



**CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA
SOUZA**

**FACULDADE DE TECNOLOGIA DE LINS PROF. ANTONIO SEABRA
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE
SISTEMAS**

MATHEUS PITAS BAPTISTA

**SYNCFRIENDS
GERENCIADOR DE HORÁRIOS PARA JOGOS MULTIJOADORES
ENTRE AMIGOS**

Escaneie a imagem para verificar a autenticidade do documento
Hash SHA256 do PDF original #8c5cace987b42fa5edc228c3f6562e4a97c5dd9c9b2244f9ed2c3dacfc13c8f4
<https://valida.ae/0df721762f3507afb0b3de824f3da73c4eeb34f2a7c3c7eba>

**LINS/SP
2º SEMESTRE/2023**





CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA
FACULDADE DE TECNOLOGIA DE LINS PROF. ANTONIO SEABRA
CURSO SUPERIOR DE TECNOLOGIA EM 2023

MATHEUS PITAS BAPTISTA

SYNCFRIENDS
GERENCIADOR DE HORÁRIOS PARA JOGOS MULTIJOGADORES
ENTRE AMIGOS

Trabalho de Conclusão de Curso (TCC) apresentado à
Faculdade de Tecnologia de Lins para a obtenção do
título de Tecnólogo (a) em Análise e Desenvolvimento
de Sistemas.

Orientador: Prof. Me. Thiago Seti Patricio

Escaneie a imagem para verificar a autenticidade do documento
Hash SHA256 do PDF original #8c5cace987b42fa5edc228c3f6562e4a97c5dd9c9b2244f9ed2c3dacf13c8f4
<https://valida.ae/0df721762f3507afb0b3de824f3da73c4eeb34f2a7c3c7eba>





Escaneie a imagem para verificar a autenticidade do documento
Hash SHA256 do PDF original #8c5cace987b42fa5edc228c3f6562e4a97c5dd9c9b2244f9ed2c3dacfc13c8f4
<https://valida.ae/0df721762f3507afb0b3de824f3da73c4eeb34f2a7c3c7eba>

Baptista, Matheus Pitas

B222s Syncfriends: gerenciador de horários para jogos multijogadores entre amigos / Matheus Pitas Baptista. — Lins, 2023.

48f.

Trabalho de Conclusão de Curso (Tecnologia em Análise e Desenvolvimento de Sistemas) — Faculdade de Tecnologia de Lins Professor Antonio Seabra: Lins, 2023.

Orientador(a): Me. Thiago Seti Patricio

1. Agendamento. 2. NodeJS. 3. ReactJS. 4. NodeMailer. 5. Jogos. I. Patricio, Thiago Seti. II. Faculdade de Tecnologia de Lins Professor Antonio Seabra. III. Título.

CDD 004.21





Escaneie a imagem para verificar a autenticidade do documento
Hash SHA256 do PDF original #8c5cace987b42fa5edc228c3f6562e4a97c5dd9c9b2244f9ed2c3dacfc13c8f4
<https://valida.ae/0df721762f3507afb0b3de824f3da73c4eeb34f2a7c3c7eba>

MATHEUS PITAS BAPTISTA

**SYNCFRIENDS:
GERENCIADOR DE HORÁRIOS PARA JOGOS MULTIJOGADORES ENTRE
AMIGOS**

Trabalho de Conclusão de Curso apresentado à
Faculdade de Tecnologia de Lins como parte dos
requisitos para obtenção do título de Tecnólogo em
Análise e Desenvolvimento de Sistemas sob
orientação do prof. Me. Thiago Seti Patricio.

Data de aprovação: ____/____/____



Prof. Me. Thiago Seti Patricio



Prof. Me. Júlio Fernando Lieira



Prof. Me. Luiz Fernando de Oliveira Silva





Escaneie a imagem para verificar a autenticidade do documento
Hash SHA256 do PDF original #8c5cace987b42fa5edc228c3f6562e4a97c5dd9c9b2244f9ed2c3dacfc13c8f4
<https://valida.ae/0df721762f3507afb0b3de824f3da73c4eeb34f2a7c3c7eba>



Dedico a todos que foram fundamentais na
conclusão deste projeto, tanto tecnicamente quanto
emocionalmente.

Matheus Pitas Baptista



AGRADECIMENTOS

Gostaria de agradecer primeiramente à Deus, pela saúde que tenho e com isso conseguir me capacitar cada vez mais e atingir meus objetivos, à minha mãe Célia Pereira Pitas, e aos meus irmãos Amilton Baptista Neto e Lucas Pitas Baptista, que sempre me auxilia em diversos aspectos da vida, e agradecer também ao professor Thiago Seti Patricio, que orientou esse Trabalho e deu suporte a todas as dúvidas presentes.

