UTFPR-UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
Bacharelado em Engenharia de Software - 6º Período

DISCIPLINA: *Oficina de Integração 1 - ES46F-ES61*
PROFESSOR: *Eduardo Cotrin Teixeira*

Documento de Projeto de Software

Sistema de controle para o projeto de extensão Lúdico

Gabriel Kenji Inoue
José Pedro Cunha do Amaral
Luis Augusto Casa Grande Fonseca
Pedro Lucas Vila Landgraf
Romulo Augusto Custodio Souza

Cornélio Procópio
2025

Sumário

1 INTRODUÇÃO 3
1.1 Contexto 3
1.2 Justificativa 4
1.3 Proposta 5
1.4 Organização do Documento 6
2 DESCRIÇÃO GERAL DO SISTEMA 7
2.1 Objetivos (Gerais e Específicos) 7
2.2 Limites e Restrições 7
2.3 Descrição dos Usuários do Sistema 7
3 DESENVOLVIMENTO DO PROJETO 9
3.1 Tecnologias e ferramentas 9
3.2 Metodologia de desenvolvimento 9
3.3 Cronograma realizado 10
4 REQUISITOS DO SISTEMA 14
4.1 Requisitos Funcionais 14
4.2 Requisitos Não-funcionais 15
4.3 Diagramas de Casos de Uso 16
4.4 Protótipos de Telas 17
5 ANÁLISE DO SISTEMA 20
6 IMPLEMENTAÇÃO 29
6.1 Descrição do código 29
6.2 Implantação 37
6.3 Telas principais 38
7 CONSIDERAÇÕES FINAIS 45
8 BIBLIOGRAFIA 46

1 Introdução

1.1 Contexto

O programa de extensão Lúdico tem como público-alvo os alunos da UTFPR e a comunidade externa, abrangendo principalmente Londrina, Cornélio Procópio e quaisquer cidades próximas que haja eventos realizados pelo Lúdico. O objetivo do programa é promover o desenvolvimento de análise crítica, raciocínio lógico, relações interpessoais e atuar como ferramenta de inserção cultural, e faz isso por meio de suas 3 atividades principais, as quais serão melhor desenvolvidas nos tópicos abaixo, elas são: o Board Games, o RPG e o Escape Room.

O ambiente do lúdico é gerenciado pelo coordenador, que em regra geral é quem costuma registrar os eventos, enquanto os alunos bolsistas auxiliam na gestão das atividades, principalmente no registro de empréstimos e pessoas nos eventos. Há cerca de 10 alunos bolsistas simultâneos, e com essa equipe, são gerenciados os eventos do Lúdico. Atualmente é realizado mensalmente um evento público em que são praticadas as atividades listadas anteriormente.

Após a apresentação do contexto geral do projeto de extensão, passa-se à análise das atividades realizadas na organização, que serão detalhadas a seguir.

O board games envolve jogos de mesa em geral, como por exemplo o Jenga ou xadrez, esses jogos podem possuir tabuleiro, cartas, placas, peças entre outros objetos, os quais serão utilizados para estabelecer e apoiar a dinâmica do jogo e entreter os jogadores. Durante o evento ocorre um empréstimo dos jogos para os participantes jogarem até o seu encerramento, sendo esse controle gerenciado pelos alunos bolsistas, que registram o horário e a pessoa que retira o jogo, assim garantindo a integridade e a preservação dos jogos.

Os RPGs geralmente também são jogos de mesa, porém há um aspecto de história e fantasia envolvida, além da possibilidade de interpretar personagens e compor um grupo com objetivos e que desenvolvem a história do mundo do jogo. Normalmente há 2 tipos de jogadores, sendo o mestre, que é o responsável pela narração da história e pode criar o mundo fictício, ficando a critério do sistema do jogo; e o jogador, o qual interpreta/controla seu personagem e escolhe quais ações irá tomar. Será necessário registrar os sistemas de RPG disponíveis, que tenha interessados em jogar ou, pois esses sistemas servirão de base para as campanhas.

Além disso, os RPGs possuem campanhas, as quais são histórias que podem ser criadas pelo mestre ou não, e que serão exploradas pelos jogadores. As campanhas podem possuir 1 ou mais seções, que são as partidas em que os jogadores se reúnem para jogar, ficando a critério do sistema e dos jogadores. Será necessário registrar as campanhas de RPG, vinculando-as a um sistema de RPG e aos jogadores participantes.

O escape room é um tipo de jogo de imersão, geralmente sendo realizado em uma sala tematizada, a qual terá enigmas a serem resolvidos e uma história própria (há diversas possibilidades e temas possíveis). Cada pista resolvida leva a outra, até que se chegue à solução final do jogo, possuindo

em média 60 minutos, mas podendo variar de acordo com os jogadores e a história. Será necessário registrar os escapes rooms disponíveis e as sessões agendadas.

1.2 Justificativa

Atualmente a gerência do projeto Lúdico possui um software legado de registro de informações feito em C# e Windows Forms criado em 2016 e sem suporte desde 2018. Em um levantamento de requisitos, o sistema possui principalmente 2 problemas em sua aplicação de gerenciamento, os quais são: limitação no sentido de usabilidade, como por exemplo, ter de escolher manualmente cada jogo de mesa que será incluso no evento que está sendo criado no registro; e o sistema atende apenas as necessidades do Board-game, devido de ser a única atividade do projeto na época do desenvolvimento do sistema. Um outro requisito problemático não funcional relacionado ao sistema legado é relacionado ao seu suporte, já que ele não se encontra documentado e contém um código de baixa legibilidade devido à falta de padrões consistentes e alto nível de acoplamento entre os módulos.

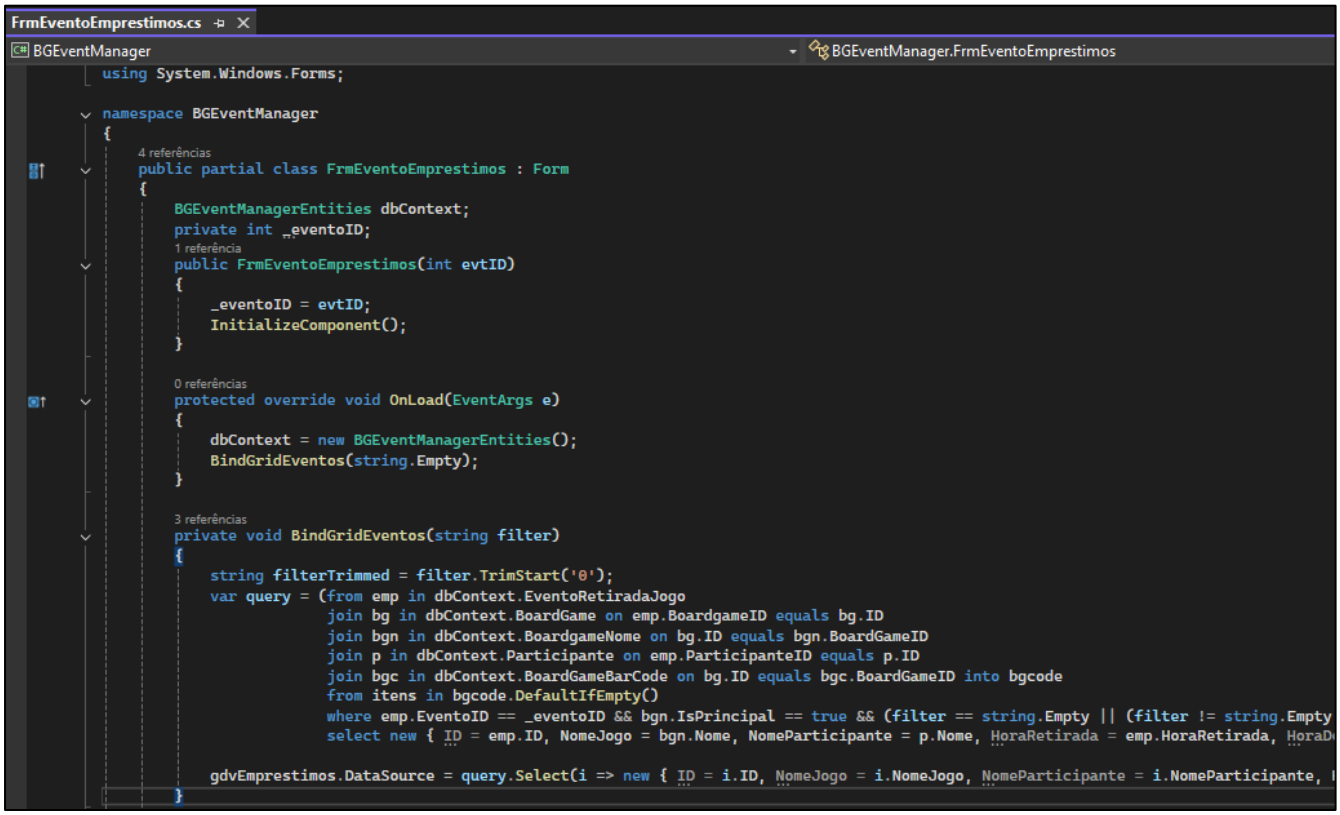


Figura 1 - Foto de uma parte do código do sistema legado para a consulta de participantes

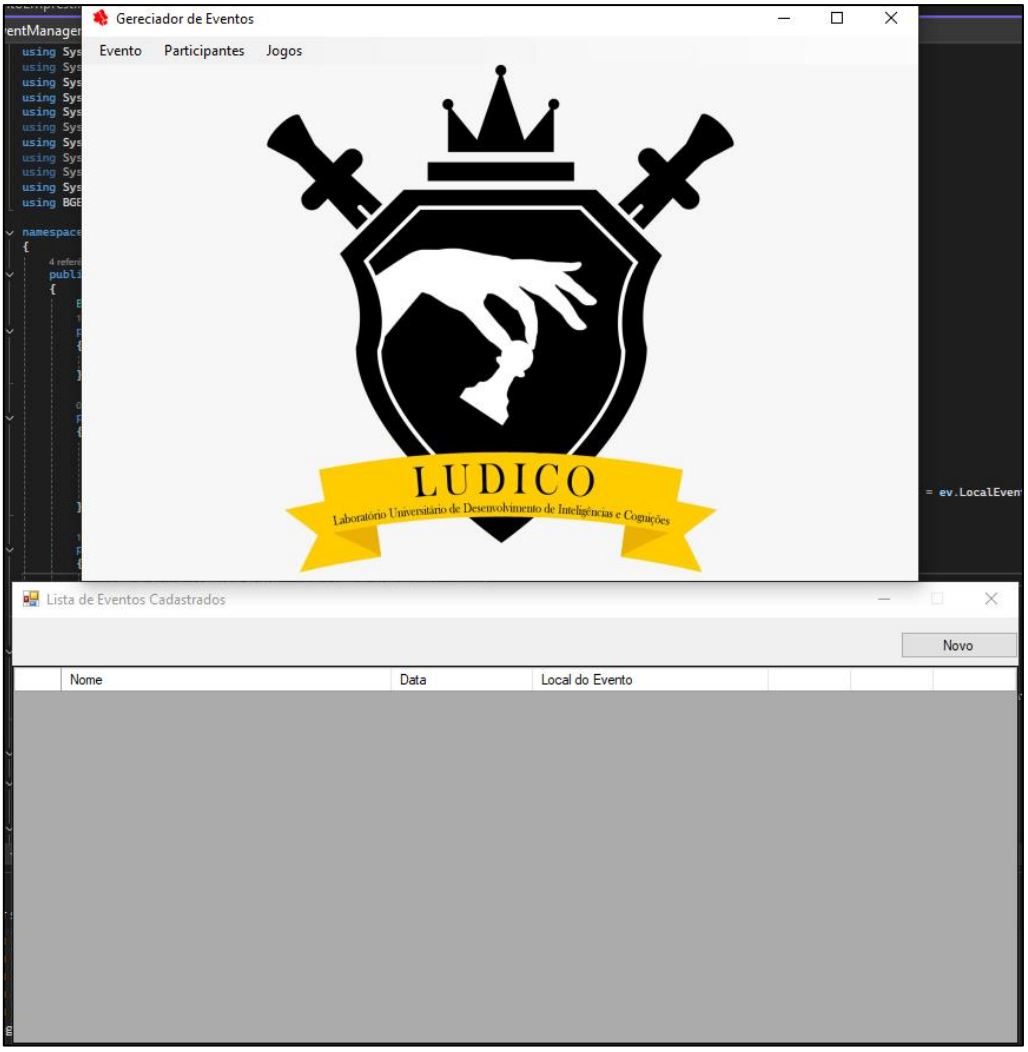


Figura 2 - Foto da interface do sistema legado de gerenciamento do Projeto Lúdico

No sistema os principais afetados são os responsáveis pela organização, porque precisam tomar muito cuidado e fazer esforço por conta das limitações impostas pelo sistema, além das pessoas que participam das outras 2 atividades que não são o board-games, pois, suas necessidades de registro de atividades, como por exemplo, histórico de presença ou quais atividades existem, não são atendidas e é necessário anotar manualmente os registros por outras formas, como o uso de papeis ou uma planilha Excel, assim descentralizando as informações.

1.3 Proposta

A proposta do projeto é desenvolver uma nova aplicação desktop voltada para registro de pessoas, jogos, empréstimos de jogos de mesa físicos que o lúdico possui, eventos e campanhas/seções referentes ao projeto do lúdico, para substituir o programa atual, que possui limitações de funcionalidade e usabilidade.

A nova solução será projetada com foco na melhoria da interface e usabilidade, adicionar funções pedidas pelo coordenador do projeto de extensão e facilitar o lançamento de informações para a nuvem.

O sistema terá impacto direto na organização do Lúdico e registro de campanhas de RPGs, oferendo uma ferramenta de registro mais prática, eficiente e aberta para melhorias futuras.

1.4 Organização do Documento

O documento está dividido em oito seções principais de forma que a abordagem seja lógica, clara e progressiva.

- Seção 1: apresenta a introdução ao projeto, que compreende uma breve apresentação do programa de extensão Lúdico, a necessidade de uma nova solução e os objetivos propostos da aplicação desenvolvida.

- Seção 2: disserta acerca do sistema de forma geral, contemplando os objetivos específicos, as limitações, os usuários-alvo e o benefício que a solução trará para o projeto.

- Seção 3: detalha o desenvolvimento do projeto, contemplando as tecnologias utilizadas, a metodologia – Scrum adaptado, os módulos do sistema e o cronograma de execução.

- Seção 4: será apresentada as especificações do sistema, funcionais e não funcionais, diagramas de caso de uso e protótipo das telas.

- Seção 5: demonstra e explica os diagramas do projeto: modelo de banco de dados, diagrama de classe e diagrama de atividades.

2 Descrição Geral do Sistema

2.1 Objetivos (Gerais e Específicos)

O objetivo geral é desenvolver um programa desktop com foco nas funcionalidades necessárias para a parte de RPG do Lúdico (sendo elas tanto registro de RPG quanto de campanhas, algo já explicado anteriormente) e na usabilidade do sistema, oferecendo uma solução mais eficiente e prática em relação ao sistema atual.

Objetivos específicos são: identificar pontos fracos que possam ser melhorados / corrigir suas limitações; projetar uma interface mais amigável para os responsáveis pela organização do projeto; implementar as funções requeridas pelo coordenador, sendo elas: confirmar antes de excluir um registro, permitir a organização de acordo com as colunas das tabelas de pesquisa e agilizar no processo de criação de evento; desenvolver um sistema que facilite a sincronização com a nuvem de forma simplificada e deixar a possibilidade de futuramente utilizar um banco de dados online.

2.2 Limites e Restrições

O projeto desenvolvido é um software desktop com banco de dados local, limitando a utilização do sistema para apenas um usuário e em uma única máquina sem acesso simultâneo a um mesmo banco de dados. Durante o planejamento do sistema, foi decidido disponibilizar o produto de forma gratuita e sem custo de manutenção na sua entrega da segunda semana de junho de 2025.

Em sua entrega final, o sistema foi projetado para executar no sistema operacional Windows sem a necessidade de se conectar à Internet para o seu funcionamento. É necessário ter no mínimo 2 GBs de memória RAM e 500 MB de armazenamento de espaço para a execução do sistema de forma plena e sem problemas.

Com a finalização do sistema, o código estará disponível na versão de produção para o projeto de extensão caso algum integrante do projeto queira melhorar ou aprimorar o sistema ou adicionar novas funcionalidades, além de configurar a disponibilidade de uso para outros bancos de dados para uma futura implementação de forma remota.

2.3 Descrição dos Usuários do Sistema

A solução impactará diretamente os participantes do projeto de extensão, sendo esses os usuários responsáveis pelo registro de participantes dos eventos. Embora o sistema funcione a partir do uso único de um usuário em uma máquina e sem a divisão de permissões e usuários no sistema, facilitando a centralização e registro de informação para um participante no evento. Os papéis foram divididos em três:

- Coordenador do projeto: Responsável pela supervisão do sistema, aprovando novos recursos e analisando os dados, além de acessar as informações do banco de dados.

- Monitores do projeto: Registram os participantes nos eventos, inserindo os dados e organizando o sistema.

- Participantes do evento: Embora não utilize o sistema, o público tem como função passar as suas informações para os monitores registrarem no sistema.

3 Desenvolvimento do Projeto

3.1 Tecnologias e ferramentas

Para o desenvolvimento do projeto foi decidido utilizar a mesma base do sistema anterior, sendo o SQLite para o banco de dados e comunicação com o backend, esse sistema permite o armazenamento de dados de forma local sem a instalação de uma biblioteca na execução final do aplicativo.

O projeto utilizou C# para a lógica e abstração de informações para o backend pela modelagem OOP (Orientada a Objetos), que esse facilita a comunicação a abstração de informações e a comunicação com o SQLite e o Windows Forms para a interface do usuário

No gerenciamento de projeto, foi decidido o uso da IDE Visual Studio Code, que facilita a integração com o GitHub para controle de versão e distribuição, e por fim, o NuGet como gerenciador de pacotes para adição de bibliotecas de terceiros de forma efetiva.

3.2 Metodologia de desenvolvimento

Para o desenvolvimento do projeto, foi utilizada a metodologia ágil Scrum, com algumas adaptações devido ao tempo de desenvolvimento disponível para os integrantes do grupo. Não foram realizadas reuniões diárias (daily scrum); em vez disso, o grupo optou por trocas de feedback e alinhamentos por meio do canal de comunicação do WhatsApp, além de reuniões virtuais pelo Discord às quartas-feiras e sábados. A definição e acompanhamento de tarefas semanais foram apoiados pela ferramenta Notion, enquanto o progresso foi monitorado pela plataforma Jira. O projeto foi dividido em 4 sprints com duração de 1 a 2 semanas, iniciando em 04/05/2025 e seguindo até 14/06/2025.

