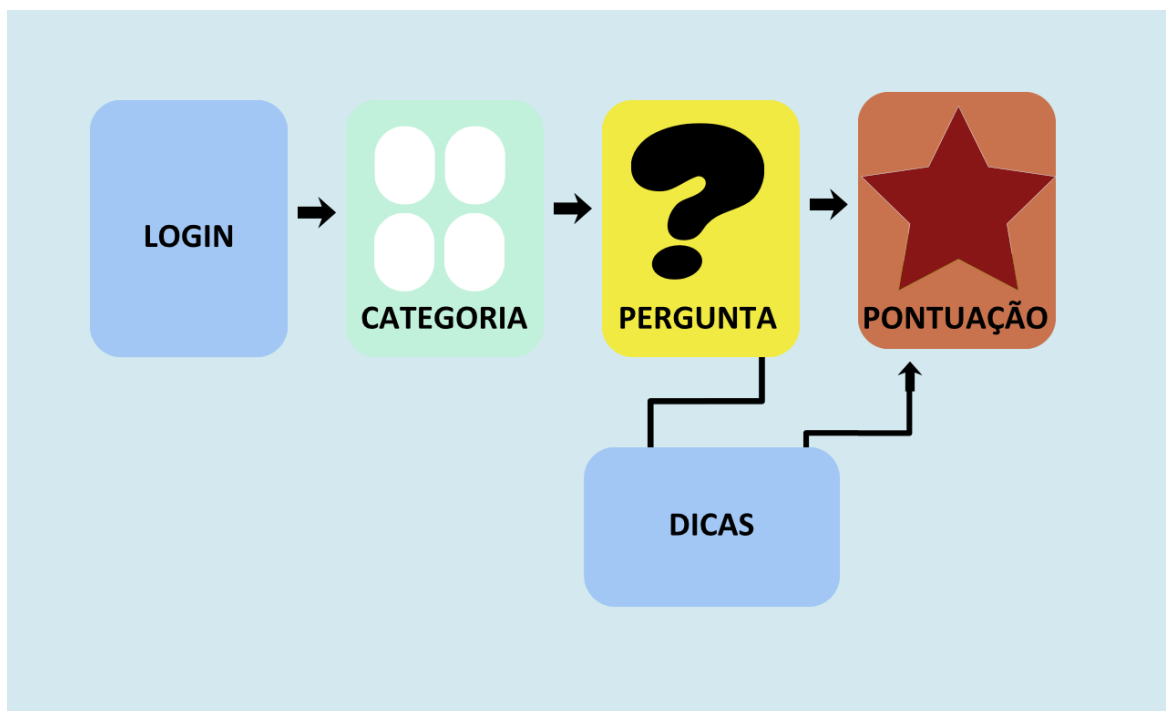
O RNF04 trata da segurança. Todos os dados de login e pontuação devem ser armazenados de forma segura, com validação e restrição de acessos não autenticados. O sistema deve utilizar técnicas de proteção de dados para impedir acessos indevidos, garantindo a integridade das informações dos usuários.

O RNF05 garante o funcionamento contínuo do sistema, mesmo em uso prolongado ou sob múltiplas interações simultâneas. Devem ser implementadas rotinas de verificação e correção de erros para evitar falhas e travamentos, mantendo o jogo estável durante toda a execução.



Por fim, o RNF06 refere-se à manutenção do software. O código deve ser modular, documentado e seguir boas práticas de desenvolvimento, permitindo que futuras atualizações, como novas perguntas, categorias ou melhorias visuais — sejam implementadas sem comprometer as funcionalidades existentes.

### Modelos do sistema



### Evolução do sistema

#### Plataforma de Execução

O sistema será inicialmente desenvolvido para rodar em dispositivos móveis (Android e iOS) e navegadores web modernos. Considera-se que o hardware terá capacidade suficiente para suportar gráficos básicos e conexões com a internet para atualização de conteúdo.

#### Conectividade e Atualizações

O jogo pressupõe que os usuários terão acesso regular à internet para sincronização de dados, atualização de perguntas e estatísticas.

#### Interface e Usabilidade

A interface será simples e intuitiva. Presume-se que os dispositivos suportam telas sensíveis ao toque ou interfaces apontadoras, com resolução mínima para garantir a usabilidade.



#### Banco de Dados e Conteúdo

O sistema manterá um banco de perguntas, de dicas e pontos, poderá ser facilmente expandido ou modificado.