Matheus Pitas Baptista





RESUMO

O sistema de gerenciamento de horários para jogos multijogadores entre amigos, "SyncFriends", tem como objetivo proporcionar uma melhor organização para as atividades de lazer agendadas por um grupo de amigos. Com o uso de tecnologias como: HTML, CSS, JavaScript, ReactJS e NodeJS, pode-se criar uma aplicação que notifica um agendamento marcado para o usuário através de um Email, evitando assim, o esquecimento do compromisso ou o preenchimento desse horário com outras atividades. Para que isso aconteça, o uso da biblioteca "Nodemailer" como responsável pelo envio de Email e a biblioteca "Axios" para consulta no banco de dados através de uma API interna serão indispensáveis.

Palavras-chave: Agendamento. NodeJS. ReactJS. Nodemailer. Jogos.





ABSTRACT

The "SyncFriends" multiplayer game scheduling system aims to enhance the organization of leisure activities planned by a group of friends. By utilizing technologies such as HTML, CSS, JavaScript, ReactJS and NodeJS, it is possible to create an application that notifies users of scheduled appointments via email, preventing forgetfulness or scheduling conflicts with other activities. To achieve this, the use of the "Nodemailer" library for email sending and the "Axios" library for database queries through an internal API will be essential.

Keywords: Scheduling. NodeJS. ReactJS. NodeMailer. Games.





LISTA DE ILUSTRAÇÕES

Figura 2.1 – Google Agenda: Agenda.....	16
Figura 2.2 – Teamup: Tela inicial.....	17
Figura 2.3 – Teamup: Agenda.....	18
Figura 3.1 – Diagrama de Caso de Uso.....	20
Figura 4.1 – Diagrama de Entidade Relacionamento.....	27
Figura 4.2 – Diagrama de Atividades: Realizar Login.....	29
Figura 4.3 – Diagrama de Atividades: Realizar Cadastro.....	29
Figura 4.4 – Diagrama de Atividades: Busca Amigos.....	30
Figura 4.5 – Diagrama de Atividades: Gerenciar Solicitações de Amizade.....	31
Figura 4.6 – Diagrama de Atividades: Ver Notificações.....	31
Figura 4.7 – Diagrama de Atividades: Gerenciar Agendamentos.....	32
Figura 4.8 – Todas as Rotas.....	34
Figura 4.9 – Rotas de Usuário.....	35
Figura 4.10 – Código Upload de Imagens.....	35
Figura 4.11 – Arquivo de configuração para Upload de Imagens.....	35
Figura 4.12 – Rota com Middleware de Upload de Imagens.....	36
Figura 4.13 – Rotas de Amigos.....	36
Figura 4.14 – Rotas de Jogos.....	36
Figura 4.15 – Rotas de Agendamentos.....	37
Figura 4.16 – Rotas de Participantes.....	37
Figura 4.17 – Rota de Envio de E-mail.....	37
Figura 5.1 – SyncFriends: Tela inicial.....	38
Figura 5.2 – SyncFriends: Sobre.....	39
Figura 5.3 – SyncFriends: Entrar.....	40
Figura 5.4 – SyncFriends: Cadastro.....	40
Figura 5.5 – SyncFriends: Painel de Controle.....	41
Figura 5.6 – SyncFriends: Solicitação de Amizade.....	42
Figura 5.7 – SyncFriends: Notificações.....	42
Figura 5.8 – SyncFriends: Buscar Amigo.....	43
Figura 5.9 – SyncFriends: Listar Agendamentos.....	44
Figura 5.10 – SyncFriends: Criar Agendamento.....	45





Escaneie a imagem para verificar a autenticidade do documento
Hash SHA256 do PDF original #8c5cace987b42fa5edc228c3f6562e4a97c5dd9c9b2244f9ed2c3dacf13c8f4
<https://valida.ae/0df721762f3507afb0b3de824f3da73c4eeb34f2a7c3c7eba>

LISTA DE TABELAS

Tabela 2.1 – Análise Comparativa das Funcionalidades entre Plataformas.	19
--	----