O Scrum, como metodologia ágil, prevê um conjunto de práticas voltadas para o desenvolvimento iterativo e incremental, com foco na entrega contínua de valor, colaboração do time e adaptação rápida a mudanças. Dentre seus principais pilares estão a transparência, a inspeção e a adaptação.

Em relação aos papéis do Scrum, foram definidos da seguinte forma:

Luís Augusto atua como Scrum Master e Product Owner (Gestor do Produto), sendo responsável por garantir que a metodologia seja seguida, remover impedimentos e facilitar a comunicação entre os membros, além de priorizar e validar as funcionalidades a serem desenvolvidas;

Gabriel Kenji é arquiteto e engenheiro, participando ativamente das decisões estruturais do sistema;

José Pedro é arquiteto e desenvolvedor, contribuindo tanto na modelagem quanto na implementação;

Pedro Lucas e Romulo Augusto atuam como testers e desenvolvedores, testando e construindo as funcionalidades da aplicação.

Essa divisão respeita os princípios do Scrum, mesmo que adaptado, garantindo papéis claros dentro do time e favorecendo o andamento organizado das atividades do projeto. O desenvolvimento do software teve como embasamento dois protótipos principais, sendo eles o sistema legado utilizado anteriormente pelo projeto de extensão Lúdico, e o outro protótipo de telas para apoiar no desenvolvimento da interface do sistema.

O projeto terá os seguintes módulos:

- Módulo Usuários: Para o registro de pessoas com interação ao sistema

- Módulo Jogos: Para o registro de jogos utilizados nos eventos

- Módulo Eventos: Para o registro de eventos, permitindo registrar a presença de pessoas, os empréstimos de jogos e os jogos disponíveis para o empréstimo.

- Módulo Escape Room: Para o registro de Escape Room e seus participantes da sessão

- Módulo RPG: Para o registro de jogos de RPG utilizados nos eventos, permitindo o registro de campanhas e seções.

Entre as ferramentas utilizadas para a organização do projeto, destaca-se o uso do **Jira**, que permite o estabelecimento de *sprints*, a definição de tarefas e a distribuição eficiente entre os membros da equipe.

O **Scrum Master** é responsável pela divisão e gestão das sprints, enquanto o **Arquiteto** do projeto atua como guia técnico, assegurando que todos sigam os padrões definidos a partir do Sprint 1.

3.3 Cronograma realizado

Primeiramente, será apresentado o cronograma base, criado respeitando o cronograma inicial fornecido pelo professor da disciplina de Oficina de Integração. O cronograma foi dividido em semanas com o objetivo de separar o trabalho em problemas menores, facilitando o acompanhamento sem tornar o gerenciamento muito complexo.

- 29/03/25: Definição do grupo, divisão de tarefas e das funções, e escolha do projeto de extensão.

- 05/04/25: Definição do produto a ser desenvolvido, ferramentas e levantamento de requisitos.

- 12/04/25: Escolha da metodologia, documentação dos requisitos funcionais e não funcionais, criação de casos de uso e protótipos de tela.

• 19/04/25: Criação da modelagem do banco de dados, diagrama de classes e diagrama de atividades.

• 26/04/25: Estruturação do projeto para desenvolvimento dos primeiros módulos.

• 03/05/25: Criação do módulo de instituições.

• 10/05/25: Criação do módulo de participantes.

• 17/05/25: Criação do módulo de jogos.

• 24/05/25: Criação do módulo de eventos.

• 31/05/25: Criação do módulo do RPG.

• 07/06/25: Criação do módulo de Escape Room.

• 14/06/25: Documentação, finalização e ajustes finais.

• 21/06/25: Entrega do projeto para o projeto de extensão e oficina de integração.

Logo em seguida, será apresentado o cronograma que foi implementado e seguido por meio da plataforma Jira, com foco apenas no aspecto de desenvolvimento da aplicação, a implementação do Jira foi explicada anteriormente no tópico de metodologia.

3.4 Ferramenta de organização Jira

Sprint 1: Estabeleceu a base estrutural do projeto. Apesar de alguns desafios iniciais de organização, foi concluída com sucesso e serviu como referência para as demais sprints. Durante essa sprint foram definidos e criados os blocos de instituição e participante.

Resumo	Status	Sprint ↑	Data de entrega
Criação da tela inicial com os menus superiores	CONCLUÍDO	Sprint 1	10 de mai. de 2025
Criação da Tela de visualização de instituições registra...	CONCLUÍDO	Sprint 1	17 de mai. de 2025
Realização de função de Pesquisa de instituições regist...	CONCLUÍDO	Sprint 1	17 de mai. de 2025
Criar funcionalidade de edição de instituição	CONCLUÍDO	Sprint 1	17 de mai. de 2025
Criar funcionalidade de deletar instituição	CONCLUÍDO	Sprint 1	17 de mai. de 2025
Criar tela de visualização de participantes	CONCLUÍDO	Sprint 1	17 de mai. de 2025
Criar tela de criação de participante	CONCLUÍDO	Sprint 1	17 de mai. de 2025
Criar funcionalidade de pesquisa de participante	CONCLUÍDO	Sprint 1	17 de mai. de 2025
Criar funcionalidade de edição de participante	CONCLUÍDO	Sprint 1	17 de mai. de 2025

Figura 3 - Cards do Jira realizados na Sprint 1

Sprint 2: Com tarefas bem definidas, teve alguns problemas durante o desenvolvimento, mas foi finalizada. A equipe estava mais preparada e com a estrutura pronta. Foram criados os blocos de jogos, locais de eventos e gerenciamento de eventos.

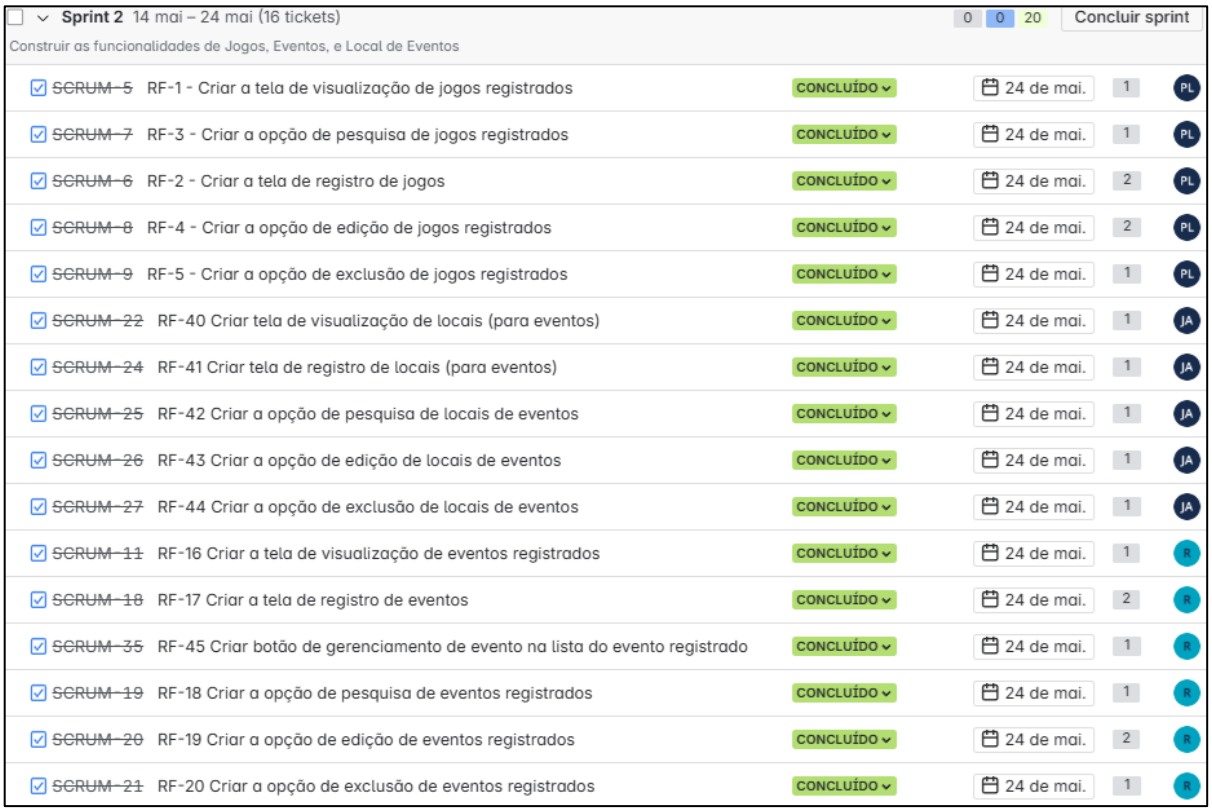
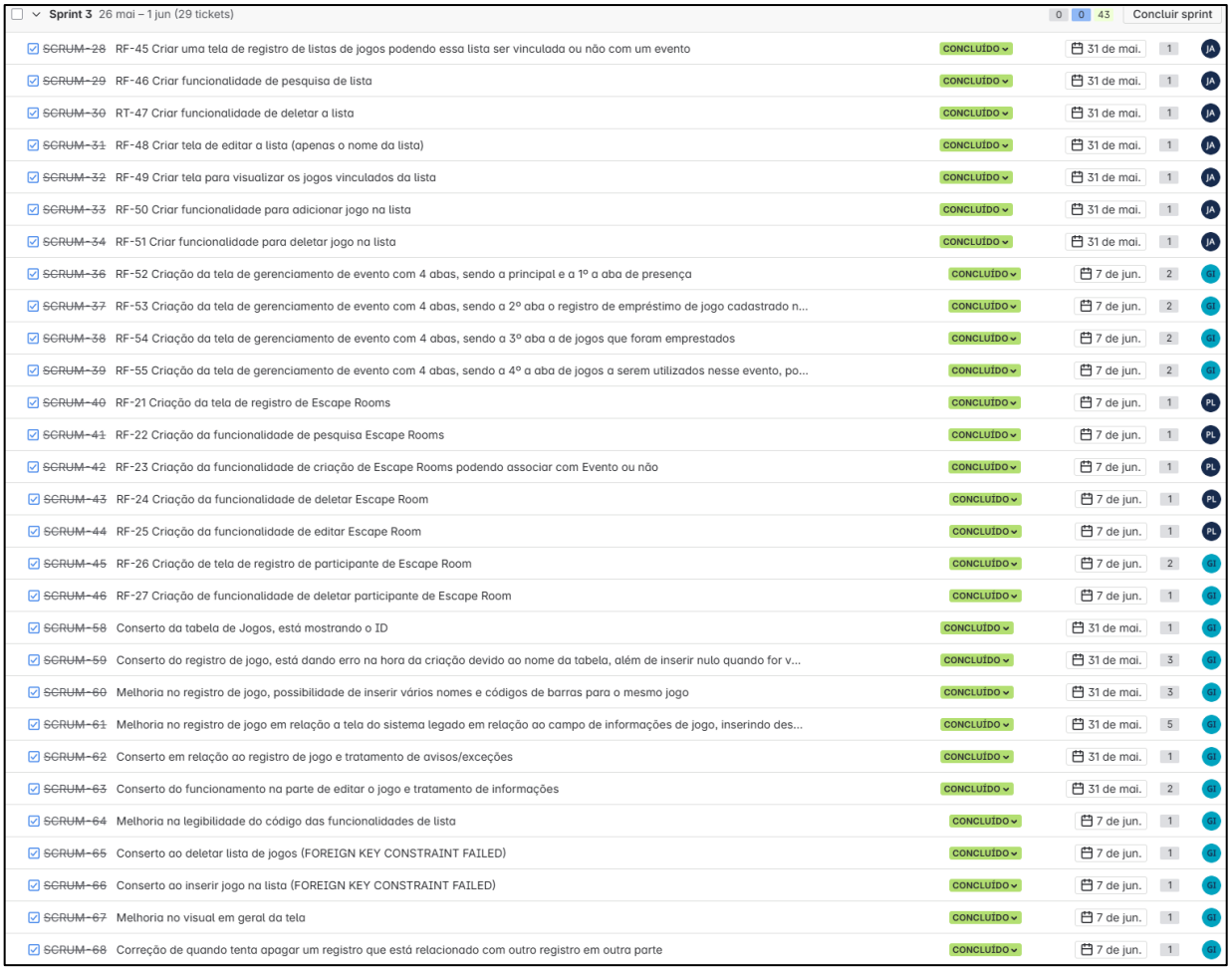


Figura 4 - Cards do Jira realizados na Sprint 2

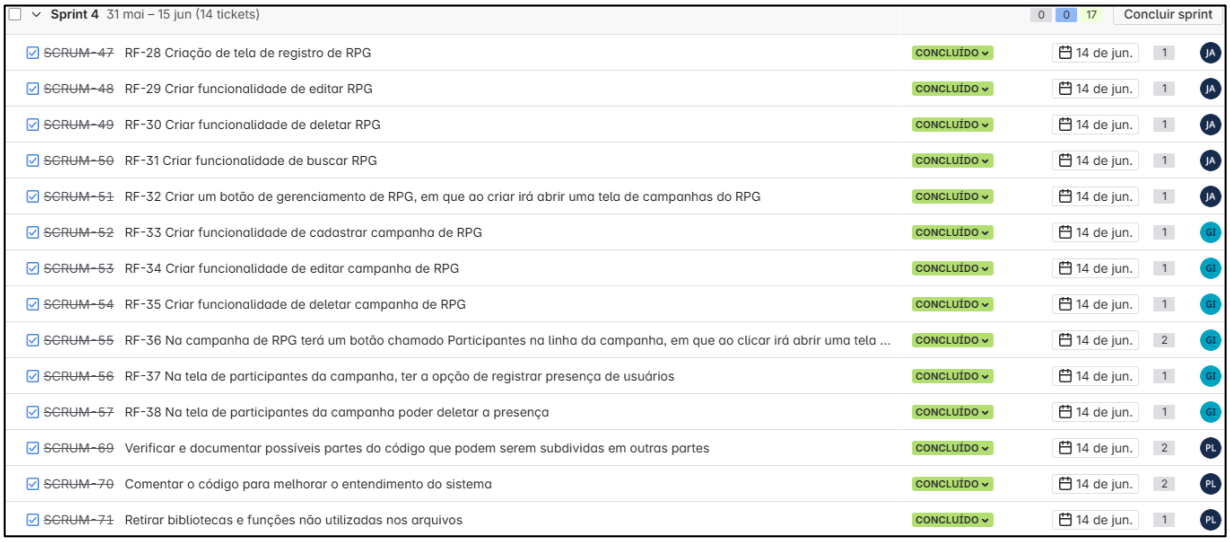
Sprint 3: A mais longa e exigente. A equipe adiantou tarefas para aliviar a próxima sprint. Foram criados os blocos de lista de predefinição, gerenciamento de evento e escape rooms, além da correção de falhas.



Sprint 3	26 mai – 1 jun (29 tickets)				
SCRUM-28	RF-45 Criar uma tela de registro de listas de jogos podendo essa lista ser vinculada ou não com um evento	CONCLUÍDO	31 de mai.	1	JA
SCRUM-29	RF-46 Criar funcionalidade de pesquisa de lista	CONCLUÍDO	31 de mai.	1	JA
SCRUM-30	RT-47 Criar funcionalidade de deletar a lista	CONCLUÍDO	31 de mai.	1	JA
SCRUM-31	RF-48 Criar tela de editar a lista (apenas o nome da lista)	CONCLUÍDO	31 de mai.	1	JA
SCRUM-32	RF-49 Criar tela para visualizar os jogos vinculados da lista	CONCLUÍDO	31 de mai.	1	JA
SCRUM-33	RF-50 Criar funcionalidade para adicionar jogo na lista	CONCLUÍDO	31 de mai.	1	JA
SCRUM-34	RF-51 Criar funcionalidade para deletar jogo na lista	CONCLUÍDO	31 de mai.	1	JA
SCRUM-36	RF-52 Criação da tela de gerenciamento de evento com 4 abas, sendo a principal e a 1º a aba de presença	CONCLUÍDO	7 de jun.	2	GI
SCRUM-37	RF-53 Criação da tela de gerenciamento de evento com 4 abas, sendo a 2º aba o registro de empréstimo de jogo cadastrado n...	CONCLUÍDO	7 de jun.	2	GI
SCRUM-38	RF-54 Criação da tela de gerenciamento de evento com 4 abas, sendo a 3º aba a de jogos que foram emprestados	CONCLUÍDO	7 de jun.	2	GI
SCRUM-39	RF-55 Criação da tela de gerenciamento de evento com 4 abas, sendo a 4º a aba de jogos a serem utilizados nesse evento, po...	CONCLUÍDO	7 de jun.	2	GI
SCRUM-40	RF-21 Criação da tela de registro de Escape Rooms	CONCLUÍDO	7 de jun.	1	PL
SCRUM-41	RF-22 Criação da funcionalidade de pesquisa Escape Rooms	CONCLUÍDO	7 de jun.	1	PL
SCRUM-42	RF-23 Criação da funcionalidade de criação de Escape Rooms podendo associar com Evento ou não	CONCLUÍDO	7 de jun.	1	PL
SCRUM-43	RF-24 Criação da funcionalidade de deletar Escape Room	CONCLUÍDO	7 de jun.	1	PL
SCRUM-44	RF-25 Criação da funcionalidade de editar Escape Room	CONCLUÍDO	7 de jun.	1	PL
SCRUM-45	RF-26 Criação de tela de registro de participante de Escape Room	CONCLUÍDO	7 de jun.	2	GI
SCRUM-46	RF-27 Criação de funcionalidade de deletar participante de Escape Room	CONCLUÍDO	7 de jun.	1	GI
SCRUM-58	Conserto da tabela de Jogos, está mostrando o ID	CONCLUÍDO	31 de mai.	1	GI
SCRUM-59	Conserto do registro de jogo, está dando erro na hora da criação devido ao nome da tabela, além de inserir nulo quando for v...	CONCLUÍDO	31 de mai.	3	GI
SCRUM-60	Melhoria no registro de jogo, possibilidade de inserir vários nomes e códigos de barras para o mesmo jogo	CONCLUÍDO	31 de mai.	3	GI
SCRUM-61	Melhoria no registro de jogo em relação a tela do sistema legado em relação ao campo de informações de jogo, inserindo des...	CONCLUÍDO	31 de mai.	5	GI
SCRUM-62	Conserto em relação ao registro de jogo e tratamento de avisos/exceções	CONCLUÍDO	31 de mai.	1	GI
SCRUM-63	Conserto do funcionamento na parte de editar o jogo e tratamento de informações	CONCLUÍDO	31 de mai.	2	GI
SCRUM-64	Melhoria na legibilidade do código das funcionalidades de lista	CONCLUÍDO	7 de jun.	1	GI
SCRUM-65	Conserto ao deletar lista de jogos (FOREIGN KEY CONSTRAINT FAILED)	CONCLUÍDO	7 de jun.	1	GI
SCRUM-66	Conserto ao inserir jogo na lista (FOREIGN KEY CONSTRAINT FAILED)	CONCLUÍDO	7 de jun.	1	GI
SCRUM-67	Melhoria no visual em geral da tela	CONCLUÍDO	7 de jun.	1	GI
SCRUM-68	Correção de quando tenta apagar um registro que está relacionado com outro registro em outra parte	CONCLUÍDO	7 de jun.	1	GI