Possíveis mudanças futuras:

#### Evolução do Hardware

Com o avanço dos dispositivos, espera-se que seja possível implementar animações.

#### Mudanças nas Necessidades dos Usuários

Futuramente, os usuários podem demandar funcionalidades como recuperação de sua conta, seu usuário, através de seu e-mail.

### Apêndices

#### Servidor

O servidor hospeda o banco de dados e a lógica do jogo (Controller), sendo crucial para a segurança e estabilidade do sistema.

**Processador (CPU):** Mínimo de 2 vCPUs (Virtual CPUs). Isso é necessário para processar as regras do jogo e gerenciar múltiplas sessões de usuários simultâneas.

**Memória RAM:** Mínimo de 4 GB. Essencial para o desempenho rápido, garantindo que as ações do sistema ocorram em até 2 segundos.

**Armazenamento:** Mínimo de 50 GB em disco SSD (Solid State Drive). O armazenamento de alta velocidade é vital para o banco de dados de perguntas, usuários e cupons, suportando futuras expansões.

#### Cliente

O jogo deve ser compatível com smartphones (Android e iOS) e navegadores modernos.

**Sistema Operacional:** Mínimo Android 9.0 ou iOS 13.

**Processador (CPU):** Quad-core (1.5 GHz). Necessário para a renderização suave da interface gamificada, que deve ser intuitiva e com bom *design*.

**Memória RAM:** Mínimo de 2 GB. Garante a estabilidade e a ausência de travamentos durante o uso prolongado do jogo.

Requisitos de Banco de Dados

### **Organização Lógica de Dados**

#### **As principais entidades são:**

1. **USUARIO:** Armazenar os dados de cadastro dos jogadores (nome e senha) e manter o registro da pontuação acumulada. Se relaciona com as tabelas de DICAS e TEMA.
2. **TEMA:** Armazenar os temas do jogo (Matemática Financeira, História, Cultura e Mix). Se relaciona com a tabela de PERGUNTA.
3. **PERGUNTA:** Armazenar o texto da pergunta, a categoria à qual pertence, e a resposta correta para a validação do RFS04. Se relaciona com a tabela de TEMA e PONTOS.
4. **DICA:** Armazenar as informações sobre as dicas disponíveis na loja e o respectivo valor para a troca. Se relaciona com a tabela de CUPOM.
5. **PONTOS:** Registrar as transações de pontuação, detalhando os pontos ganhos por cada USUARIO a cada resposta correta, suportando o RFS05. Registra a pontuação do USUARIO e a PERGUNTA respondida.

## **2.2 – Histórias do Usuário**

- Como usuário típico, eu gostaria de animações ou sons leves para deixar a experiência mais divertida.
- Como usuário típico, eu gostaria de pontuação e recompensas.
- Como usuário típico, eu gostaria de um modo progressão de fases e níveis.
- Como usuário típico, eu gostaria de poder parar e voltar sem perder muito do progresso ou do contexto do jogo.
- Como usuário típico, eu gostaria de que não houvesse a necessidade de "vencer" ou "ganhar", o que importa é a experiência.

### **História do usuário para Aprendizado**

- Como usuário típico, eu gostaria de conteúdo educativo integrado ao jogo.

- Como usuário típico, eu gostaria de um jogo que ensina enquanto o usuário joga (por exemplo: matemática, idiomas, lógica, história, etc.).
- Como usuário típico, eu gostaria de um sistema de pontuação.
- Como usuário típico, eu gostaria de feedback imediato para mostrar ao jogador que acertou ou errou.
- Como usuário típico, eu gostaria de que o usuário escolhesse temas que deseja estudar ou áreas que quer reforçar.

### 3. DESIGN SPRINT – Ideação e prototipação do desafio

#### 3.1 Desafio

#### 3.2 Entender Mapear

#### 3.3 Ideação – desenho da solução (trilha do usuário)

#### 3.4 Prototipagem

## 4. REQUISITOS DE SISTEMA

### 4.1 REQUISITOS FUNCIONAIS DE SOFTWARE

RFS01	
<b>Função</b>	Permitir que o usuário se cadastre no jogo
<b>Descrição</b>	Realizar operações de CRUD (create, read, update e delete).
<b>Entradas</b>	Nome de usuário e senha
<b>Fonte</b>	Usuário
<b>Saídas</b>	Mensagem de sucesso “Cadastro realizado com sucesso!” ou erro “E-mail já cadastrado.”