LISTA DE ABREVIATURAS E SIGLAS

API	-	Application Programming Interface
JSON	-	Javascript Object Notation





SUMÁRIO

1 INTRODUÇÃO	13
2 ANÁLISE DE NEGÓCIOS	15
2.1 IDENTIFICAÇÃO DO PROBLEMA.....	15
2.2 PROPOSTA DE SOLUÇÃO	15
2.3 ANÁLISE DE MERCADO	16
2.3.1 Google Agenda	16
2.3.2 Teamup	17
2.4 ANÁLISE COMPARATIVA DAS FUNCIONALIDADES DAS PLATAFORMAS.....	18
3 ANÁLISE DE REQUISITOS.....	20
3.1 DIAGRAMA DE CASO DE USO	20
3.1.1 Realizar Login	20
3.1.2 Realizar Cadastro.....	21
3.1.3 Buscar Amigos	21
3.1.4 Gerenciar Solicitações de Amizade.....	22
3.1.4.1 Criar Solicitação de Amizade.....	22
3.1.4.2 Ver Solicitação de Amizade	23
3.1.4.3 Exclui Solicitação de Amizade	23
3.1.4.4 Aceita Solicitação de Amizade.....	23
3.1.5 Ver Notificações	24
3.1.6 Gerenciar Agendamentos	24
3.1.6.1 Criar Agendamento.....	24
3.1.6.2 Ver Agendamento.....	25
3.1.6.3 Deletar Agendamento	25
3.1.6.4 Aceitar Agendamento	25
3.1.6.5 Negar Agendamento.....	26
4 ANÁLISE E DESIGN	27
4.1 DIAGRAMA DE ENTIDADE RELACIONAMENTO.....	27
4.2 DIAGRAMA DE ATIVIDADES	28
4.2.1 Realizar Login	28
4.2.2 Realizar Cadastro.....	29
4.2.3 Busca Amigos	30
4.2.4 Gerencia Solicitações de Amizade.....	30
4.2.5 Ver Notificações	31
4.2.6 Gerenciar Agendamentos	32
4.3 APPLICATION PROGRAMMING INTERFACE	32
4.3.1 Rotas da Aplicação	33
4.3.1.1 Rotas de Usuários	34
4.3.1.2 Rotas de Amigos	36
4.3.1.3 Rotas de Jogos.....	36
4.3.1.4 Rotas de Agendamentos	36
4.3.1.5 Rotas de Participantes.....	37
4.3.1.6 Rota de Envio de Email	37
5 MANUAL DO USUÁRIO	38
5.1 INTERFACES DO USUÁRIO.....	38





5.1.1 <i>Página Inicial</i>	38
5.1.2 <i>Sobre</i>	38
5.1.3 <i>Entrar</i>	39
5.1.4 <i>Cadastro</i>	40
5.1.5 <i>Painel de Controle</i>	41
5.1.6 <i>Solicitações de Amizade</i>	41
5.1.7 <i>Notificações</i>	42
5.1.8 <i>Buscar Amigos</i>	43
5.1.9 <i>Listar Agendamentos</i>	43
5.1.10 <i>Criar Agendamento</i>	44

6 CONSIDERAÇÕES FINAIS	46
-------------------------------------	-----------

REFERÊNCIAS BIBLIOGRÁFICAS	47
---	-----------





1 INTRODUÇÃO

Atualmente, em nossa sociedade, vê-se diversos indivíduos questionando-se sobre a falta de tempo presente no cotidiano, tanto para estudos, quanto para a família ou até mesmo para o trabalho, essas dificuldades encontram-se, principalmente, em quem não tem o hábito de manter em dia seus compromissos e organizar sua rotina, consequentemente, em aproveitar seu tempo que, por muitos, hoje tem a percepção de ser escasso. Com o atual avanço da tecnologia, há o surgimento de um grupo de pessoas que fazem o uso frequente de jogos digitais, o que gera a falta de comprometimento com agendamento e horários por estarem focados e entretidos nos jogos. O projeto foca em atender a demanda que as pessoas têm necessitado dos softwares de agendamento e planejamento de horários, focado no mundo dos jogos digitais, a aplicação oferece uma solução para as necessidades dos grupos de amigos que planejam estarem pontuais em seus momentos de lazer.

Segundo Koerich e Pimenta (2023) em seu artigo: “Temporalidades juvenis e impactos do contexto pandêmico”, feito com jovens rio-grandenses de idades e perspectivas diferentes, aborda como a pandemia causada pelo Covid-19 em 2020 ocasionou numa alteração na percepção de temporalidade com enfoque na organização do tempo e nas projeções futuras, onde para alguns jovens o tempo pareceu acelerado para seus objetivos e conquistas pessoais, resultando no cansaço e frustração de não conseguirem cumprir com o que foi proposto. Para outros, parecia que o tempo estava sendo arrastado, uma sensação de presente estendido, vagaroso e entediante. Com isso, pode-se ver como a falta de planejamento e agendamento destacam-se nas duas faces do problema, refletindo assim, o quão essencial é a organização do tempo para a qualidade de vida e a conquista das projeções futuras.