Figura 5 - Cards do Jira da Sprint 3

Sprint 4: Com menos tarefas, tudo foi concluído e entregue. Foram finalizados os blocos de RPG e campanhas, com ajustes e melhorias finais na aplicação



Sprint 4	31 mai – 15 jun (14 tickets)				
SCRUM-47	RF-28 Criação de tela de registro de RPG	CONCLUÍDO	14 de jun.	1	JA
SCRUM-48	RF-29 Criar funcionalidade de editar RPG	CONCLUÍDO	14 de jun.	1	JA
SCRUM-49	RF-30 Criar funcionalidade de deletar RPG	CONCLUÍDO	14 de jun.	1	JA
SCRUM-50	RF-31 Criar funcionalidade de buscar RPG	CONCLUÍDO	14 de jun.	1	JA
SCRUM-51	RF-32 Criar um botão de gerenciamento de RPG, em que ao criar irá abrir uma tela de campanhas do RPG	CONCLUÍDO	14 de jun.	1	JA
SCRUM-52	RF-33 Criar funcionalidade de cadastrar campanha de RPG	CONCLUÍDO	14 de jun.	1	GI
SCRUM-53	RF-34 Criar funcionalidade de editar campanha de RPG	CONCLUÍDO	14 de jun.	1	GI
SCRUM-54	RF-35 Criar funcionalidade de deletar campanha de RPG	CONCLUÍDO	14 de jun.	1	GI
SCRUM-55	RF-36 Na campanha de RPG terá um botão chamado Participantes na linha da campanha, em que ao clicar irá abrir uma tela ...	CONCLUÍDO	14 de jun.	2	GI
SCRUM-56	RF-37 Na tela de participantes da campanha, ter a opção de registrar presença de usuários	CONCLUÍDO	14 de jun.	1	GI
SCRUM-57	RF-38 Na tela de participantes da campanha poder deletar a presença	CONCLUÍDO	14 de jun.	1	GI
SCRUM-69	Verificar e documentar possíveis partes do código que podem serem subdividas em outras partes	CONCLUÍDO	14 de jun.	2	PL
SCRUM-70	Comentar o código para melhorar o entendimento do sistema	CONCLUÍDO	14 de jun.	2	PL
SCRUM-71	Retirar bibliotecas e funções não utilizadas nos arquivos	CONCLUÍDO	14 de jun.	1	PL

Figura 6 - Cards do Jira da Sprint 4

4 Requisitos do Sistema

4.1 Requisitos Funcionais

ID	Funcionalidade	Prioridade
RF-1	O sistema ter uma tela de visualização de jogos registrados	Essencial
RF-2	O sistema ter uma tela de registro de jogos	Essencial
RF-3	O sistema disponibilizar a pesquisa de jogos registrados	Importante
RF-4	O sistema disponibilizar a edição de jogos registrados	Essencial
RF-5	O sistema disponibilizar a exclusão de jogos registrados	Essencial
RF-6	O sistema ter uma tela de visualização de participantes registrados	Essencial
RF-7	O sistema ter uma tela de registro de participantes	Essencial
RF-8	O sistema disponibilizar a pesquisa de participantes registrados	Importante
RF-9	O sistema disponibilizar a edição de participantes registrados	Desejável
RF-10	O sistema disponibilizar a exclusão de participantes registrados	Desejável
RF-11	O sistema ter uma tela de visualização de instituições organizacionais registradas, que são relacionadas aos participantes ou aos eventos	Importante
RF-12	O sistema ter uma tela de registro de instituições	Desejável
RF-13	O sistema disponibilizar a pesquisa de instituições registradas	Desejável
RF-14	O sistema disponibilizar a edição de instituições registradas	Desejável
RF-15	O sistema disponibilizar a exclusão de instituições registradas	Desejável
RF-16	O sistema ter uma tela de visualização de eventos registrados	Essencial
RF-17	O sistema ter uma tela de registro de eventos	Essencial
RF-18	O sistema disponibilizar a pesquisa de eventos registrados	Importante
RF-19	O sistema disponibilizar a edição de eventos registrados	Desejável
RF-20	O sistema disponibilizar a exclusão de eventos registrados	Desejável
RF-21	Durante o gerenciamento do evento, o sistema poder registrar presença de um novo ou participante já existente	Importante
RF-22	Durante o gerenciamento do evento, o sistema poder registrar os jogos disponíveis	Importante
RF-23	Durante o gerenciamento do evento, o sistema poder emprestar jogos disponíveis para um participante do evento.	Importante
RF-24	Durante o gerenciamento do evento, disponibilizar os jogos emprestados no evento e o seu tempo de entrada e saída do empréstimo.	Desejável
RF-25	O sistema ter uma tela de registro de Escape Room	Importante
RF-26	O sistema disponibilizar a pesquisa de Escape Room registrados	Importante
RF-27	O sistema disponibilizar a edição de Escape Room registrados	Desejável
RF-28	O sistema disponibilizar a exclusão de Escape Room registrados	Desejável
RF-29	Durante o gerenciamento do Escape Room, ser possível o registro de participantes	Desejável
RF-30	O sistema ter uma tela de registro de RPG	Importante
RF-31	O sistema disponibilizar a pesquisa de RPG registrados	Importante
RF-32	O sistema disponibilizar a edição de RPG registrados	Desejável

RF-33	O sistema disponibilizar a exclusão de RPG registrados	Desejável
RF-34	O sistema ter uma tela de campanhas de RPG no gerenciamento de RPG, as campanhas são histórias interativas onde jogadores interpretam personagens em aventuras contínuas.	Importante
RF-35	O sistema possibilitar a pesquisa de campanhas de RPG no gerenciamento de RPG	Desejável
RF-36	O sistema possibilitar a edição de campanhas de RPG no gerenciamento de RPG	Desejável
RF-37	O sistema possibilitar a exclusão de campanhas de RPG no gerenciamento de RPG	Desejável
RF-38	Dentro da tela de campanhas de RPG, possibilitar o registro de integrantes da campanha	Desejável
RF-39	Dentro da tela de campanhas de RPG, possibilitar a exclusão de integrantes da campanha	Desejável

4.2 Requisitos Não-funcionais

ID	**Requisito**	**Categoria**
NF-1	O sistema permitir o cadastro de jogos por código de barras	Padronização
NF-2	O sistema permitir a busca de jogos por código de barras	Padronização
NF-3	O tempo de resposta do banco de dados suficientemente rápido nos registros e consultas para evitar transtornos na sua utilização	Desempenho
NF-4	Organizar as consultas de dados ao clicar em alguma coluna	Usabilidade
NF-5	O sistema deve pedir confirmação antes de excluir algum registro	Usabilidade
NF-6	O sistema deve permitir que se crie, altere e exclua uma predefinição de evento	Usabilidade
NF-7	A interface do sistema deve ser dividida por funções para que seu uso seja mais intuitivo	Usabilidade
NF-8	O sistema de banco de dados deve ser compatível com o sistema legado	Compatibilidade
NF-9	A interface deve ser visualmente agradável para o usuário, facilitando sua interação	Usabilidade
NF-10	O sistema deve permitir backup para o banco de dados	Recursos
NF-11	O sistema deve carregar qualquer tela suficientemente rápido para evitar transtornos para o usuário	Desempenho

4.3 Diagramas de Casos de Uso

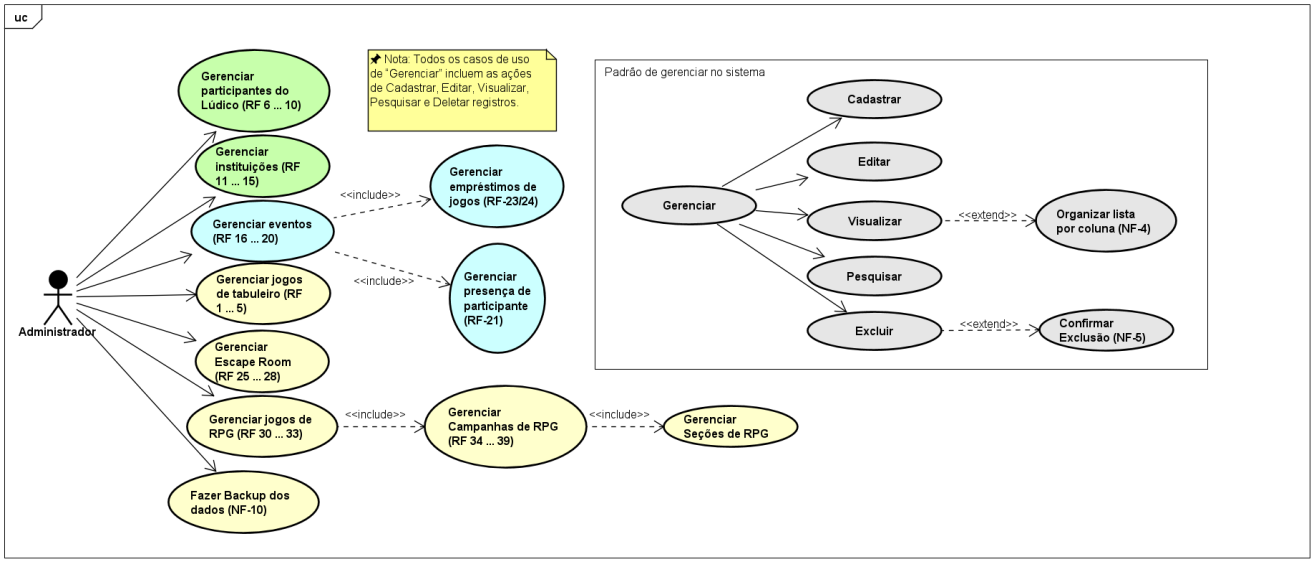


Figura 7 - Diagrama de Caso de Uso desenvolvido

Para sintetizar os diagramas, foi criado um diagrama de casos de uso que representa o sistema por inteiro, e há referência dos requisitos funcionais e não funcionais após o nome do caso de uso, prevendo as funcionalidades que o usuário pode realizar durante o uso do sistema.

4.4 Protótipos de Telas

As imagens printadas estão levemente mais escuras que o original, desenvolvendo no Figma, podendo ser acessado a partir do link do protótipo: Figma do Protótipo Lúdico.

Figura 8 - Tela Inicial

Tela Inicial

Essa tela tem como objetivo ser o centro do programa, ela poderá chamar as telas principais de cada módulo e todas as telas principais terão como chamar ela quando retornar.

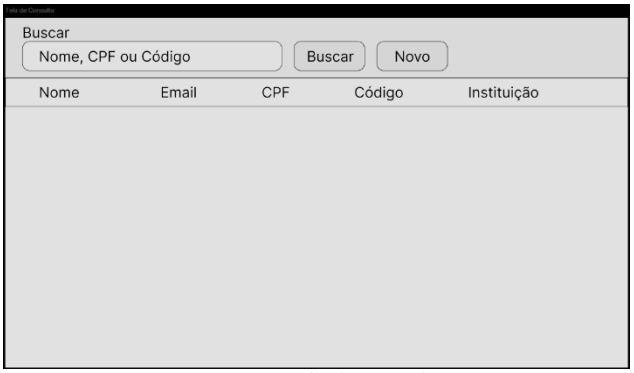


Figura 9 - Tela de Consulta

Tela de Consulta

Essa tela tem como objetivo consultar os participantes cadastrados, ela poderá chamar as telas relacionadas a consulta, e poderá ser chamada por elas e a tela inicial.

Figura 10 - Tela de Cadastro

Tela de Cadastro

Essa tela tem como objetivo cadastrar os participantes, ela poderá chamar a tela de consulta de participantes cadastrados, a qual poderá a chamar.

Figura 11 - Tela de Cadastro de Evento

Tela de Cadastro de Evento

Essa tela tem como objetivo cadastrar um evento que irá ocorrer, seja na universidade ou local exterior a ela, ela poderá chamar a tela de gerenciamento de evento e vice-versa.

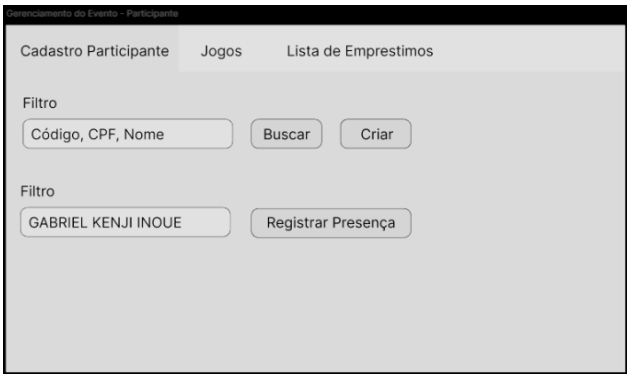


Figura 12 - Gerenciamento do Evento - Participante

Gerenciamento do Evento - Participante

Gerenciamento do Evento - Participante

Essa tela tem como objetivo registrar presença e/ou servir de atalho para a criação de um novo participante durante o evento, ela poderá chamar a tela de cadastro de participantes, as telas relacionadas ao evento e a tela inicial, e o contrário também se aplica.

Figura 13 - Gerenciamento do Evento - Jogos

Gerenciamento do Evento - Jogos

Essa tela tem como objetivo cadastrar/administrar os jogos trazidos durante o evento, ela poderá chamar a tela de cadastro de jogos, as telas relacionadas aos eventos e a tela inicial, e o contrário também se aplica.

Figura 14 - Gerenciamento do Evento – Lista de Jogos

Gerenciamento do Evento - Lista de Jogos

Essa tela tem como objetivo adicionar jogos novos ou cadastrados como uma predefinição de evento, de forma que fique menos trabalhoso registrar um evento novo, ela poderá chamar a tela de gerenciamento de evento e a tela inicial, e vice-versa.

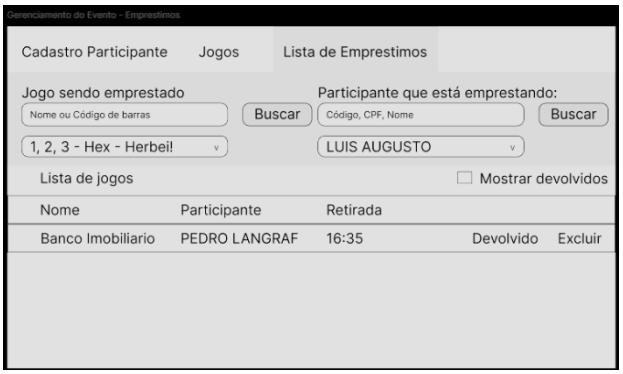


Figura 15 - Gerenciamento do Evento - Empréstimos

Gerenciamento do Evento - Empréstimos

Essa tela tem como objetivo gerenciar, adicionar e marcar como devolvido os empréstimos de jogos de tabuleiro durante um evento, ela poderá chamar as telas relacionadas de evento e a tela inicial, e vice-versa.

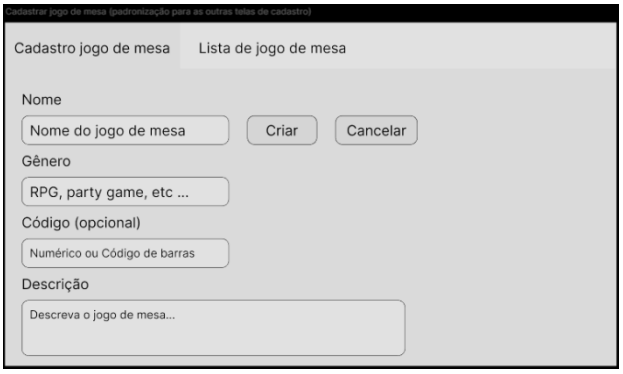


Figura 16 - Cadastrar jogo de mesa

Cadastrar jogo de mesa

Essa tela tem como objetivo cadastrar novos jogos de tabuleiro, tanto os de RPG quanto os outros como o Jenga ou Banco Imobiliário, ela poderá chamar a tela de lista de jogos, o pop-up da própria tela e a inicial, e vice-versa.

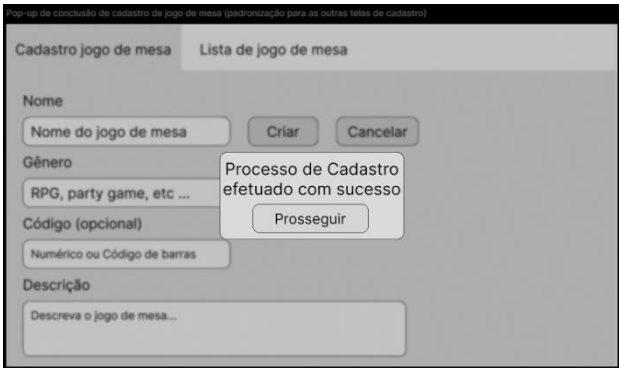


Figura 17 - Pop-up de conclusão de cadastro de jogo de mesa

Pop-up de conclusão de cadastro de jogo de mesa

Essa tela tem como objetivo passar o feedback de sucesso do cadastro, ela poderá chamar a tela de cadastro de jogos e vice-versa.