<b>Ação</b>	<p>-O usuário acessa a tela de cadastro e informa os dados obrigatórios.</p> <p>-O sistema valida os campos e salva as informações.</p>
-------------	---

RFS02	
<b>Função</b>	Login do usuário
<b>Descrição</b>	Validar credenciais e liberar acesso ao jogo.
<b>Entradas</b>	Usuário e senha
<b>Fonte</b>	Usuário
<b>Saídas</b>	Mensagem de “Login efetuado com sucesso.” ou erro “Credenciais inválidas.”

RFS03	
<b>Função</b>	Permitir ao usuário escolher a categoria das perguntas.
<b>Descrição</b>	O sistema deve apresentar as categorias disponíveis: Matemática Financeira, História, Cultura e Mix.
<b>Entradas</b>	Categoria selecionada pelo jogador.
<b>Fonte</b>	Usuário
<b>Saídas</b>	Tela com perguntas correspondentes à categoria escolhida.

<b>Ação</b>	<ul style="list-style-type: none"> <li>-O sistema exibe as categorias disponíveis.</li> <li>-O usuário escolhe uma delas.</li> <li>-O sistema carrega o conjunto de perguntas dessa categoria.</li> </ul>
-------------	---

RFS04	
<b>Função</b>	Exibir perguntas e alternativas de múltipla escolha da categoria selecionada.
<b>Descrição</b>	Cada pergunta contém quatro alternativas (A, B, C, D), sendo apenas uma correta.
<b>Entradas</b>	Categoria selecionada.
<b>Fonte</b>	Banco de dados de perguntas.
<b>Saídas</b>	Tela com pergunta e quatro opções de resposta.
<b>Ação</b>	<ul style="list-style-type: none"> <li>-O sistema seleciona aleatoriamente uma pergunta da categoria escolhida.</li> <li>-Exibe as alternativas.</li> <li>-Aguarda a resposta do jogador.</li> </ul>

RFS05	
<b>Função</b>	Atribuir pontos ao jogador conforme o número de respostas corretas.
<b>Descrição</b>	A cada resposta correta, o sistema soma pontos no perfil do jogador.
<b>Entradas</b>	Resposta selecionada pelo jogador.

<b>Fonte</b>	Usuário e banco de dados de respostas corretas.
<b>Saídas</b>	Mensagem “Resposta correta! +10 pontos” ou “Errou! Tente novamente.”
<b>Ação</b>	<p>-O sistema compara a resposta do jogador com a correta.</p> <p>-Atualiza a pontuação acumulada.</p> <p>-Exibe feedback instantâneo.</p>

RFS06	
<b>Função</b>	Permitir que o jogador utilize pontos acumulados para trocar por dicas de onde estão os cupons
<b>Descrição</b>	Os pontos conquistados nas partidas podem ser usados para adquirir dicas de onde encontrar os cupons.
<b>Entradas</b>	Pontos acumulados e seleção de dicas
<b>Fonte</b>	Usuário e sistema de pontuação.
<b>Saídas</b>	Mensagem “Cupom adquirido com sucesso!” ou “Pontos insuficientes.”
<b>Ação</b>	<p>-O sistema exibe as dicas e seus respectivos valores em pontos.</p> <p>-O jogador escolhe uma dica.</p> <p>-O sistema verifica o saldo e realiza a troca.</p>

## 4.2 REQUISITOS NÃO FUNCIONAIS DE SOFTWARE



RFS01	
<b>Função</b>	Garantir tempo de resposta rápido em todas as ações do jogo
<b>Descrição</b>	As ações do sistema (login, seleção de categoria, resposta, troca de dicas) devem ocorrer em até 2 segundos.
<b>Entradas</b>	Ações do usuário
<b>Fonte</b>	Interação do usuário direto com a interface.
<b>Saídas</b>	Respostas rápidas de carregamento de tela e feedback imediato ao usuário.
<b>Ação</b>	<ul style="list-style-type: none"><li>-Otimizar o carregamento de dados.</li><li>-Minimizar tempo de resposta em operações simples.</li></ul>