O mundo dos jogos digitais tem se tornado cada vez mais presente no cotidiano dos jovens, seja em computadores, videogames ou até em dispositivos móveis, estando quase diariamente presente como um momento de lazer e conforto após uma rotina estressante, porém a falta de moderação e controle de horário pode levar a impactos negativos na vida pessoal e profissional. Segundo Bassete (2022), em um estudo feito com a profissional Luiza Chagas Brandão, demonstrou que 30% dos jovens brasileiros possuem um comportamento problemático em relação ao uso dos jogos eletrônicos associando-se ao Transtorno de Jogo pela Internet (TJI). Baseando-se no artigo, é notório que uma das principais causas é a não moderação, o uso





incontrolável e por tempo indeterminado, que pode afetar a produtividade diária dos adolescentes. O agendamento de jogos que o SyncFriends possui pode ajudá-los a manterem um horário estabelecido para início e término dos momentos em grupo, fazendo-o ter consciência do tempo estabelecido para lazer.

No segundo capítulo, a Análise de Negócios, é tratado os tópicos como: Identificação do Problema, Proposta de Solução, Análise de Mercado e Análise Comparativa das Funcionalidades. No terceiro capítulo, é feita a Análise de Requisitos do sistema, onde é tratado as funcionalidades principais do sistema através do Diagrama de Caso de Uso. No quarto capítulo, Análise e Design, é apresentado o sistema através dos diagramas de Entidade Relacionamento e Atividades. O quinto capítulo é abordado o Manual do Usuário, esse tópico busca esclarecer como é o fluxo básico do sistema, para novos usuários. E por fim, o sexto capítulo, nele é descrito as considerações finais do projeto e sua resolução final.

Em suma, o sistema tem o objetivo de auxiliar os indivíduos do nicho de jogos eletrônicos, que queiram ter mais controle e organização das suas atividades de lazer relacionado aos mundos dos jogos digitais para impedir esquecimentos e falta de controle. Para o desenvolvimento será utilizada tecnologias modernas como ReactJS, NodeJS, a ferramenta de envio de e-mail Nodemailer, Insomnia para testar as rotas, GitHub para o versionamento do projeto e o banco de dados MariaDB.





2 ANÁLISE DE NEGÓCIOS

Neste capítulo será abordado a análise de negócio proposto para o projeto, bem como a identificação do problema, proposta de solução, análise de mercado, análise comparativa e, por fim, a proposta.

2.1 IDENTIFICAÇÃO DO PROBLEMA

Com base na análise feita com um grupo de pessoas do nicho de jogos digitais, notou-se que a falta de compromisso com o agendamento é algo que ocorre com frequência e que frustra o momento de lazer em conjunto.

Alguns dos principais problemas que enfrentam está relacionado com a falta da visualização do horário agendado e a motivação para uso contínuo do sistema, pois quando inserido às tarefas cotidianas, vê-se que há o esquecimento do compromisso pela alta carga mental de outras tarefas, e isso acontece justamente pela ausência de uma aplicação que o ajuda a encaixar a sua rotina no nicho de jogos digitais. Além disso, a formação de pequenas comunidades através do sistema de amigos da aplicação, é possível fixar a atenção do usuário às propostas do sistema.

2.2 PROPOSTA DE SOLUÇÃO

Utilizando a identificação do problema, propõe-se uma aplicação que faz o gerenciamento de horários com enfoque em jogos eletrônicos de multijogadores, que por meio de tecnologias como o NodeJS, ReactJS e NodeMailer, é possível o envio de um e-mail de aviso para os participantes sobre o compromisso marcado anteriormente, e a posterior consulta de todos os agendamentos por parte dos participantes. Essa resolução contempla uma interface amigável ao usuário, que o ajudará na utilização do sistema pela praticidade e intuitividade.

Com o cumprimento da proposta, espera-se que os participantes e usuários consigam, através da solução, estarem com mais exatidão nos compromissos e que desfrutem melhor de seus momentos de divertimento mútuo.