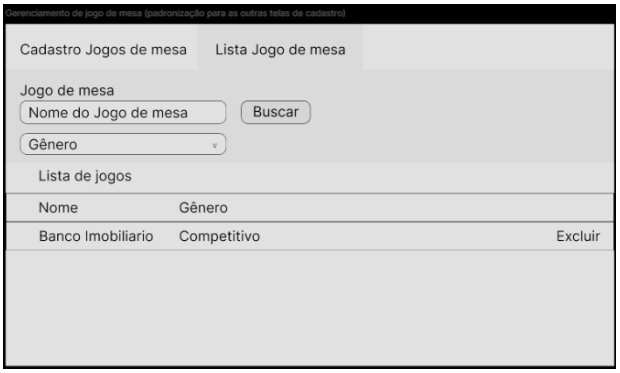


Figura 18 - Gerenciamento de jogo de mesa

Gerenciamento de jogo de mesa

Essa tela tem como objetivo a listagem/pesquisa/exclusão de jogos de mesa e rpgs, ela poderá chamar a tela de cadastro, a de pop-up da própria tela e a tela inicial, e vice-versa.

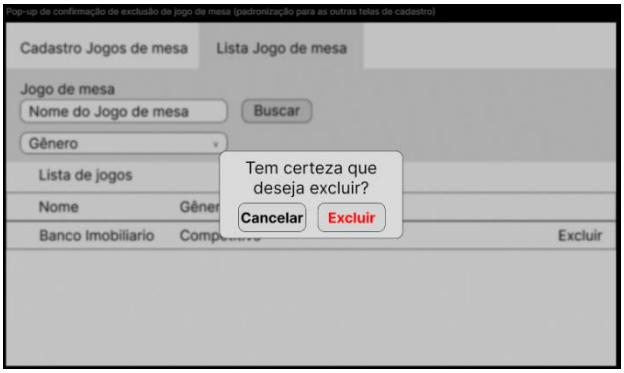


Figura 19 - Pop-up de confirmação de exclusão de jogo de mesa

Pop-up de confirmação de exclusão de jogo de mesa

Essa tela tem como objetivo confirmar que o processo de exclusão é intencional, ela poderá chamar a tela de lista de jogos e vice-versa.

5 Análise do Sistema

Modelo do Banco de Dados

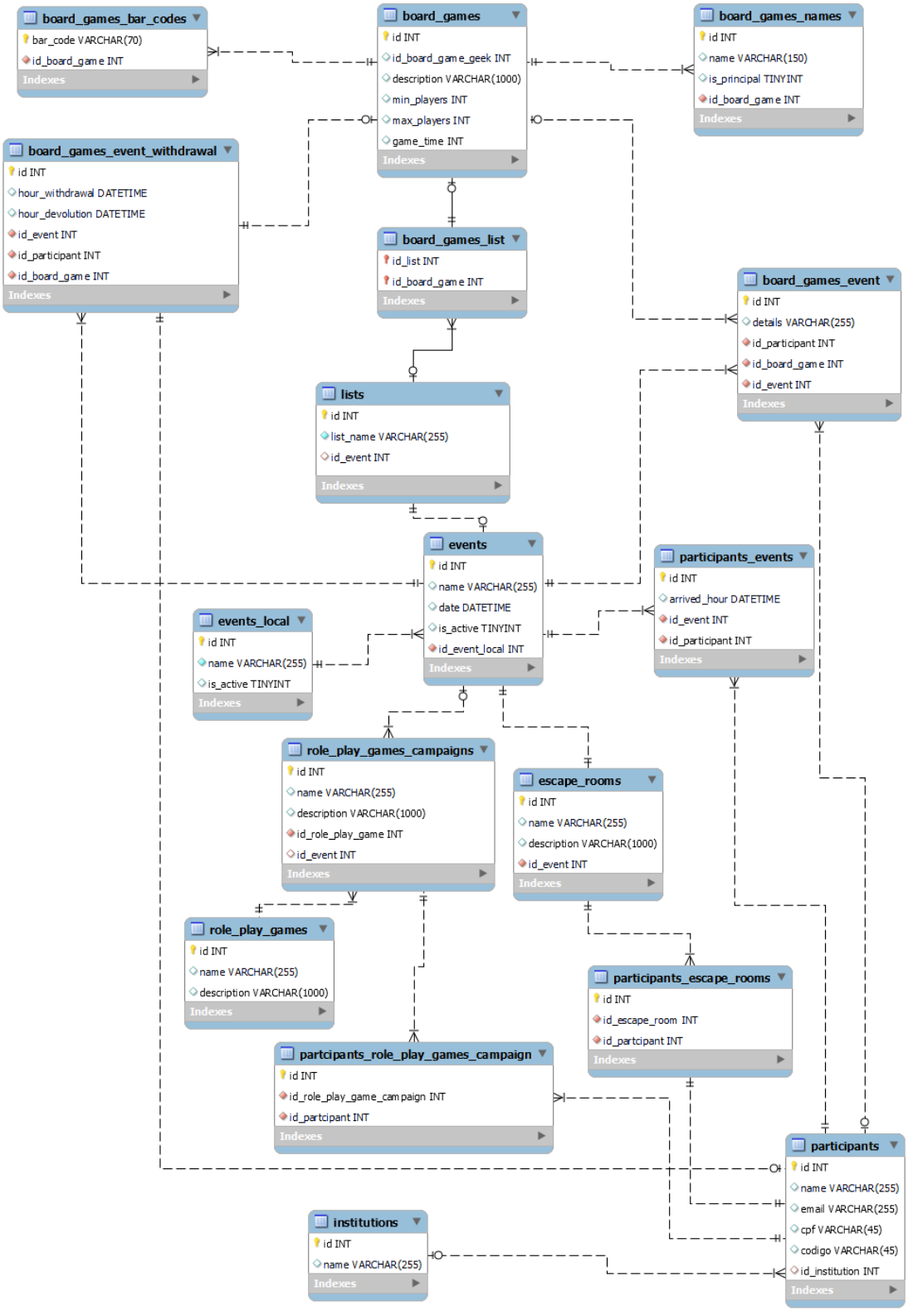


Figura 20 - Diagrama da Modelagem de Banco de Dados

Legenda	
PK	Chave Primária (Primary Key)
FK	Chave Estrangeira (Foreign Key)
NN	Não Nulo (Not Null)

Tabela: institutions			
Descrição: Essa tabela é utilizada para vincular um participante a uma instituição			
Campo	**Tipo**	**Restrição**	**Observação**
id	INT	PK	
name	TEXT		

Tabela: participants			
Descrição: Tabela utilizada para o cadastro e registro histórico de participantes dos eventos do projeto de extensão Lúdico. Ocorre a presença da chave estrangeira das instituições, não sendo obrigatória seu vínculo.			
Campo	**Tipo**	**Restrição**	**Observação**
id	INT	PK	
name	TEXT		
email	TEXT		
cpf	TEXT		
code	TEXT		
Id_institution	INT	FK	Relação com a tabela institutions

Tabela: board_games			
Descrição: Tabela utilizada para o cadastro e registro de jogos de tabuleiros de propriedade do projeto de extensão Lúdico ou para registro de informação. Essa tabela é fundamental devido a dependência de diversas outras tabelas para o funcionamento, como na realização de listas de jogos (board_games_list).			
Campo	**Tipo**	**Restrição**	**Observação**
id	INT	PK	
Id_board_game_geek	INT		Esse ID é herdado do banco de dados do sistema legado, ele é utilizado a partir do registro do site BoardGameGeek, que contém o registro de todos os jogos de tabuleiros, mas não é dependente de outra tabela.
description	TEXT		
min_players	INT		
max_players	INT		
game_time	INT		

Tabela: board_games_bar_codes			
Descrição: Tabela em que se registra o código de barra do jogo de tabuleiro (board_games), permitindo que se tenha vários jogos de tabuleiros do mesmo tipo e se diferenciando pelo cadastro do código de barra			
Campo	**Tipo**	**Restrição**	**Observação**
bar_code	TEXT	PK, NL	Código de barra do jogo, permitindo o registro e busca a partir dessa informação
Id_board_game	INT	FK, NL	Conexão com o board_game

Tabela: board_games_names			
Descrição: Tabela similar a tabela de código (board_games_bar_codes), em que um jogo pode conter diferentes nomes para o registro.			
Campo	**Tipo**	**Restrição**	**Observação**
id	INT	PK	
name	TEXT		
Is_principal	BOOL		Booleano que define se o nome registrado é o principal a ser mostrado em outras telas do sistema
Id_board_game	INT	FK, NL	Conexão com o board_game

Tabela: events_local			
Descrição: Tabela para o registro de locais que os eventos			
Campo	**Tipo**	**Restrição**	**Observação**
id	INT	PK	
name	TEXT	NL	

Tabela: events			
Descrição: Tabela para o registro dos eventos do projeto de extensão Lúdico, sendo a tabela central de todo o projeto. Outras tabelas contém uma associação com essa tabela, como participants_events, escape_rooms, lists, role_play_games_campaigns e escape_rooms.			
Campo	**Tipo**	**Restrição**	**Observação**
id	INT	PK	
name	TEXT		
date	DATETIME		
Is_active	BOOL		Booleano para definir se o evento se encontra ativo no momento.
Id_event_local	INT	FK	Realiza conexão com a tabela events_local

Tabela: board_games_lists			
Descrição: Tabela para o relacionamento muitos para muitos entre board_games e lists. Ocorre o armazenamento do ID do Jogo e do ID da Lista do Jogo, permitindo armazenar e realizar a conexão entre as 2 tabelas			
Campo	**Tipo**	**Restrição**	**Observação**
Id_list	INT	FK	ID da tabela lists
Id_board_game	INT	FK	ID da tabela board_games

Tabela: lists			
Descrição: Tabela para o registro de listas de jogos de tabuleiros, permitindo pré-definir os jogos disponíveis no evento para o controle de empréstimos.			
Campo	**Tipo**	**Restrição**	**Observação**
id	INT	PK	
List_name	TEXT	NL	Nome da lista
Id_event	INT	FK	ID da tabela events, não sendo obrigatória o seu vínculo

Tabela: lists			
Descrição: Tabela para o registro de listas de jogos de tabuleiros, permitindo pré-definir os jogos disponíveis no evento para o controle de empréstimos.			
Campo	**Tipo**	**Restrição**	**Observação**
id	INT	PK	
List_name	TEXT	NL	Nome da lista
Id_event	INT	FK	ID da tabela events, não sendo obrigatória o seu vínculo

Tabela: board_games_event_withdrawal			
Descrição: Tabela para o registro de empréstimos de jogos de tabuleiros durante a realização do evento, permitindo o gerenciamento, registro e evitar que jogos sejam esquecidos ou perdidos no evento.			
Campo	**Tipo**	**Restrição**	**Observação**
id	INT	PK	
hour_withdrawal	DATETIME		Datetime para o registro da retirada do jogo de tabuleiro durante o evento
hour_devolution	DATETIME		Datetime para o registro da devolução do jogo de tabuleiro durante o evento
Id_event	INT	FK	ID da tabela events, em que ocorre a associação do emprestimo ao evento
Id_participant	INT	FK	ID do participante da tabela participants que realizou o empréstimo do jogo
Id_board_game	INT	FK	ID do jogo de tabuleiro da tabela board_games que foi emprestado

Tabela: board_games_event			
Descrição: Tabela para o registro de jogos disponíveis para a realização do evento, permitindo o histórico de jogos que foram utilizados em outros eventos e quais jogos estão disponíveis para a realização de empréstimos do evento.			
Campo	**Tipo**	**Restrição**	**Observação**
id	INT	PK	
Id_event	INT	FK	ID da tabela events, em que ocorre a associação do jogo ao evento.
Id_participant	INT	FK	ID do participante da tabela participants que realizou o cadastro do jogo para o evento.
Id_board_game	INT	FK	ID do jogo de tabuleiro da tabela board_games para associação da disponibilidade do jogo ao evento.

Tabela: participants_events			
Descrição: Tabela para o registro de participantes que realizaram a presença no evento do projeto Lúdico, permitindo o histórico para a comprovação de presença			
Campo	**Tipo**	**Restrição**	**Observação**
id	INT	PK	
arrived_hour	DATETIME		Datetime para o registro da chegada do participante.
Id_event	INT	FK	ID da tabela events, em que ocorre a associação da presença ao evento
Id_participant	INT	FK	ID do participante da tabela participants para a associação da presença ao participante.

Tabela: escape_rooms			
Descrição: Tabela para o registro de Escape Rooms realizados pelo projeto Lúdico.			
Campo	**Tipo**	**Restrição**	**Observação**
id	INT	PK	
name	TEXT		
description	TEXT		Texto utilizado para detalhamento do Escape Room e descrição dele.
Id_event	INT	FK	ID do evento da tabela events para associar o Escape Room a um evento.

Tabela: participants_escape_rooms			
Descrição: Tabela para o registro de participantes que tiveram presença nos Escape Rooms realizados pelo projeto Lúdico.			
Campo	**Tipo**	**Restrição**	**Observação**
id	INT	PK	
Id_escape_room	INT	FK	ID do Escape Room da tabela escape_rooms para associar a presença no Escape Room
Id_participant	INT	FK	ID do participante da tabela participants que realizou a presença no Escape Room

Tabela: role_play_games			
Descrição: Tabela para o registro de Role Play Games, podendo gerenciar as campanhas vinculadas ao mesmo.			
Campo	**Tipo**	**Restrição**	**Observação**
id	INT	PK	
name	TEXT		
description	TEXT		Texto para descrição e detalhamento do RPG

Tabela: role_play_games_campaigns			
Descrição: Tabela para o registro de campanhas/sessões dos RPGs, podendo esses serem realizados fora ou durante um evento do Lúdico.			
Campo	**Tipo**	**Restrição**	**Observação**
id	INT	PK	
name	TEXT		
description	TEXT		
Id_event	INT	FK	ID do evento para relacionar com a campanha, podendo ou não ser feito durante um evento.
Id_role_game_campaign	INT	FK	ID do RPG da tabela role_play_games para associar a campanha/sessão com o RPG.

Tabela: participants_role_play_games_campaigns			
Descrição: Tabela para o registro de presença de participantes nas campanhas do RPG.			
Campo	**Tipo**	**Restrição**	**Observação**
id	INT	PK	
Id_ role_play_games_campaigns	INT	FK	ID da campanha/sessão do RPG em que o participante realizou a presença.
Id_participant	INT	FK	ID da tabela participants para registrar a presença de participantes na campanha específica.

Diagrama de Classes

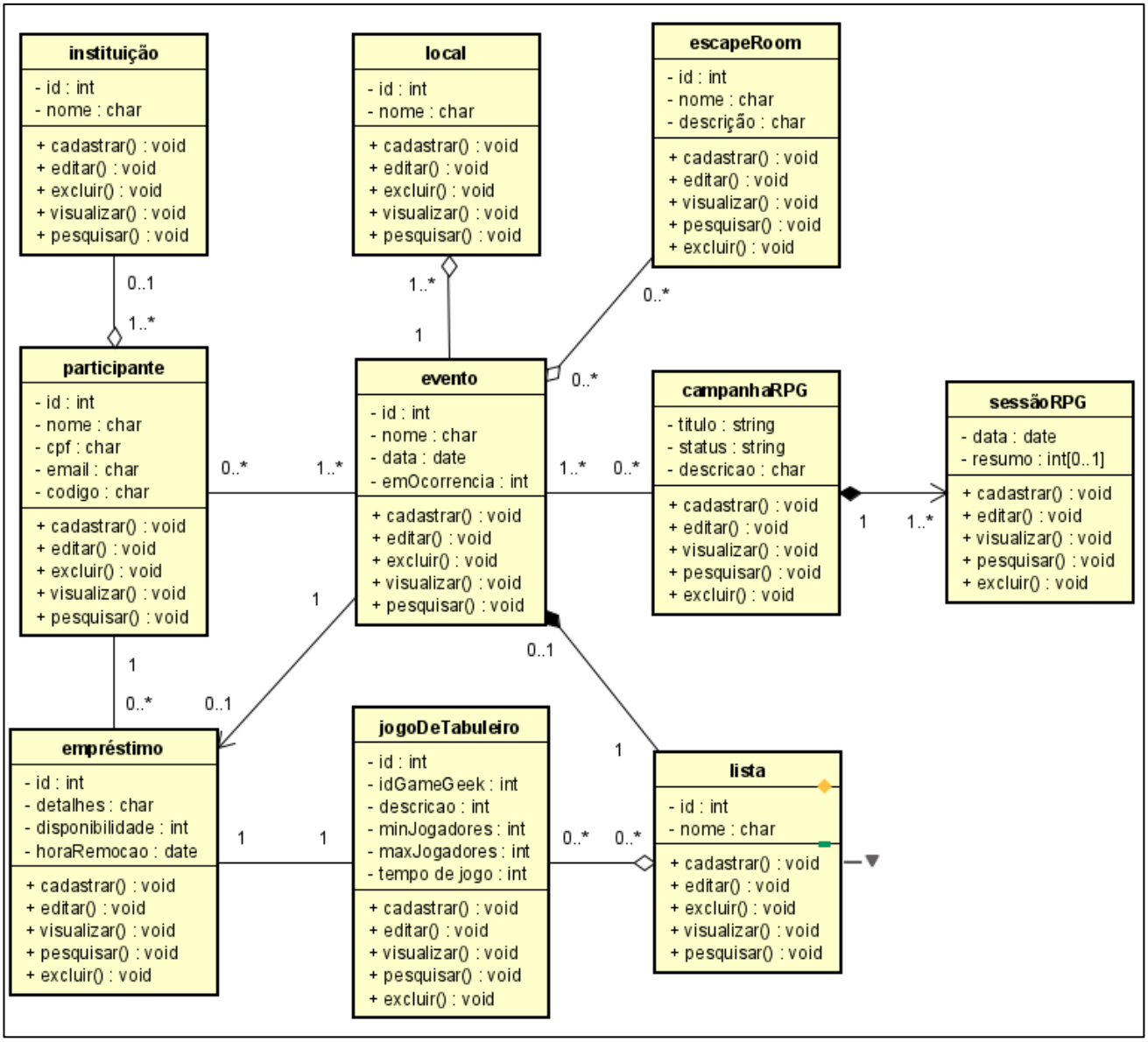


Figura 21 - Diagrama de Classe do sistema

O diagrama de classes apresenta de forma resumida a estrutura da aplicação, a qual tem como ponto central a classe evento, que há relações de associação, agregação e composição para com outras classes.

Cada classe possui atributos referentes as informações que serão guardadas ou identificação da mesma, e todas possuem os métodos de CRUD, visto que o sistema tem como objetivo armazenar os dados do projeto de extensão.