RFS02	
<b>Função</b>	Assegurar interface intuitiva e fácil de usar.
<b>Descrição</b>	O sistema deve apresentar design limpo, com botões grandes e menus visuais
<b>Entradas</b>	Ações do usuário
<b>Fonte</b>	Usuário
<b>Saídas</b>	Feedback visual imediato (mudança de cor e mensagens de acertos ou erros.
<b>Ação</b>	<ul style="list-style-type: none"><li>-Disponibilizar ícones intuitivos.</li><li>-Evitar telas confusas.</li><li>-Garantir legibilidade</li></ul>

RFS03	
<b>Função</b>	Garantir o funcionamento em diferentes dispositivos

<b>Descrição</b>	O jogo deve funcionar corretamente em smartphones Android, navegadores e desktops.
<b>Entradas</b>	Execução via navegador ou aplicativo.
<b>Fonte</b>	Aparelho do usuário
<b>Saídas</b>	Execução estável a diferentes telas
<b>Ação</b>	<ul style="list-style-type: none"> <li>-Implementar design responsivo</li> <li>-Testar em múltiplos navegadores.</li> <li>-Ajustar o layout</li> </ul>

RFS04	
<b>Função</b>	Proteger os dados dos usuários
<b>Descrição</b>	As informações de login e pontuação devem ser armazenadas de forma segura
<b>Entradas</b>	Dados de login
<b>Fonte</b>	Usuário
<b>Saídas</b>	Confirmação de operação segura e proteção contra acesso indevido.
<b>Ação</b>	<ul style="list-style-type: none"> <li>-Proteger dados sensíveis</li> <li>-Validar formulários</li> <li>-Restringir acessos não autenticados</li> </ul>

RFS05	
<b>Função</b>	Garantir o funcionamento contínuo do sistema
<b>Descrição</b>	O jogo deve permanecer estável mesmo com uso prolongado ou múltiplas interações
<b>Entradas</b>	Ações simultâneas de usuários
<b>Fonte</b>	Sessões no jogo

<b>Saídas</b>	Sistema operando sem travamentos ou falhas
<b>Ação</b>	-Corrigir erros de exceção. -Monitorar desempenho.

RFS06	
<b>Função</b>	Facilitar a manutenção e evolução do sistema
<b>Descrição</b>	O código deve seguir boas práticas e permitir inclusão de novas perguntas e categorias
<b>Entradas</b>	Atualizações do sistema
<b>Fonte</b>	Equipe de desenvolvimento
<b>Saídas</b>	Sistema atualizado sem impacto nas funcionalidades existentes
<b>Ação</b>	-Documentar o código -Modularizar as telas, funções e categorias.

## 5. CASOS DE USO

### 1 - Responder Perguntas de uma Categoria

**Ator:** Usuário

**Fluxo normal:**

- 1 – Usuário faz login no sistema.
- 2 – Sistema exibe as categorias disponíveis: Matemática Financeira, História, Cultura e Mix.
- 3 – Usuário seleciona uma categoria.
- 4 – Sistema carrega uma pergunta da categoria escolhida e exibe quatro alternativas.
- 5 – Usuário escolhe uma alternativa.



6 – Sistema verifica se a resposta está correta.

7 – Se correta, soma 10 pontos e exibe a mensagem “Resposta correta!”.

**Extensões:**

1a – Se o usuário não escolher nenhuma categoria, o sistema solicita a seleção obrigatória.

2a – Se o usuário não escolher uma alternativa, o sistema exibe a mensagem “Selecione a resposta correta.”

3a – Se a resposta for incorreta, o sistema exibe “Resposta errada! Tente novamente.”

**2 - Trocar Pontos por Dicas**

**Ator:** Usuário

**Fluxo normal:**

1 – Usuário acessa a aba “Loja de Dicas”.

2 – Sistema exibe as dicas disponíveis e seus valores em pontos.

3 – Usuário seleciona a dica desejada.

4 – Sistema verifica se o usuário possui pontos suficientes.

5 – Se sim, desconta os pontos e libera a dica.

6 – Sistema exibe a mensagem “Dica adquirida com sucesso!”.

**Extensões:**

1a – Se o usuário não tiver pontos suficientes, o sistema exibe “Pontos insuficientes.”

2a – Se a loja estiver temporariamente indisponível, o sistema informa “Tente novamente mais tarde.”

**3a - Cadastrar Novo Usuário**

**Ator:** Usuário

**Fluxo normal:**

1 – Usuário acessa a tela de cadastro.