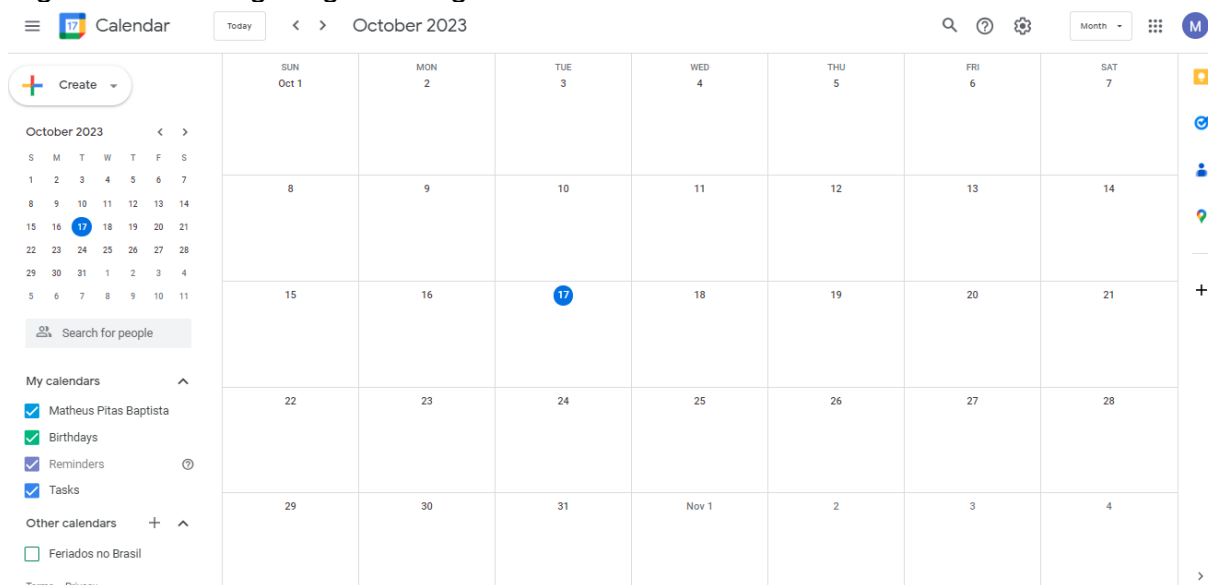


2.3 ANÁLISE DE MERCADO

Para o capítulo de análise de mercado, será utilizado para comparação os principais sistemas de agendamento disponíveis, bem como suas funcionalidades e design.

2.3.1 Google Agenda

Figura 2.1 – Google Agenda: Agenda.



Fonte: Google Agenda, 2023.

Como demonstrado na Figura 2.1, Uma das soluções de agendamento mais famosas que existem atualmente é o Google Agenda, essa aplicação da Google permite que diversas pessoas possam gerenciar seus horários simultaneamente e consigam visualizar as atividades de outras pessoas para saber se pode solicitar algo em um determinado horário.

Com o seu design minimalista, a aplicação gerencia os horários a partir de espaços quadrados em branco que podem ser preenchidos com as marcações de reuniões, lembretes, avisos e tarefas, onde cada vazão é o dia do mês, mas caso precise, pode-se alterar o escopo pela aba do menu, podendo visualizar através de semana, mês e ano.

É notável que sua interface limpa e intuitiva tenha um impacto enorme na popularidade e utilização do sistema, o agendamento em conjunto e sua integração com outros aplicativos da própria empresa como o “Gmail”, também é um ponto