É possível dividir o sistema em 2 blocos principais, sendo a classe evento juntamente com suas dependentes, e a classe participante, que agrega instituição e possui associação com empréstimo.

Diagrama de Atividades

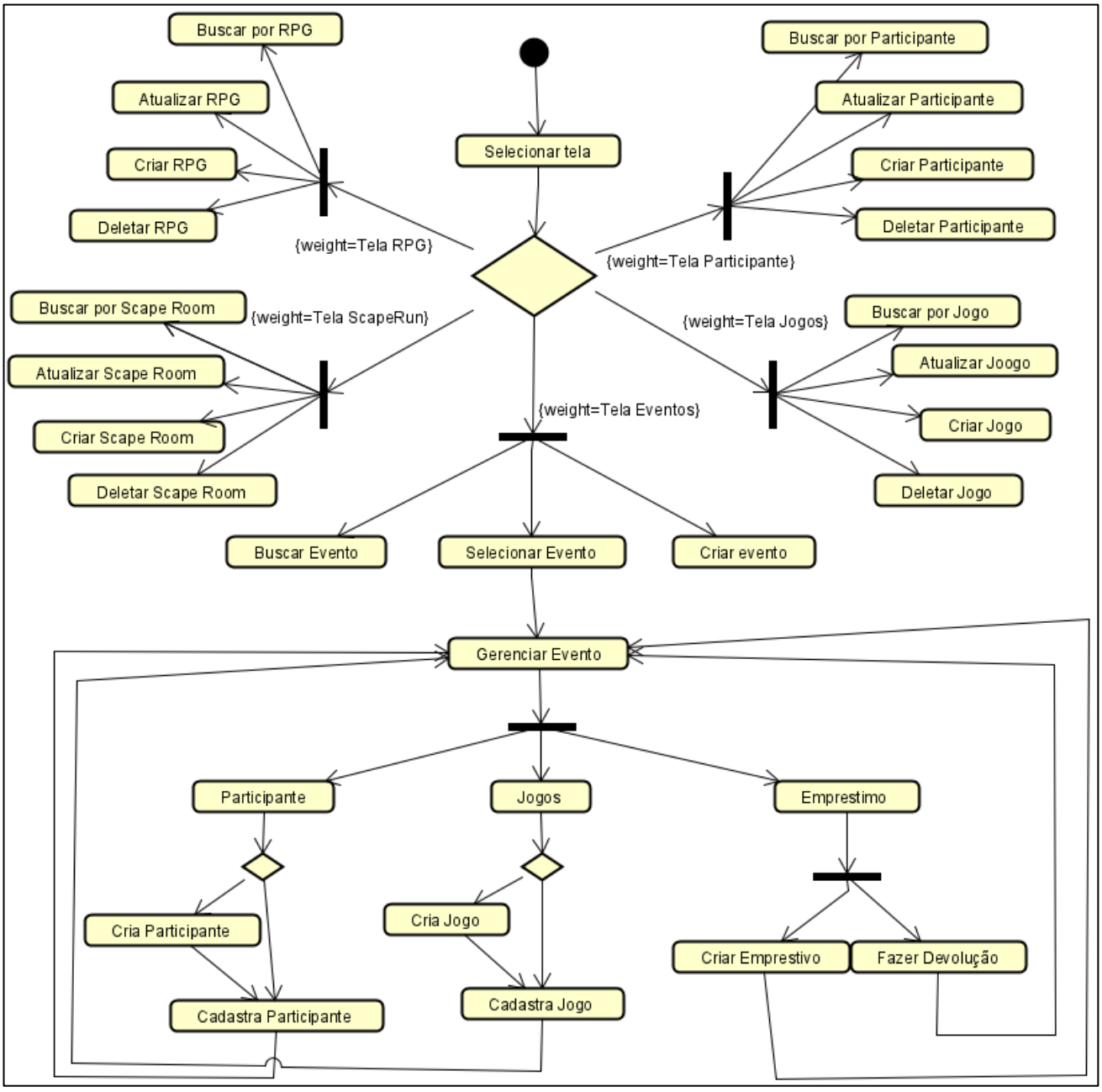


Figura 22 - Diagrama de Atividades do sistema

O diagrama de atividades representa o fluxo de navegação entre as telas e funcionalidades principais do sistema de gerenciamento de eventos com jogos. A partir da tela inicial, o usuário pode selecionar diferentes seções do sistema: RPGs, Escape Rooms, Participantes, Jogos e Eventos. Cada uma dessas seções permite ações como buscar, criar, atualizar e deletar registros.

O diagrama de atividades representa o fluxo principal de ações do sistema do projeto LÚDICO. A partir da tela inicial, o usuário escolhe entre os módulos: Eventos, Participantes, Jogos, Escape Room e RPG — cada um com as opções de buscar, criar, atualizar e deletar dados.

No módulo **Eventos**, ao selecionar um evento, o usuário acessa a funcionalidade **Gerenciar Evento**, onde é possível:

- **Associar participantes ao evento** (através das opções *Cria Participante* e *Cadastra Participante*);
- **Associar jogos ao evento** (com *Cria Jogo* e *Cadastra Jogo*);
- **Controlar os empréstimos de jogos**, incluindo a **criação de empréstimos** e o **registro de devoluções**.

Esse fluxo auxilia na implementação ao organizar as ações conforme a navegação pelas telas e módulos do sistema, facilitando o entendimento da lógica da interface e dos relacionamentos entre dados.

6 Implementação

6.1 Descrição do código

A estrutura do projeto é realizada como MVC (Model-View-Controller), do qual é um padrão de arquitetura principalmente utilizado em softwares de interface gráfica, separando os arquivos em três componentes principais, do qual seria o Model, no qual seria a representação dos dados e lógica de negócios, o View, do qual seria a interface gráfica para a apresentação ao usuário pelo uso do Windows Forms e o Controller, que se responsabiliza como um intermediário entre a comunicação do View com o Repository conectado ao banco de dados tratando as validações e os dados processados. O sistema contém também o uso de Utils, que são arquivos auxiliares para validação, conversão e operações de dados dos Models.

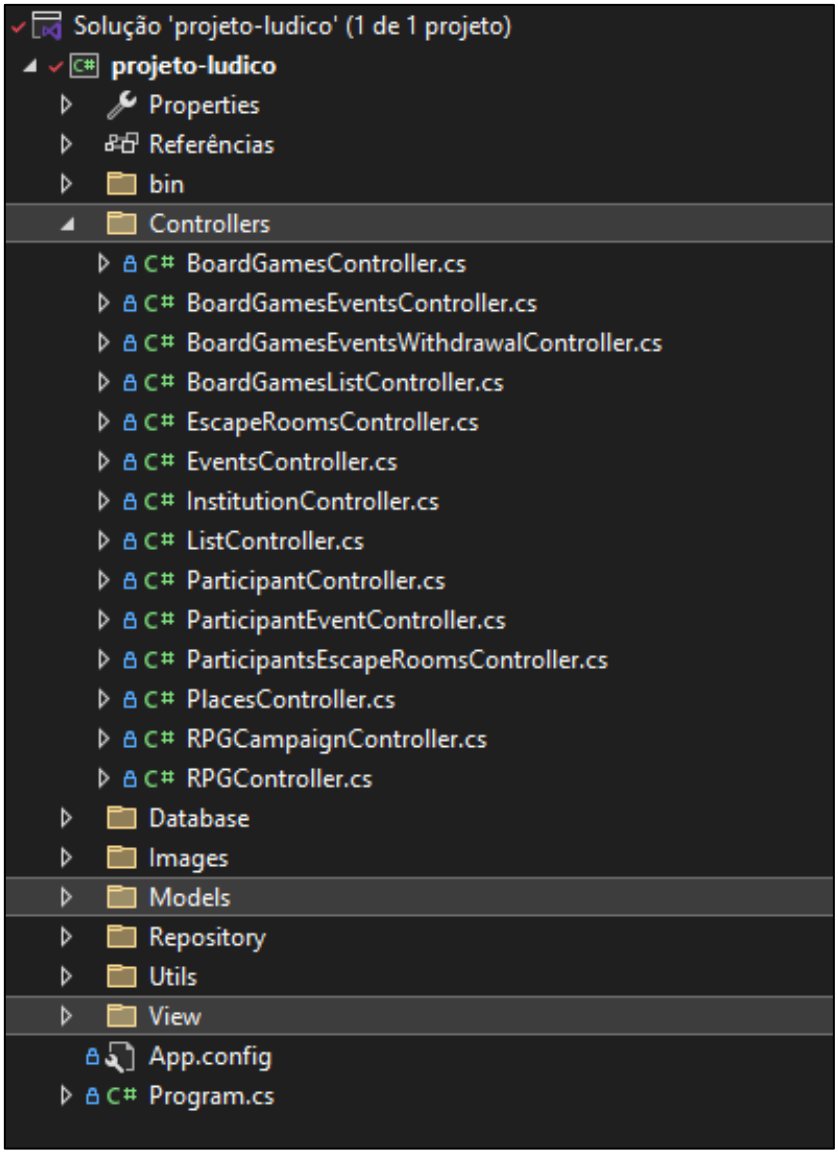


Figura 23 - Foto da organização de componentes do código

Em relação a interação do sistema com o banco de dados, utilizou-se os componentes Repository e Database (Data Access), do qual o Database retorna a conexão do arquivo do SQLite (Database.db) para o Repository realizar a transações dos dados tratados, permitindo a leitura, edição, inserção ou deleção de informações das tabelas.

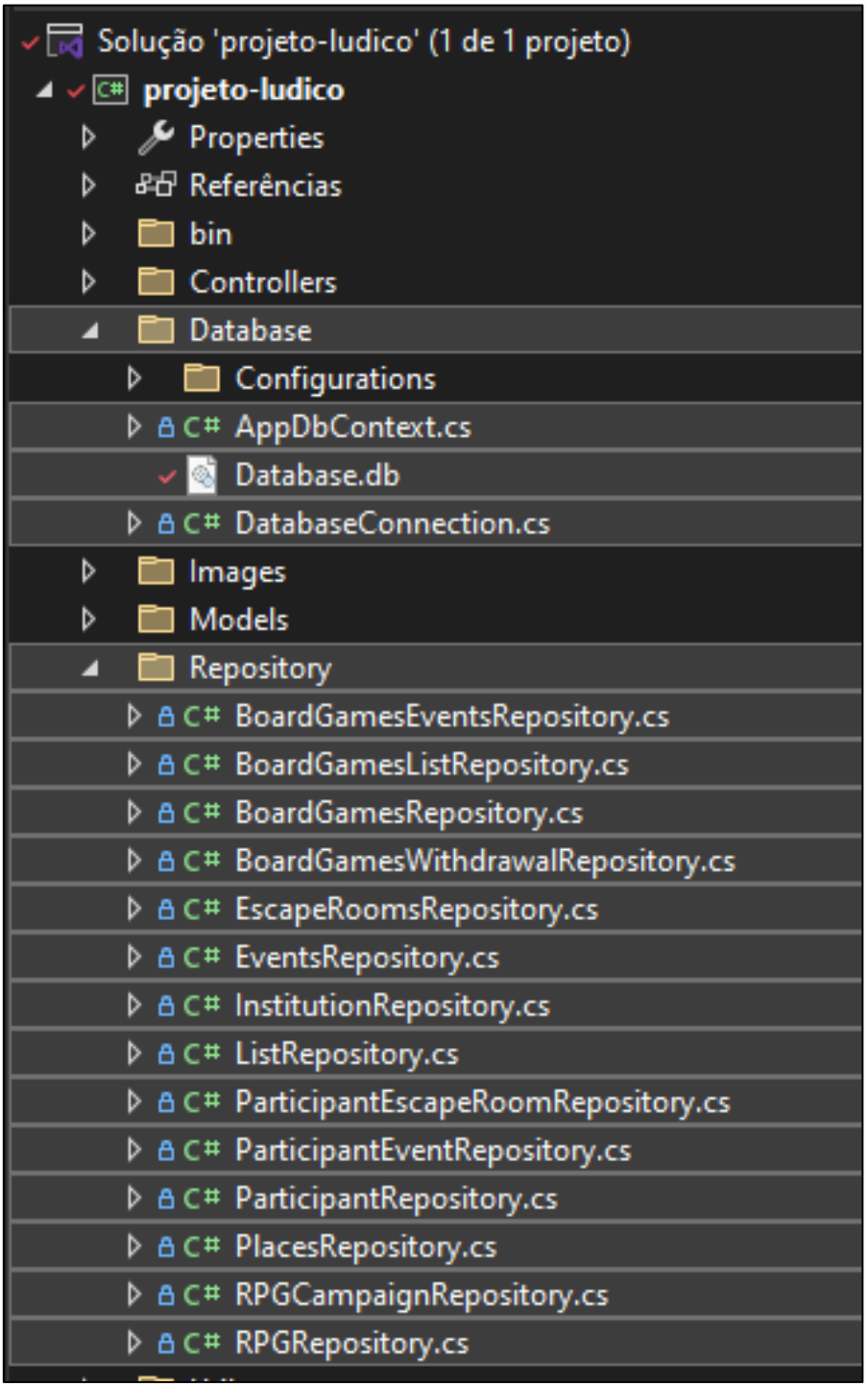


Figura 24 - Foto da organização de componentes Repository e Database

Nas transações realizadas pelos componentes Repository, duas abordagens distintas foram utilizadas para a execução de queries no banco de dados. A primeira envolveu o uso da biblioteca Entity Framework, que abstrai a estrutura de uma classe ou objeto (componentes Model) para uma tabela no

banco de dados, permitindo a execução de transações por meio do arquivo AppDbContext.cs. A segunda abordagem utilizou a biblioteca SQLConnection, na qual as queries SQL são executadas diretamente no código, utilizando o arquivo DatabaseConnection.

Para transações mais complexas, foi priorizado o uso do SQLConnection devido à sua maior rapidez, menor tempo de processamento e maior flexibilidade no tratamento de dados. Por outro lado, o Entity Framework foi utilizado para transações mais simples, já que, em operações mais complexas com múltiplas relações entre tabelas, o processo de abstração realizado pela biblioteca tende a aumentar o tempo de execução das queries.

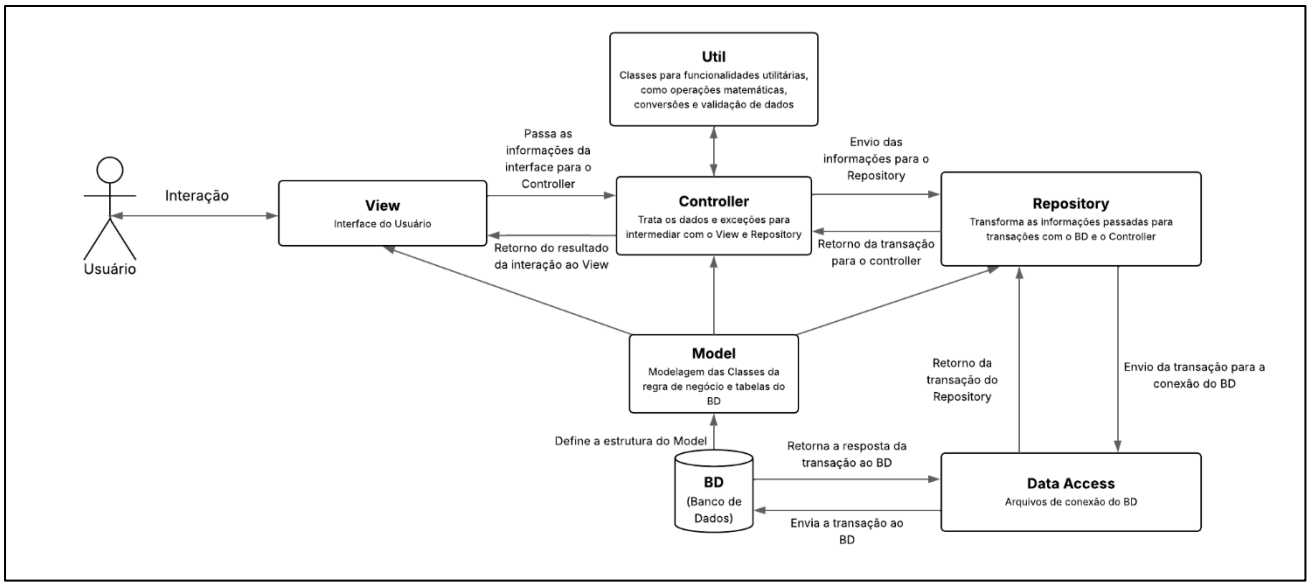


Figura 25 – Diagrama de Componentes do Sistema

Tendo como exemplo da estrutura, temos o Model do participante, que é uma classe com os atributos da tabela participants, assim definindo a estrutura dos outros componentes View e Controllers.

```
namespace projeto_ludico.Models
{
    56 referências
    public class ParticipantsModel
    {
        // Propriedades
        21 referências
        public int Id { get; set; }
        9 referências
        public int id_institution { get; set; }
        17 referências
        public string name { get; set; }
        8 referências
        public string email { get; set; }
        8 referências
        public string cpf { get; set; }
        8 referências
        public string code { get; set; }

        11 referências
        public ParticipantsModel() { }
    }
}
```

Figura 26 - Model do Participants

Com a definição do Model, desenvolveu-se o View com a interface gráfica, do qual o backend resgata as informações inseridas nos formulários para assim enviar ao componente Controller e tratar os dados para o Repository.