2 – Sistema solicita nome e senha.

3 – Usuário preenche os campos obrigatórios.

4 – Sistema valida os dados informados.

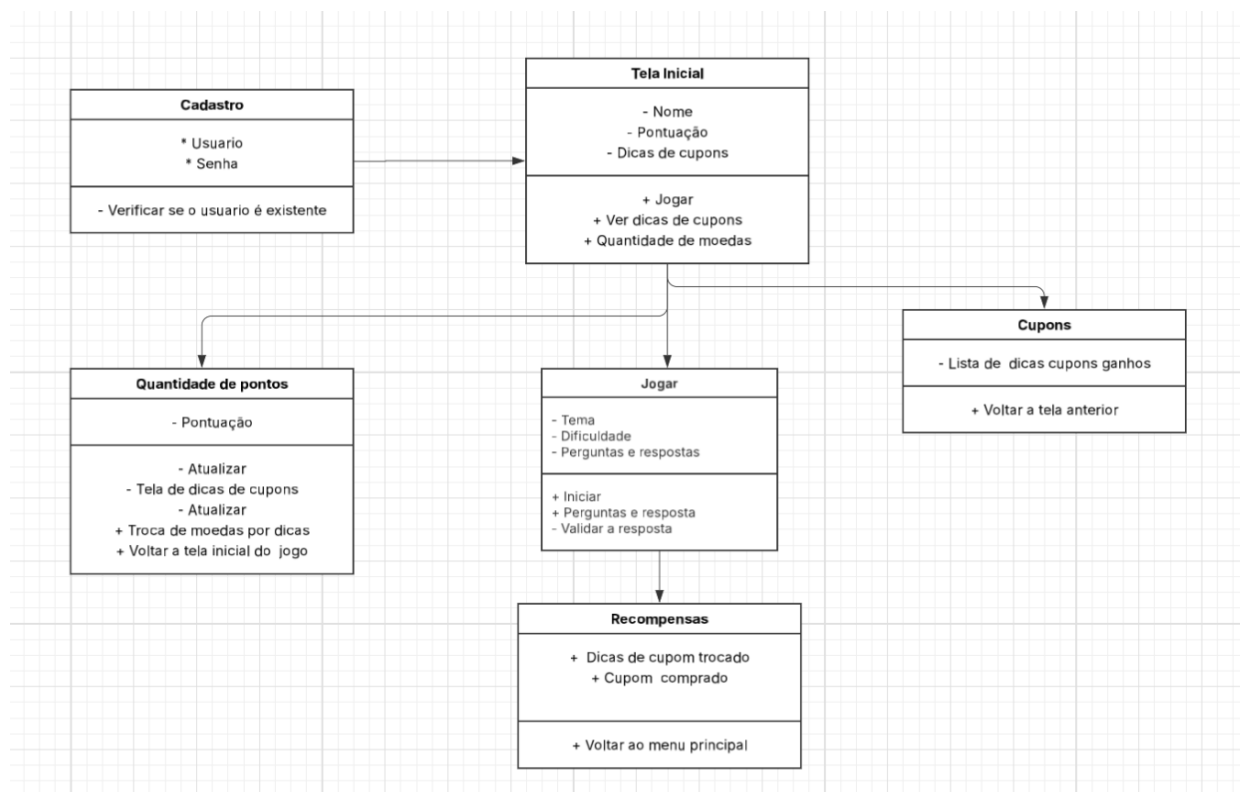
5 – Sistema grava o cadastro no banco de dados.

6 – Sistema exibe a mensagem “Cadastro realizado com sucesso!”.