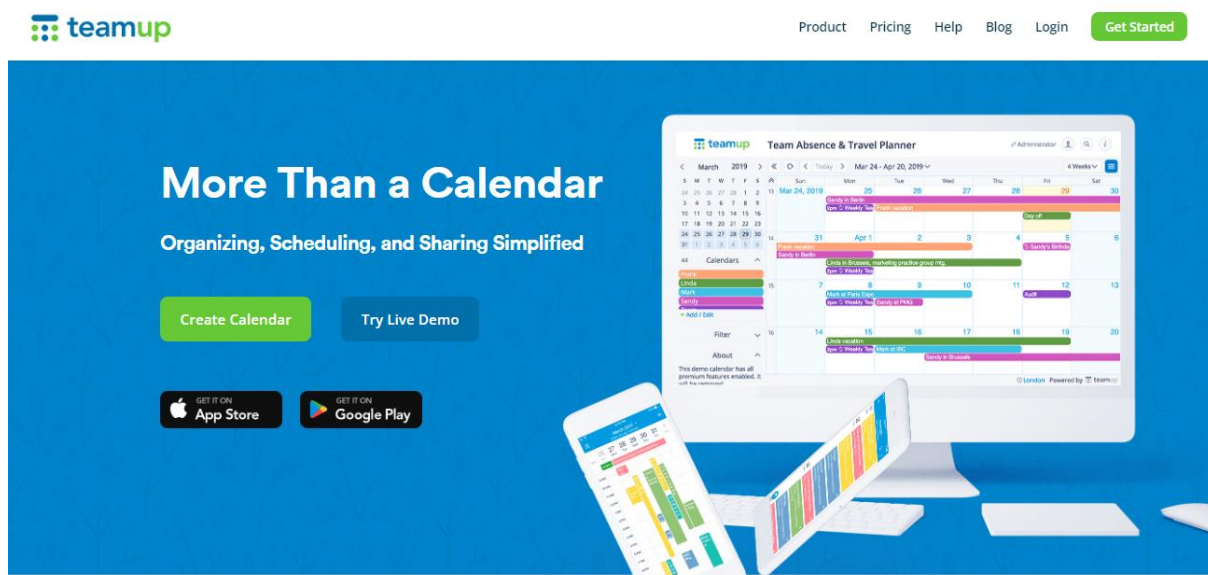




notório de destaque, assim como a possibilidade de ver os feriados nacionais e assim planejar melhor o que fazer nesses dias. Com isso, percebe-se que é um sistema muito bom para algo genérico que atende diversos tipos de pessoas e situação, mas que para nichos não é assertiva.

2.3.2 Teamup

Figura 2.2 – Teamup: Tela inicial.



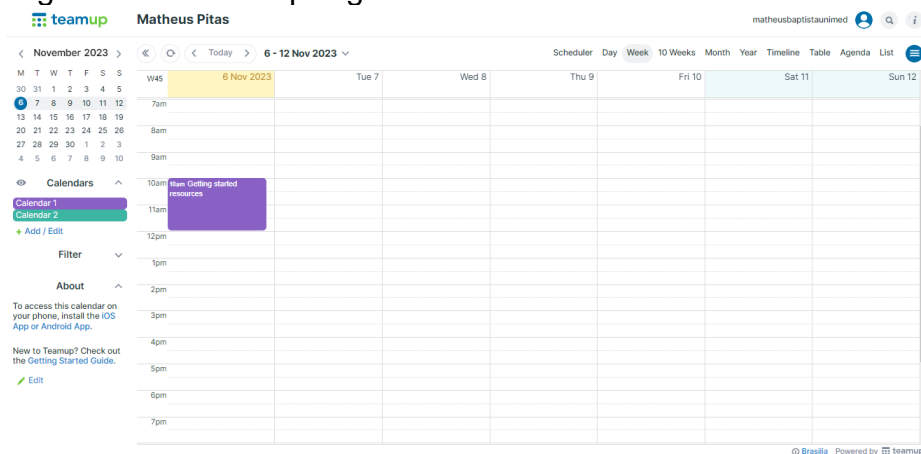
Fonte: Teamup, 2023.

O Teamup, apresentado na figura 2.2, é um sistema de gerenciamento de agenda online projetado para equipes que queiram ter mais controle sobre os horários e reuniões, o sistema possui recursos como criação, gerenciamento e compartilhamento de cronograma e calendário com grupos, equipes e organizações. A solução possui diversas maneiras de personalização para deixá-lo com o visual mais próximo do que o usuário queira, além disso, possui um algoritmo de recorrência de eventos, onde o usuário pode utilizá-lo para marcar todos os seus eventos semanais ou mensais.





Figura 2.3 – Teamup: Agenda.



Fonte: Teamup, 2023.

A aplicação demonstra ser uma ótima solução para o que propõe, porém tem um visual não intuitivo e pouco agradável, além disso, para que os usuários possam realmente manter-se no sistema com o melhor que ele oferece, deverão optar pela opção paga, pois, a gratuita é limitada em funções essenciais como a criação de calendários.

2.4 ANÁLISE COMPARATIVA DAS FUNCIONALIDADES DAS PLATAFORMAS

Contextualizada as informações referentes aos softwares citados acima, será mostrada na figura 2.1 a análise comparativa das funcionalidades entre os programas adjacente às do aplicativo proposto, sendo uma tabela dividida entre as funcionalidades e os aplicativos.





Tabela 2.1 – Análise Comparativa das Funcionalidades entre Plataformas.

Funcionalidades	Google Agenda	Teamup	SyncFriends
Gratuito	X	X	X
Consulta de Agendamentos	X	X	X
Grade de Visualização	X	X	
Filtros de Datas	X	X	
Criação de Agendamentos	X	X	X
Busca de Agendamentos	X	X	
Lista de Amizades			X
Seleção de Jogos			X
Envio de Email	X	X	X
Cadastro	X	X	X
Login	X	X	X

Fonte: Elaborado pelo autor, 2023.

Demonstrada as funcionalidades e feita a comparação, percebemos que o aplicativo possui pontos fortes naquilo que é proposto a fazer para o nicho especificado como: Lista de Amizades e Seleção de Jogos.