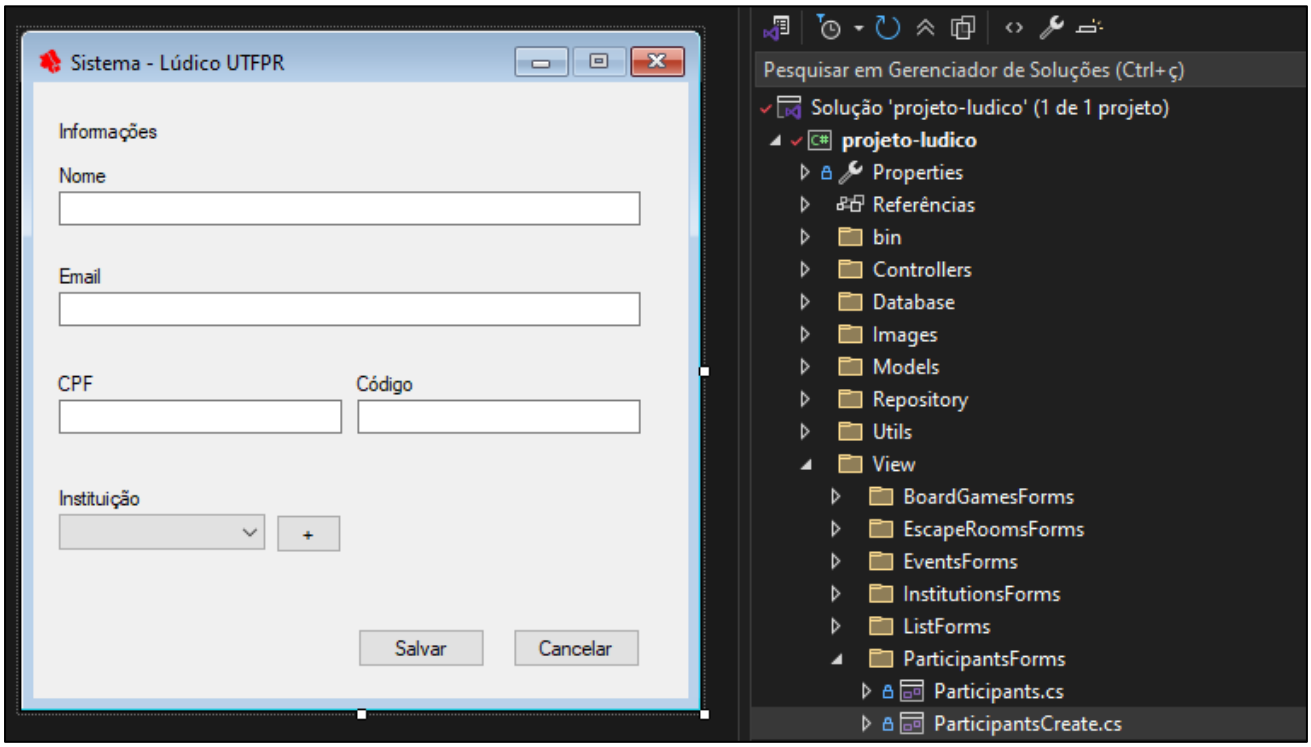


Figura 27 - Interface Gráfica do Componente View de Criação de Participantes do Sistema

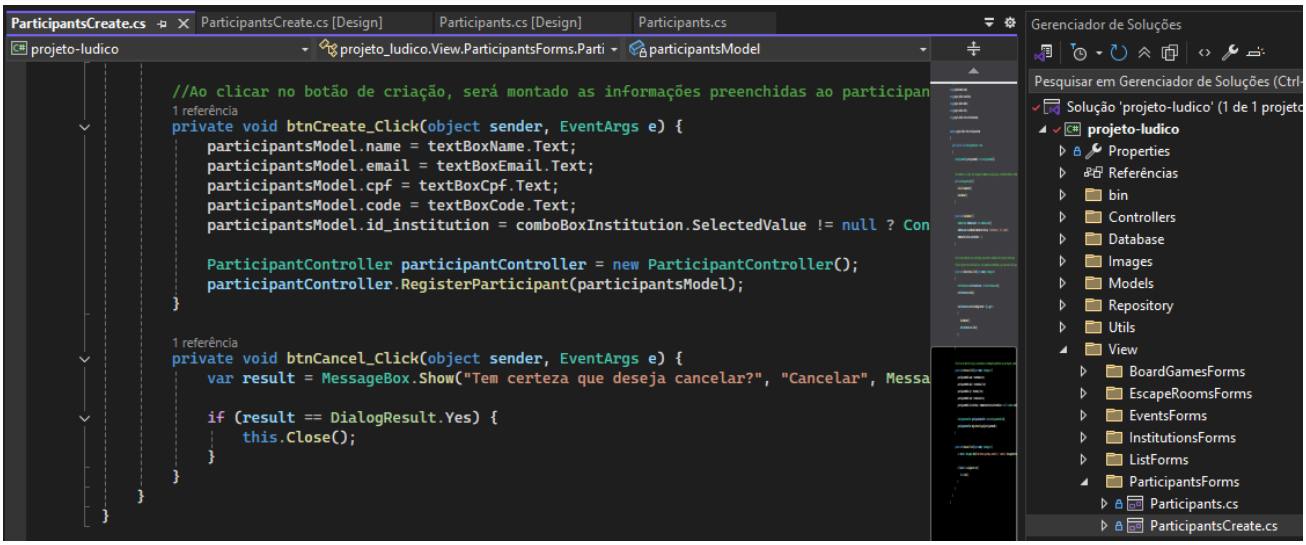


Figura 28 - Backend da Interface do Componente View de Criação de Participantes do Sistema

O Controller recebe o Model para validar os dados ao componente Repository, verificando exceções e retornar para o View as mensagens de erro caso algum erro ocorra, como falha na transação ou erro na conexão do banco.

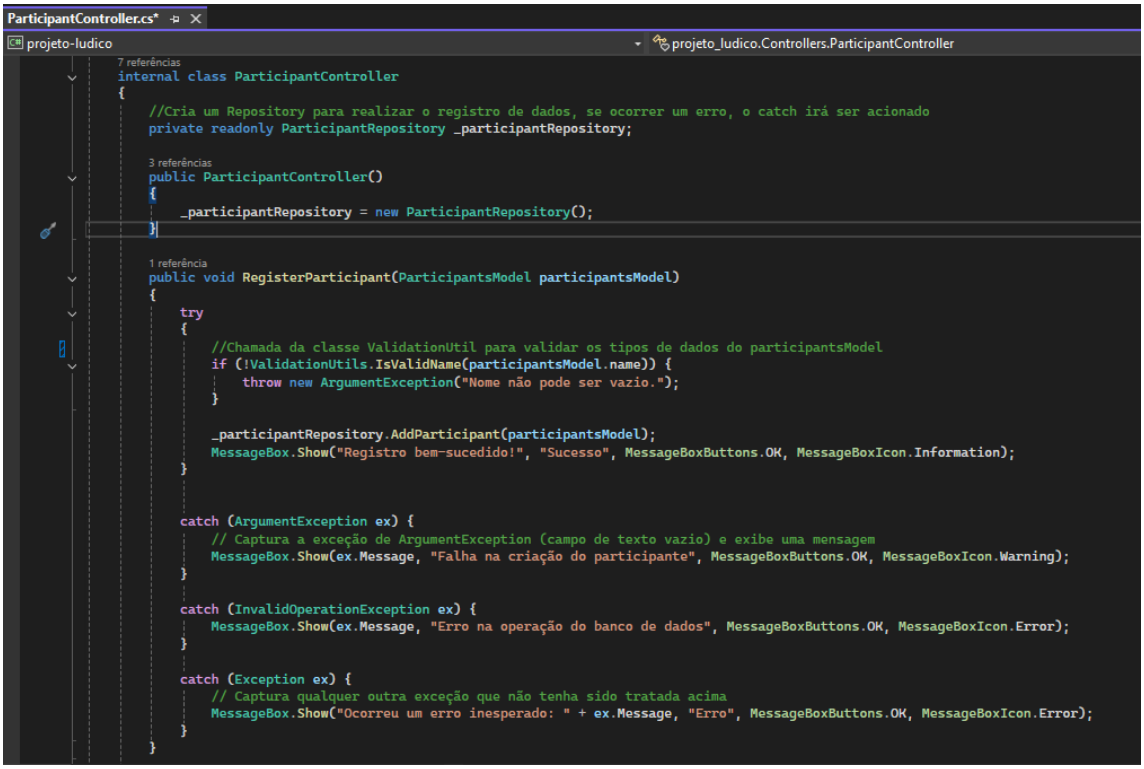


Figura 29 – Controller do Participant, que enviará as informações para o Repository

Com as informações passadas pelo Controller, o Repository usa a biblioteca SQLConnection que realiza a transação pelo uso da classe DatabaseConnection e uma query estruturada, definindo quais informações enviarem do Model do parâmetro para o arquivo de banco de dados.

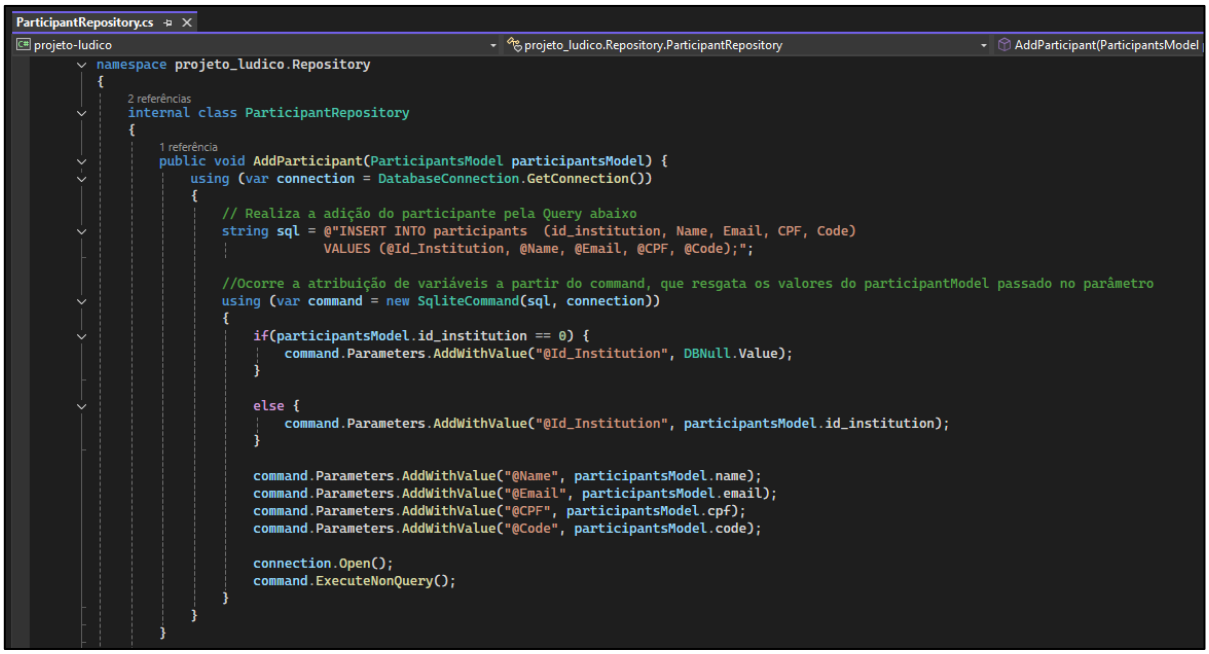


Figura 30 - Código do Repository do Participant com a função de adicionar um participante no banco

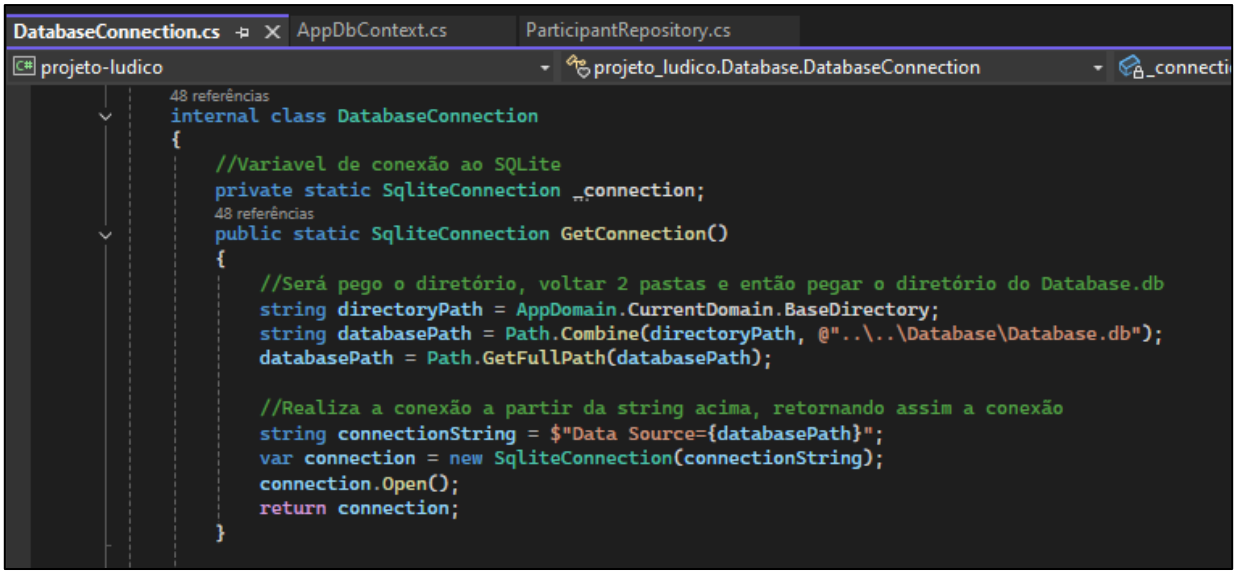


Figura 31 – Classe DatabaseConnection em que retorna a conexão do arquivo Database.db para transações

Já para uma transação usando EntityFramework, a configuração e a estrutura do código do componente Repository é mais simplificada, precisando definir a estrutura em um arquivo Configuration para o AppDbContext, do qual abstrai o Model para a tabela e realização de transações.

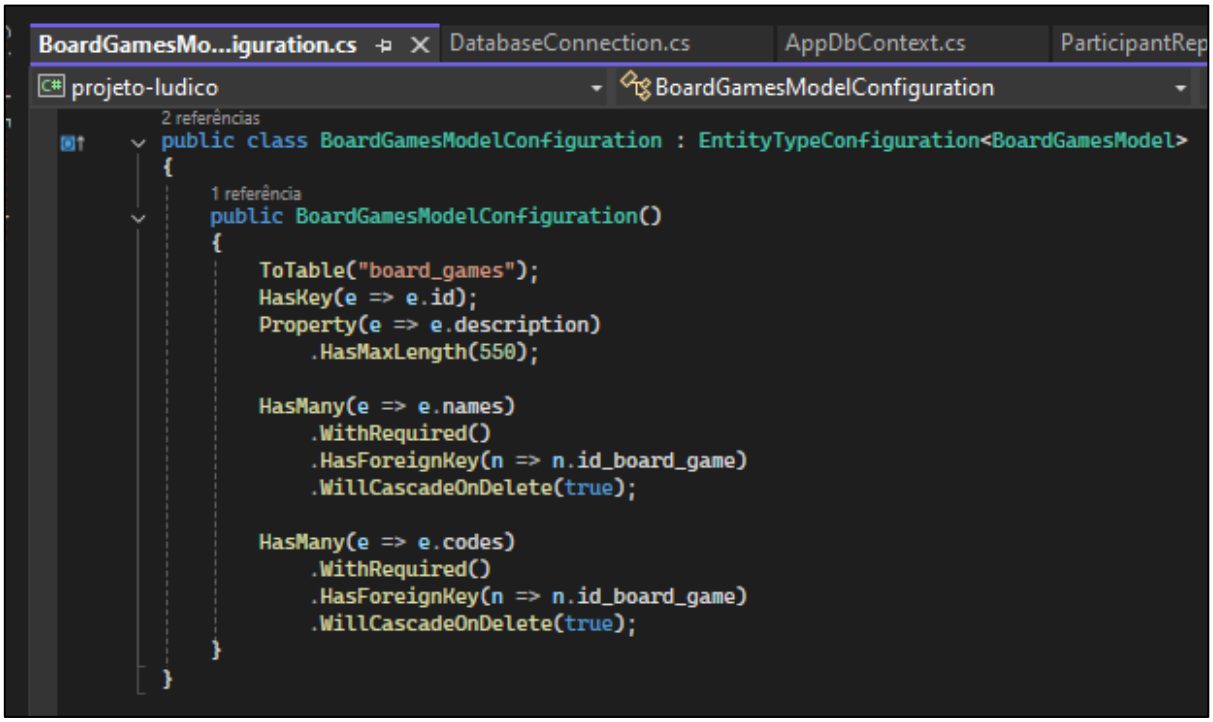


Figura 32 – Classe Configuration para o AppDbContext

AppDbContext.cs

projeto-ludico

```
1 referência
private static DbConnection CreateConnection()
{
    string connectionString = CreateConnectionString();
    var connection = new SQLiteConnection(connectionString);
    return connection;
}

1 referência
private static string CreateConnectionString()
{
    string directoryPath = AppDomain.CurrentDomain.BaseDirectory;
    string databasePath = Path.Combine(directoryPath, @"..\..\Database\Database.db");
    databasePath = Path.GetFullPath(databasePath);
    return $"Data Source={databasePath};Version=3;";
}

8 referências
public DbSet<BoardGamesModel> BoardGames { get; set; }
2 referências
public DbSet<BoardGamesNamesModel> BoardGamesNames { get; set; }
2 referências
public DbSet<BoardGamesBarCodesModel> BoardGamesBarCodes { get; set; }
0 referências
public DbSet<ListModel> List { get; set; }
5 referências
public DbSet<BoardGamesListModel> BoardGamesList { get; set; }
1 referência
public DbSet<ParticipantsModel> Participants { get; set; }
5 referências
public DbSet<ParticipantsEscapeRoomModel> ParticipantsEscapeRooms { get; set; }

0 referências
protected override void OnModelCreating(DbModelBuilder modelBuilder)
{
    modelBuilder.Configurations.Add(new BoardGamesModelConfiguration());
    modelBuilder.Configurations.Add(new BoardGamesNamesModelConfiguration());
    modelBuilder.Configurations.Add(new BoardGamesBarCodesModelConfiguration());
    modelBuilder.Configurations.Add(new ListsModelConfiguration());
    modelBuilder.Configurations.Add(new BoardGamesListModelConfiguration());
    modelBuilder.Configurations.Add(new ParticipantsModelConfiguration());
    modelBuilder.Configurations.Add(new ParticipantsEscapeRoomModelConfiguration());
}
}
```

Figura 33 – Importação dos Configurations para o arquivo AppDbContext e conectar com o Database.db

Com a definição do AppDbContext, o Repository apenas chama a transação pelo AppDbContext:

BoardGamesRepository.cs | AppDbContext.cs

projeto-ludico

```
using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;
using projeto_ludico.Models;

namespace projeto_ludico.Repository
{
    internal class BoardGamesRepository
    {
        public void AddBoardGames(BoardGamesModel boardGameModel)
        {
            using (var context = new AppDbContext())
            {
                context.BoardGames.Add(boardGameModel);
                context.SaveChanges();
            }
        }
```

Figura 34 – Repository do Board Games com o uso do EntityFramework

Como convenção no código, aplicou-se o Pascal Case para os arquivos e funções, isso é, todas as palavras começam com a inicial maiúscula na nomeação dos arquivos:

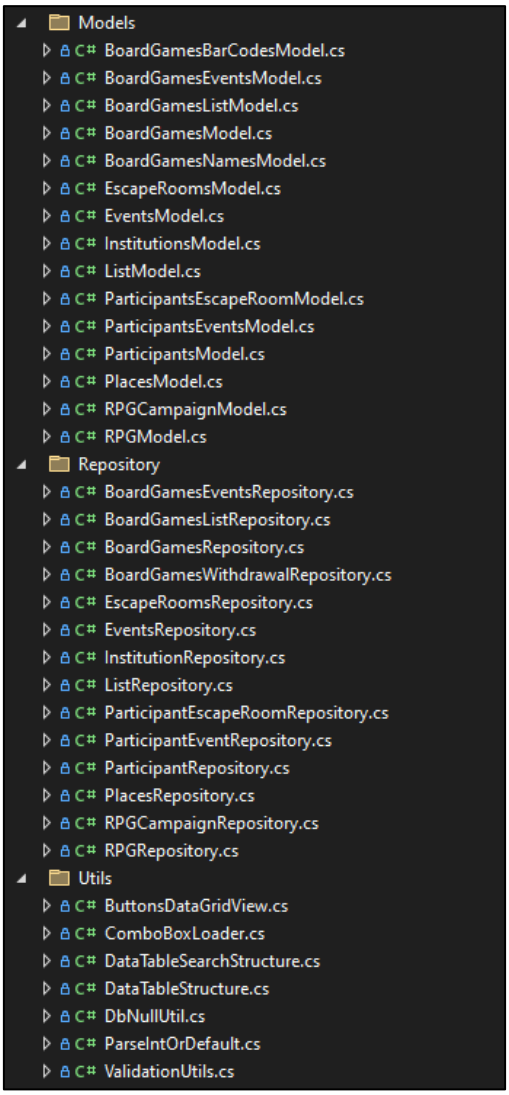


Figura 35 – Aplicação do Pascal Case nos arquivos

Para o uso de variáveis, foi aplicado o Camel Case, do qual a primeira letra da palavra inteira é minúscula, mas é usado a letra maiúscula em cada palavra composta:

```
private ParticipantsModel loadNewValues() {
    ParticipantsModel newParticipantsModel = new ParticipantsModel();
    newParticipantsModel.Id = participantsModel.Id;
    newParticipantsModel.name = textBoxName.Text;
    newParticipantsModel.email = textBoxEmail.Text;
    newParticipantsModel.cpf = textBoxCpf.Text;
    newParticipantsModel.code = textBoxCode.Text;
    newParticipantsModel.id_institution = comboBoxInstitution.SelectedValue != null ? Convert.ToInt32(comboBoxInstitution.SelectedValue) : 0;

    return newParticipantsModel;
}
```