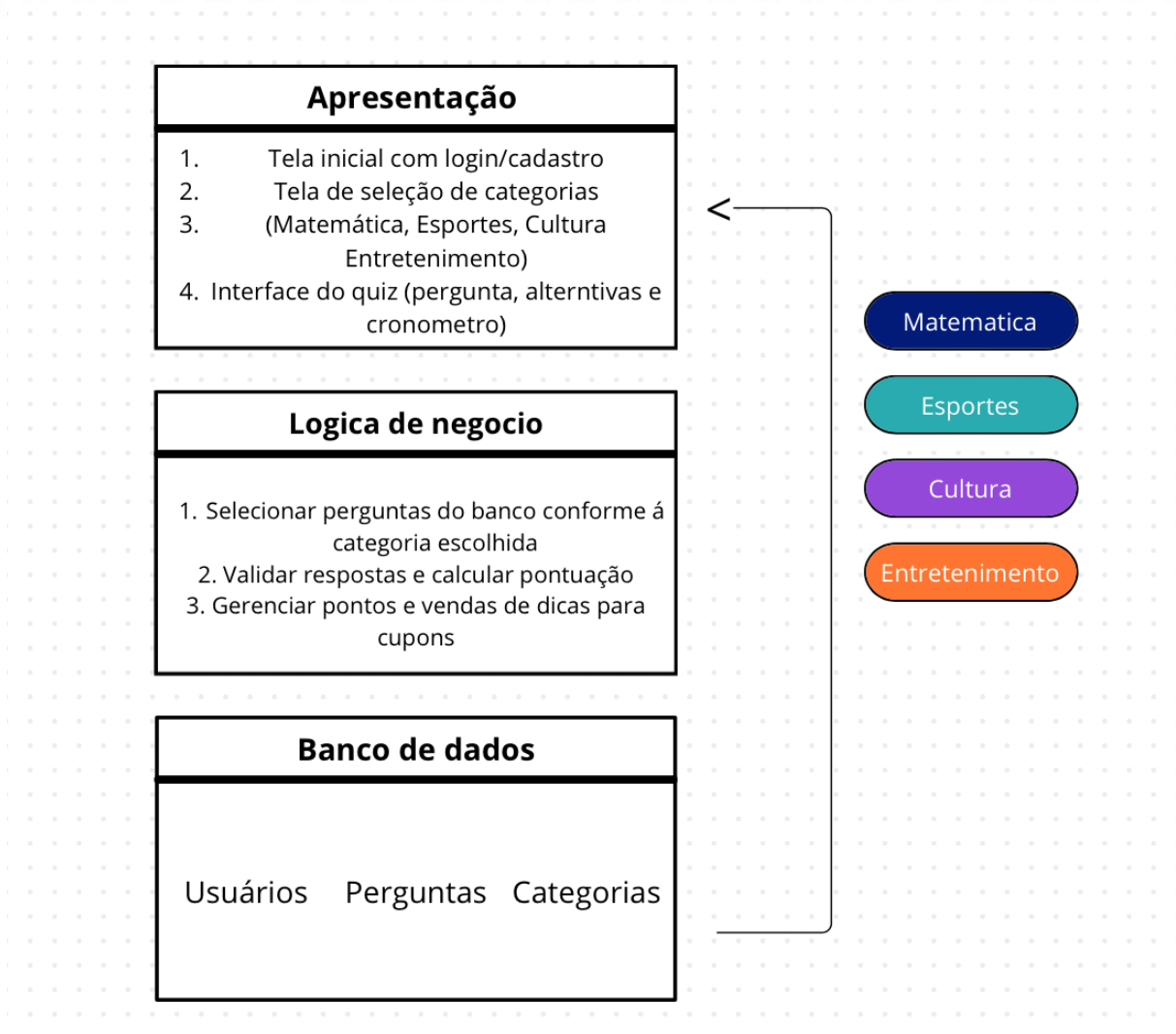
**Extensões:**

- 1a – Se algum campo obrigatório estiver vazio, o sistema solicita o preenchimento.
- 2a – Se o nome (username único) já estiver cadastrado, o sistema exibe a mensagem “username já existente.”

## 6. DIAGRAMA DE CLASSE



## 7. ARQUITETURA DO SISTEMA



## 8. REFERÊNCIAS BIBLIOGRÁFICAS

ETERMAX. *Perguntados*. [Jogo eletrônico]. Buenos Aires: Etermax, 2013. Disponível em: <https://www.perguntados.com>. Acesso em: 09 nov. 2025.

SBT. *Show do Milhão*. [Programa de televisão e jogo eletrônico]. São Paulo: Sistema Brasileiro de Televisão, 2001. Versão digital disponível em: <https://www.sbt.com.br>. Acesso em: 09 nov. 2025.

KAHOOT! AS. *Kahoot!*. [Plataforma de aprendizado interativo]. Oslo: Kahoot! AS, 2013. Disponível em: <https://www.kahoot.com>. Acesso em: 09 nov. 2025.



SOMMERVILLE, I. **Engenharia de Software**. 11ª Edição. São Paulo: Pearson Addison-Wesley, 2017.