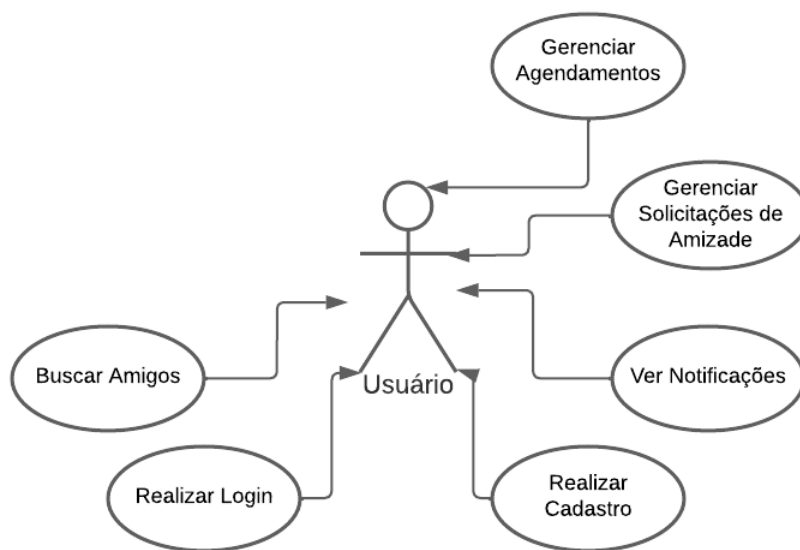
3 ANÁLISE DE REQUISITOS

Neste capítulo será demonstrada a Análise de Requisitos, onde é possível entender qual será a estrutura para satisfazer a demanda proposta, levantando informações e criando diagramas que expliquem o fluxo das funcionalidades do sistema.

3.1 DIAGRAMA DE CASO DE USO

Segundo a IBM(2021), os diagramas de caso de uso conseguem modelar o comportamento de um sistema e quais requisitos serão necessários, além de identificar as interações entre um sistema e seus agentes. Mediante a isso, o sistema dispõe do seguinte caso de uso, apresentado na figura 3.1:

Figura 3.1 – Diagrama de Caso de Uso.



Fonte: Elaborado pelo autor, 2023.

3.1.1 Realizar Login

Nesse caso de uso, o usuário será responsável por fazer o preenchimento dos dados nos sistemas, que fará a verificação com a base de dados disponível e autenticará o agente.



**Fluxo Básico:**

O fluxo básico começa quando o usuário clicar na aba “Entrar” do sistema.

Fb1 – O usuário inicia o caso de uso.

Fb2 – O usuário preenche os campos necessários.

Fb3 – O sistema verifica as credenciais fornecidas.

Fb4 – O sistema valida as credenciais.

Fb5 – O sistema transfere o usuário para o painel de controle.

Fb6 – O caso de uso se encerra.

Fluxos Alternativos:

Fa1 – Credenciais incorretas. No item Fb2, caso o usuário preencha incorretamente os dados necessário, o sistema informará um erro e será necessário refazer o preenchimento.

3.1.2 Realizar Cadastro

Nesse caso de uso, o usuário será responsável por fazer seu próprio cadastro no sistema através de seus dados pessoais.

Fluxo Básico:

O fluxo básico começa quando o usuário clicar na aba “Entrar” do sistema e logo após na opção “Cadastrar”.

Fb1 – O usuário inicia o caso de uso.

Fb2 – O usuário preenche os campos necessários.

Fb3 – O sistema verifica os dados fornecidos.

Fb4 – O sistema armazena os dados.

Fb5 – O caso de uso se encerra.

Fluxos Alternativos:

Fa1 – Dados fornecidos incorretos. No item Fb2, caso o usuário não preencha ou preencha incorretamente os dados, o sistema informará um erro e será necessário corrigir o preenchimento dos campos.

3.1.3 Buscar Amigos



Nesse caso de uso, o usuário será responsável por buscar novas pessoas para adicionar.

Fluxo Básico:

O fluxo básico começa quando o usuário clicar na aba “Buscar Amigos” do painel de controle.

Fb1 – O usuário inicia o caso de uso.

Fb2 – O usuário preenche o campo de busca.

Fb3 – O sistema verifica o dado fornecido.

Fb4 – O sistema retorna os usuários que pertencem a busca.

Fb5 – O caso de uso se encerra.

3.1.4 Gerenciar Solicitações de Amizade

Nesse caso de uso, o usuário será responsável por manter a solicitação de amizade.

3.1.4.1 Criar Solicitação de Amizade

Nesse caso de uso, o usuário será responsável por criar a solicitação de amizade.

Fluxo Básico:

O fluxo básico começa quando o usuário está na aba “Buscar Amigos” e fez a pesquisa de algum nome.

Fb1 – O usuário inicia o caso de uso.

Fb2 – O usuário clica no botão “Adicionar Amigo”.

Fb3 – O usuário clica no botão “Enviar Solicitação”.