Figura 36 – Exemplo de função com Pascal Case e variáveis Camel Case

Foi aplicado o uso de comentários em todos os tipos de componentes do sistema, no qual explica o fluxo e funcionamento das funções ou das variáveis, como exemplificado no Controller do Participant:

```
public void RegisterEscapeRooms(EscapeRoomsModel escapeRoomsModel)
{
    try
    {
        //Chamada da classe ValidationUtil para validar os tipos de dados do escapeRoomsModel
        if (!ValidationUtils.IsValidName(escapeRoomsModel.name)) {
            throw new ArgumentException("Nome não pode ser vazio.");
        }

        if (escapeRoomsModel.id_event == 0)
        {
            throw new ArgumentException("Não foi selecionado o evento do escape room.");
        }

        _escapeRoomsRepository.AddEscapeRooms(escapeRoomsModel);
        MessageBox.Show("Registro bem-sucedido!", "Sucesso", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }


    catch (ArgumentException ex)
    {
        // Captura a exceção de ArgumentException (campo de texto vazio) e exibe uma mensagem
        MessageBox.Show(ex.Message, "Falha na criação do escape room", MessageBoxButtons.OK, MessageBoxIcon.Warning);
    }

    catch (InvalidOperationException ex) {
        MessageBox.Show(ex.Message, "Erro na operação do banco de dados", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }

    catch (Exception ex) {

        // Captura qualquer outra exceção que não tenha sido tratada acima
        MessageBox.Show("Ocorreu um erro inesperado: " + ex.Message, "Erro", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

Figura 37 – Foto do código do Controller do Escape Rooms, no qual contém os comentários sobre o funcionamento da função RegisterEscapeRooms

6.2 Implantação

Para a execução do software, é necessário que tenha a biblioteca .NET Framework 4.8 instalado na máquina, exceto se o sistema operacional da máquina seja o Windows 10 ou superior por esse já conter instalado nativamente. Para o download da biblioteca, é possível acessar clicando no seguinte link: .NET Framework 4.8. Execute o instalador e clique em "Instalar" para iniciar a instalação automaticamente.

Com o .NET Framework 4.8 instalado, é necessário baixar o arquivo compactado do projeto no seguinte link: Link do Projeto. Ao concluir o download, descompacte a pasta, abra e então execute o arquivo para usar o software.

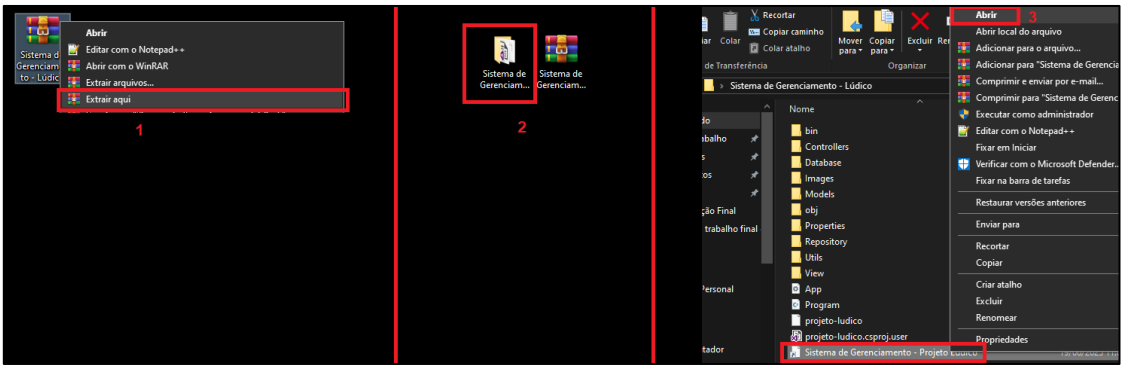


Figura 38 – Fluxo para execução do software

6.3 Telas principais

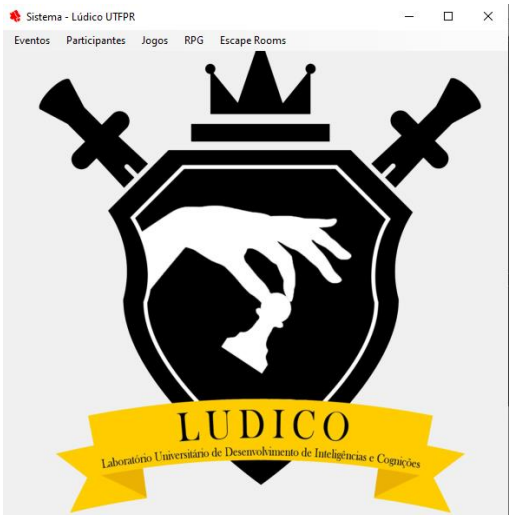


Figura 39 - Tela Inicial

Tela Inicial

Tela inicial do sistema do qual o usuário pode navegar para as outras telas a partir do menu superior.

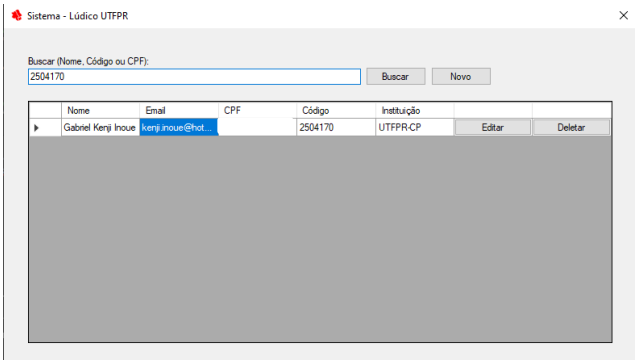


Figura 40 – Tela de Consulta de Participantes do Sistema

Tela de Consulta de Participantes

Tela de consulta dos registros feitos de participantes, podendo visualizar e pesquisar as informações cadastradas, além de permitir o acesso para cadastrar, editar ou deletar os dados pelos botões.

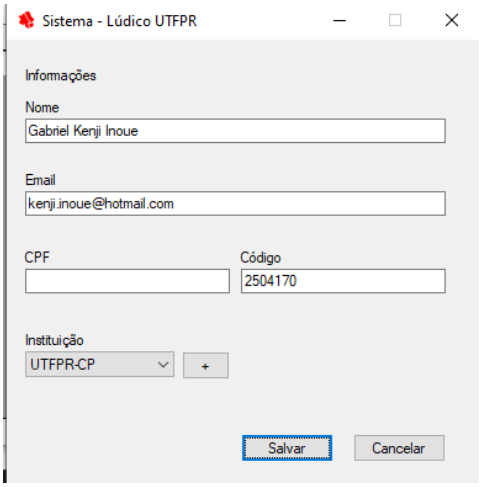


Figura 41 – Tela de Cadastro de Participante

Tela de Cadastro de Participantes

Tela de cadastro de participantes, registrando as informações para então salvar no banco de dados que será mostrado na tela de consulta.

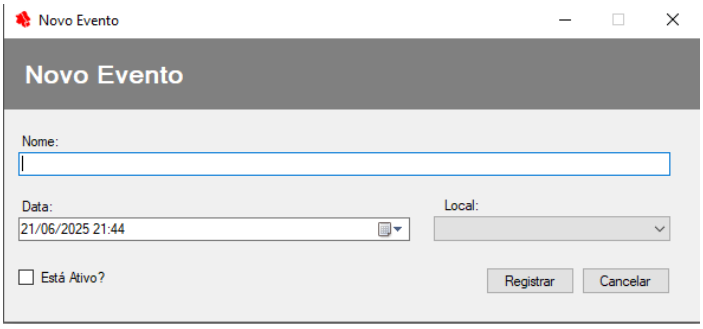


Figura 42 – Tela de Cadastro de Eventos

Tela de Cadastro de Eventos

Tela para cadastrar um evento que irá ocorrer. Com o registro, será mostrado as informações na tela de consulta de evento.

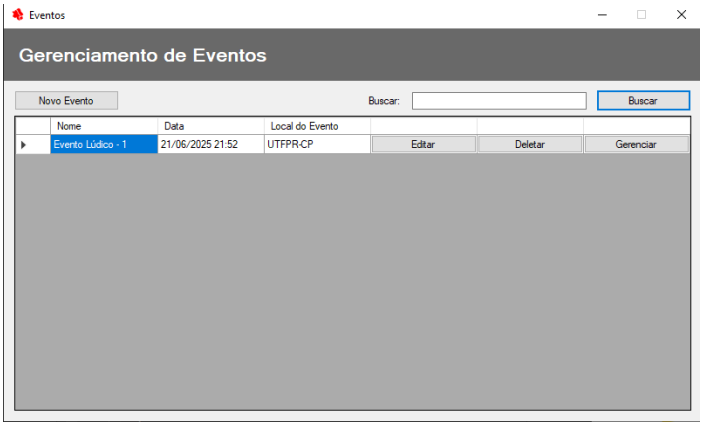


Figura 43 – Tela de Consulta de Eventos

Tela de Consulta de Eventos

Tela para consultar os eventos registrados. Ao clicar no botão “Gerenciar” da linha do evento, será aberto as telas de Gerenciamento de Evento, do qual compõe as abas de Registro de Participantes do Evento, Registro de Jogos, Empréstimo de Jogo e Lista de Empréstimos. A tela também disponibiliza o acesso de outras telas para editar, deletar e criar um evento pelos botões.

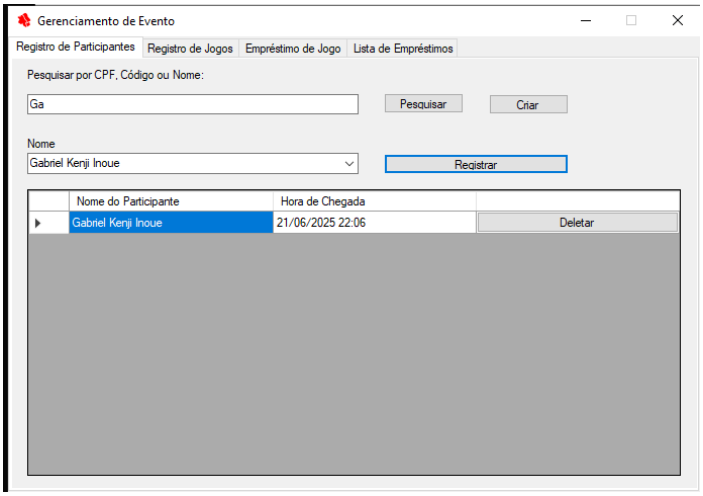


Figura 44 – Tela de Registro de Participantes do Evento

Tela de Registro de Participantes do Evento

Tela para cadastrar a presença de um participante no evento do Lúdico, podendo pesquisar por CPF, Código ou Nome do participante e então o resultado aparecerá na caixa de seleção, precisando apertar o botão “Registrar” ao lado para confirmar a presença do participante, que aparecerá na tabela.

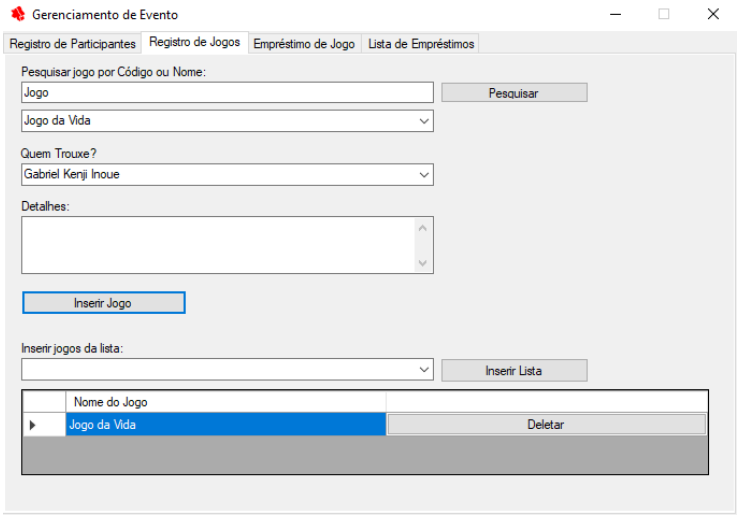


Figura 45 – Tela de Registro de Jogos do Evento

Tela de Registro de Jogos do Evento

Tela para cadastrar os jogos disponíveis do evento para realização de empréstimo aos participantes, sendo necessário inserir o participante bolsista do Lúdico responsável pelo jogo. Uma outra opção disponibilizada é o cadastro dos jogos pela lista, em que de vez o usuário cadastrar os jogos um de cada vez, ele importa todos os jogos de uma lista criada previamente.

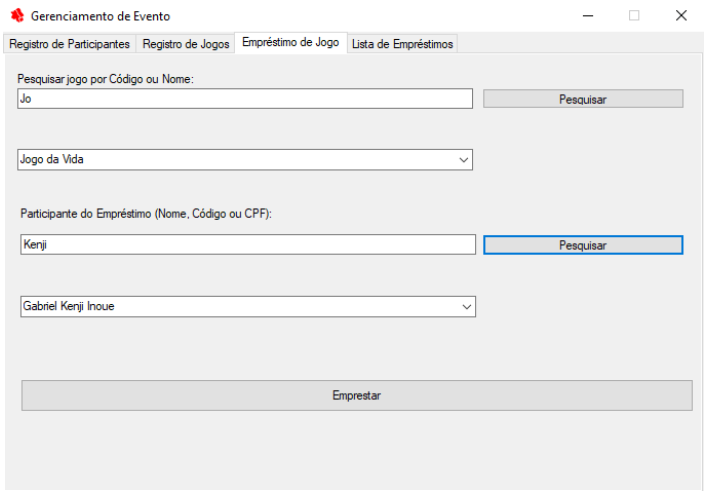


Figura 46 – Tela de Empréstimo de Jogo

Tela de Empréstimo de Jogo

Tela destinada ao registro de empréstimos dos jogos cadastrados na aba Registro de Jogos, permitindo que os participantes utilizem os jogos durante o evento. Para efetuar o registro, é necessário preencher os formulários com o jogo selecionado e o nome do participante que receberá o empréstimo.

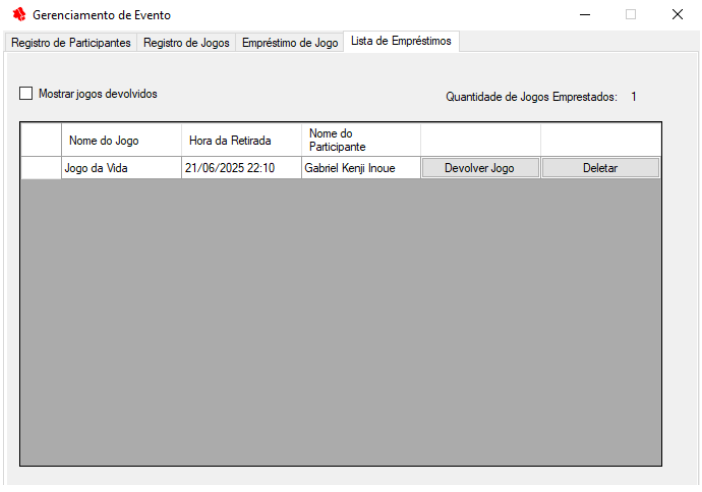


Figura 47 – Tela de Lista de Empréstimos

Tela de Lista de Empréstimos

Tela para visualização e controle dos jogos emprestados para os participantes, podendo visualizar o horário de retirada e devolver o jogo.

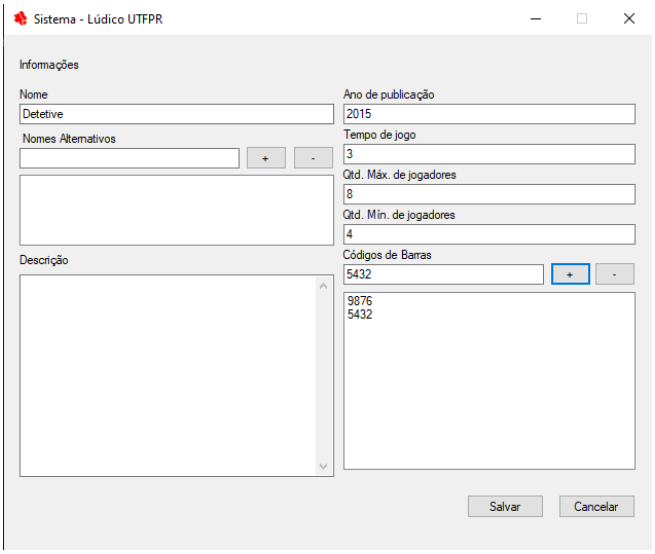


Figura 48 – Tela de Cadastro de Jogos

Tela de Cadastro de Jogos

Tela de cadastro de jogos, registrando as informações para então salvar no banco de dados que será mostrado na tela de consulta.

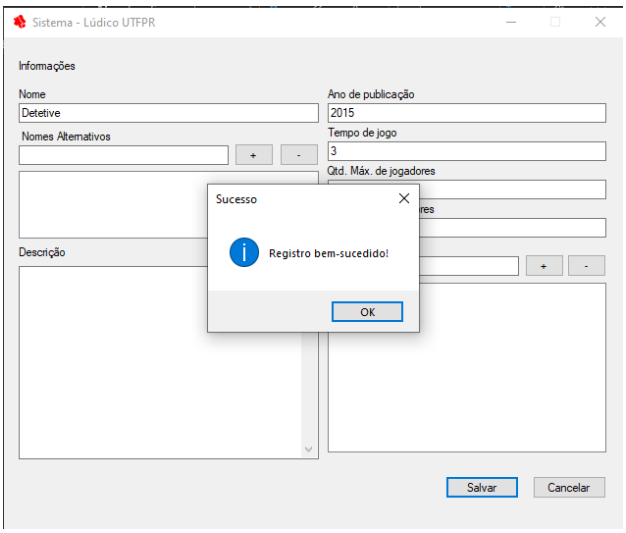


Figura 49 – Tela de Pop-Up de Funcionamento do Cadastro

Pop-Up de Funcionamento do Cadastro

Pop-up do funcionamento do registro do cadastro de jogos, informando que a transação funcionou ao usuário depois dele preencher e clicar no botão Salvar da tela.

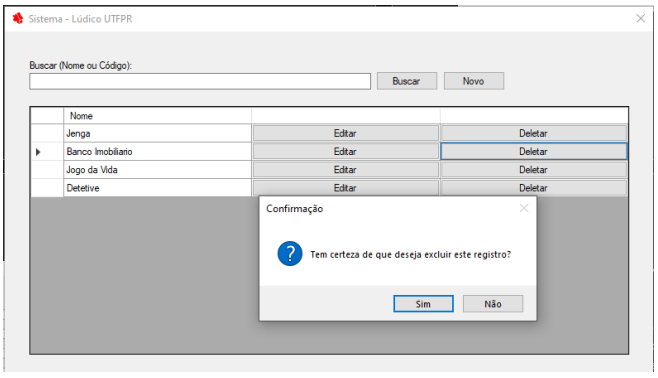


Figura 50 – Tela de Pop-Up de Confirmação de Deleção

Pop-Up de Confirmação de Deleção

Pop-up para confirmar se o usuário quer realmente deletar ao clicar no botão Deletar da linha do item selecionado.

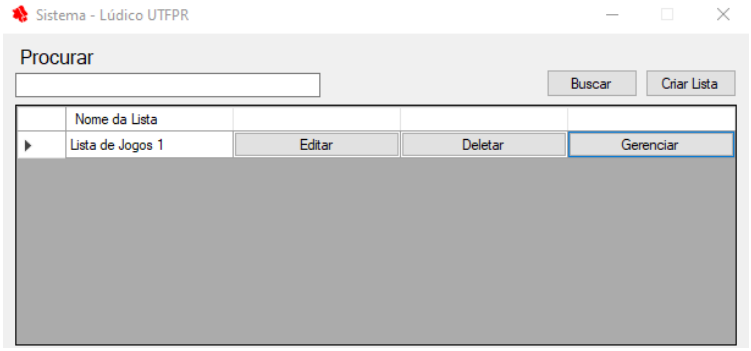


Figura 51 – Tela de Consulta de Listas de Jogos

Tela de Consulta de Listas de Jogos

Tela para controlar as listas de jogos para usar nos eventos. Ao clicar no botão “Gerenciar” da linha, será aberto a tela de Jogos da Lista. A tela também disponibiliza o acesso de outras telas para editar, deletar e criar uma lista pelos botões.

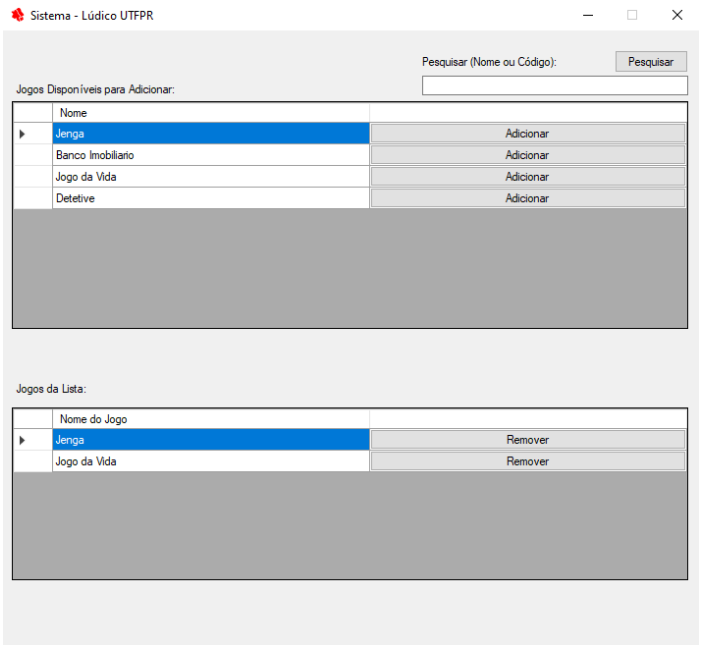


Figura 52 – Tela de Jogos da Lista

Tela de Jogos da Lista

Tela para controlar quais jogos adicionar lista, sendo a primeira tabela todos os jogos cadastrados no sistema e a tela inferior os jogos adicionados na lista.

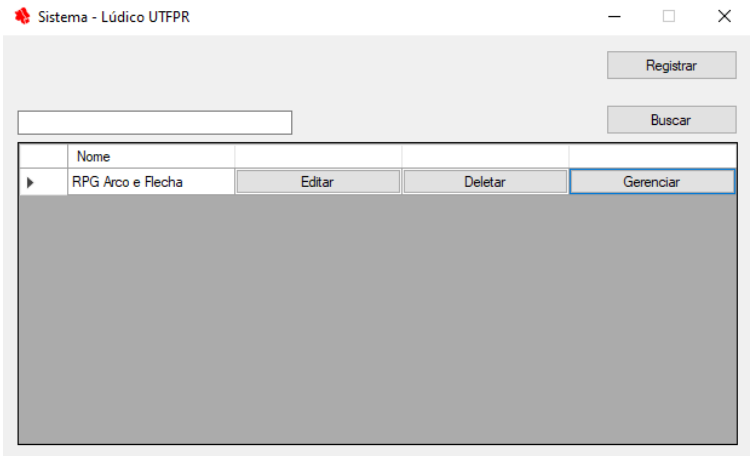


Figura 53 – Tela de Consulta de RPGs

Tela de Consulta de RPGs

Tela para controlar os RPGs realizados pelo Lúdico. Ao clicar no botão “Gerenciar” da linha, será aberto a tela de campanhas/sessões do RPG. A tela também disponibiliza o acesso de outras telas para editar, deletar e criar um RPG pelos botões.

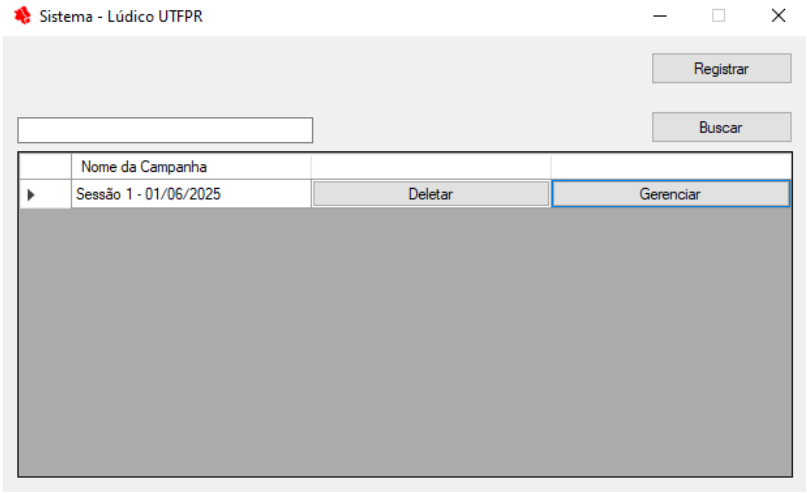


Figura 54 – Tela de Consulta de Campanhas/Sessões de RPG

Tela de Consulta de Campanhas/Sessões de RPG

Tela para controlar as campanhas/sessões dos RPGs realizados pelo Lúdico. Ao clicar no botão “Gerenciar” da linha, será aberto a tela de participantes da sessão. A tela também disponibiliza o acesso de outras telas para deletar e criar uma sessão pelos botões.

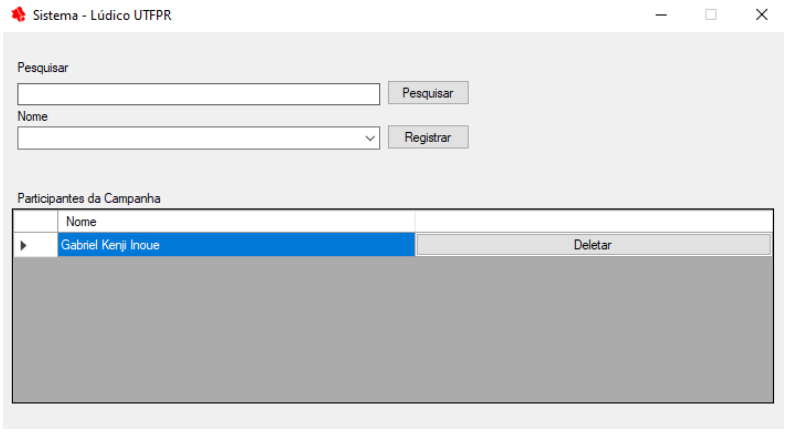


Figura 55 – Tela de Consulta de Participantes das Campanhas/Sessões de RPG

Tela de Consulta de Participantes das Campanhas/Sessões de RPG

Tela para controlar os participantes das campanhas/sessões dos RPGs realizados pelo Lúdico. O usuário pode pesquisar pelo nome no qual aparecerá na caixa de seleção e então clicar em “Registrar” para adicionar na campanha.

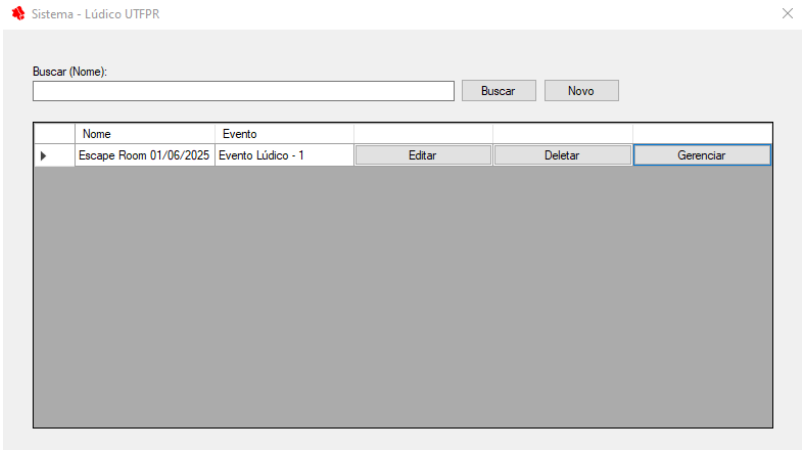


Figura 56 – Tela de Consulta de Escape Rooms

Tela de Consulta de Escape Rooms

Tela para controlar os Escape Rooms realizados pelo Lúdico. Ao clicar no botão “Gerenciar” da linha, será aberto a tela de Gerenciamento de Participantes do Escape Room. A tela também disponibiliza o acesso de outras telas para editar, deletar e criar um Escape Room pelos botões.

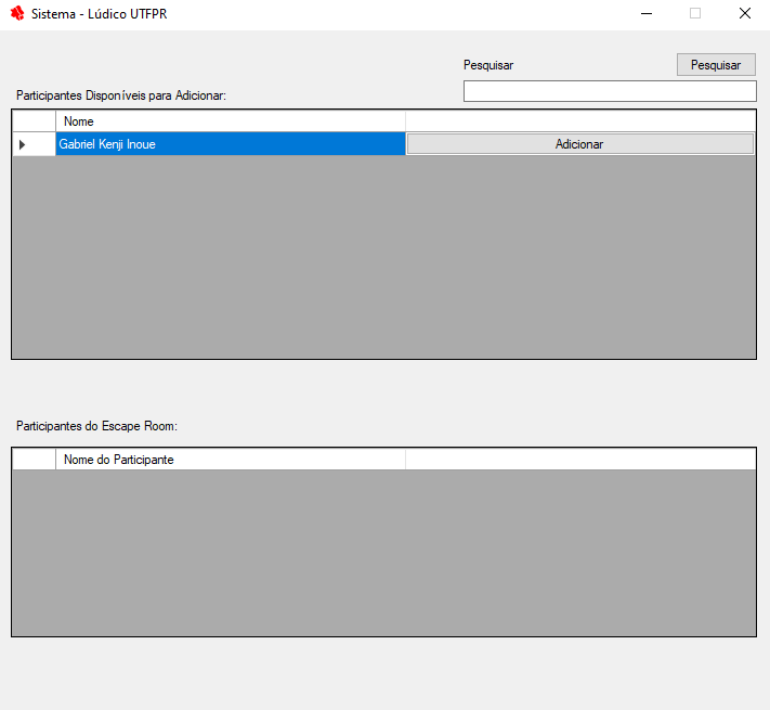


Figura 57 – Tela de Gerenciamento de Participantes do Escape Room

Tela de Gerenciamento de Participantes do Escape Room

Tela para controlar os participantes do Escape Rooms realizados pelo Lúdico. Sendo a primeira tabela todos os participantes do evento vinculado do evento do Escape Room enquanto a tela inferior mostra os participantes do Escape Room.

7 Considerações Finais

O desenvolvimento do sistema representou um avanço significativo em relação à ferramenta anteriormente utilizada pelo projeto de extensão Lúdico. A proposta de criar uma solução mais intuitiva, funcional e alinhada às necessidades reais da equipe foi, em grande parte, alcançada. A nova aplicação oferece uma forma mais prática de registrar eventos, participantes, jogos de mesa, sessões de escape room e campanhas de RPG, promovendo maior organização e facilidade no uso diário.

Naturalmente, durante o processo surgiram desafios, especialmente relacionados ao tempo disponível e à complexidade de certos recursos. Algumas funcionalidades consideradas complementares como o suporte a banco de dados remoto, foram deixadas para versões futuras. Ainda assim, o projeto foi pensado para ser flexível, permitindo que novas melhorias sejam incorporadas com relativa facilidade.

A aplicabilidade do sistema é concreta: ele atende às demandas atuais do Lúdico e tem potencial para ser adaptado a outros contextos semelhantes. Acreditamos que sua adoção contribuirá para a rotina dos monitores e para a qualidade geral da organização dos eventos.

Por fim, o sistema deixa portas abertas para continuidade. A base construída pode servir como ponto de partida para outros estudantes ou membros do projeto que queiram evoluir a aplicação, seja aprimorando recursos existentes, integrando com outras plataformas ou expandindo sua atuação para além do ambiente atual.

8 Bibliografia

JIANG, Jiachen. Como faço para usar C# e .NET com SQLite?. *In*: JIANG, Jiachen. **Como faço para usar C# e .NET com SQLite?**. [*S. l.*], 2017?. Disponível em: https://learn.microsoft.com/pt-br/shows/on-dotnet/how-do-i-use-csharp-and-dotnet-with-sqlite. Acesso em: 25 maio 2025.

CARLOS MACORATTI, José. C# - Usando o banco de dados SQLite - I. *In*: **C# - Usando o banco de dados SQLite - I**. [*S. l.*], 2017?. Disponível em: https://www.macoratti.net/17/04/cshp_sqlite1.htm. Acesso em: 24 maio 2025.

GUIA de Início Rápido: **Usar o .NET (C#) para consultar um banco de dados.** [*S. l.*], 26 set. 2024. Disponível em: https://learn.microsoft.com/pt-br/azure/azure-sql/database/connect-query-dotnet-core?view=azuresql. Acesso em: 24 maio 2025.

MICROSOFT. Guia de Introdução ao Visual Studio IDE. *In*: **Guia de Introdução ao Visual Studio IDE**. [*S. l.*], 2022?. Disponível em: https://visualstudio.microsoft.com/pt-br/vs/getting-started/. Acesso em: 17 maio 2025.

DEVMEDIA. Padrão MVC - Java Magazine. *In*: **Padrão MVC - Java Magazine**. [*S. l.*], 2011. Disponível em: https://www.devmedia.com.br/padrao-mvc-java-magazine/21995. Acesso em: 14 jun. 2025.

MICROSOFT LEARN. DataGridView Classe. *In*: **DataGridView Classe**. [*S. l.*], 2015?. Disponível em: https://learn.microsoft.com/pt-br/dotnet/api/system.windows.forms.datagridview?view=windowsdesktop-8.0. Acesso em: 31 maio 2025.

MICROSOFT LEARN. Tutorial: Criar um aplicativo do Windows Forms no Visual Studio com C#. *In*: **Tutorial: Criar um aplicativo do Windows Forms no Visual Studio com C#**. [*S. l.*], 12 jun. 2025. Disponível em: https://learn.microsoft.com/pt-br/visualstudio/ide/create-csharp-winform-visual-studio?view=vs-2022. Acesso em: 3 maio 2025.