Fb4 – O sistema envia um pedido de amizade para a pessoa selecionada.

Fb5 – O caso de uso se encerra.

Fluxos Alternativos:

Fa1 – Credenciais incorretas. No item Fb3, o usuário pode clicar em “Cancelar Solicitação” onde irá voltar ao início da solicitação.





3.1.4.2 Ver Solicitação de Amizade

Nesse caso de uso, o usuário será responsável por ver a solicitação de amizade.

Fluxo Básico:

O fluxo básico começa quando o usuário está no painel de controle.

Fb1 – O usuário inicia o caso de uso.

Fb2 – O usuário clica na aba “Adicionar Amigos”.

Fb3 – O sistema mostra todas as solicitações de amizade disponíveis.

Fb4 – O caso de uso se encerra.

3.1.4.3 Exclui Solicitação de Amizade

Nesse caso de uso, o usuário será responsável por excluir a solicitação de amizade.

Fluxo Básico:

O fluxo básico começa quando o usuário está na aba “Adicionar Amigos”.

Fb1 – O usuário inicia o caso de uso.

Fb2 – O usuário clica no botão vermelho com um “X”

Fb3 – O sistema exclui a solicitação.

Fb4 – O caso de uso se encerra.

3.1.4.4 Aceita Solicitação de Amizade

Nesse caso de uso, o usuário será responsável por aceitar a solicitação de amizade.

Fluxo Básico:

O fluxo básico começa quando o usuário está na aba “Adicionar Amigos”.

Fb1 – O usuário inicia o caso de uso.

Fb2 – O usuário clica no botão verde.

Fb3 – O sistema aceita a solicitação.





Fb4 – O caso de uso se encerra.

3.1.5 Ver Notificações

Nesse caso de uso, o usuário será responsável por ver as notificações

Fluxo Básico:

O fluxo básico começa quando o usuário está no painel de controle.

Fb1 – O usuário inicia o caso de uso.

Fb2 – O usuário clica na aba “Notificações”.

Fb3 – O sistema mostra as notificações.

Fb4 – O caso de uso se encerra.

3.1.6 Gerenciar Agendamentos

Nesse caso de uso, o usuário será responsável por fazer o preenchimento dos dados no sistema, que fará a verificação com a base de dados disponível e autenticará o agente.

3.1.6.1 Criar Agendamento

Nesse caso de uso, o usuário será responsável por criar o agendamento

Fluxo Básico:

O fluxo básico começa quando o usuário está no painel de controle.

Fb1 – O usuário inicia o caso de uso.

Fb2 – O usuário clica no botão “Agendar”.

Fb3 – O usuário preenche todos os campos e clica em “Agendar”.

Fb4 – O sistema salva as informações no banco.

Fb4 – O sistema envia os convites para os usuários selecionados.

Fb5 – O caso de uso se encerra.

Fluxos Alternativos:

Fa1 – Credenciais incorretas. No item Fb3, caso o usuário não preencha corretamente o sistema irá retornar o erro.





3.1.6.2 Ver Agendamento

Nesse caso de uso, o usuário será responsável por ver o agendamento

Fluxo Básico:

O fluxo básico começa quando o usuário está no painel de controle.

Fb1 – O usuário inicia o caso de uso.

Fb2 – O usuário clica na aba “Listar Agendamento”.

Fb3 – O sistema retorna todos os agendamentos do usuário.

Fb4 – O caso de uso se encerra.

3.1.6.3 Deletar Agendamento

Nesse caso de uso, o usuário será responsável por deletar o agendamento

Fluxo Básico:

O fluxo básico começa quando o usuário está na aba “Listar Agendamento”.

Fb1 – O usuário inicia o caso de uso.

Fb2 – O usuário clica no botão “Remover”.

Fb3 – O sistema deleta o agendamento do usuário.

Fb4 – O caso de uso se encerra.

3.1.6.4 Aceitar Agendamento

Nesse caso de uso, o usuário será responsável por aceitar o agendamento

Fluxo Básico:

O fluxo básico começa quando o usuário está na aba “Notificações”.

Fb1 – O usuário inicia o caso de uso.

Fb2 – O usuário clica no botão roxo.

Fb3 – O sistema ativa o agendamento.

Fb4 – O caso de uso se encerra.





3.1.6.5 Negar Agendamento

Nesse caso de uso, o usuário será responsável por negar o agendamento

Fluxo Básico:

O fluxo básico começa quando o usuário está na aba “Notificações”.

Fb1 – O usuário inicia o caso de uso.

Fb2 – O usuário clica no botão preto com “X”.

Fb3 – O sistema tira a participação do agendamento.

Fb4 – O caso de uso se encerra.





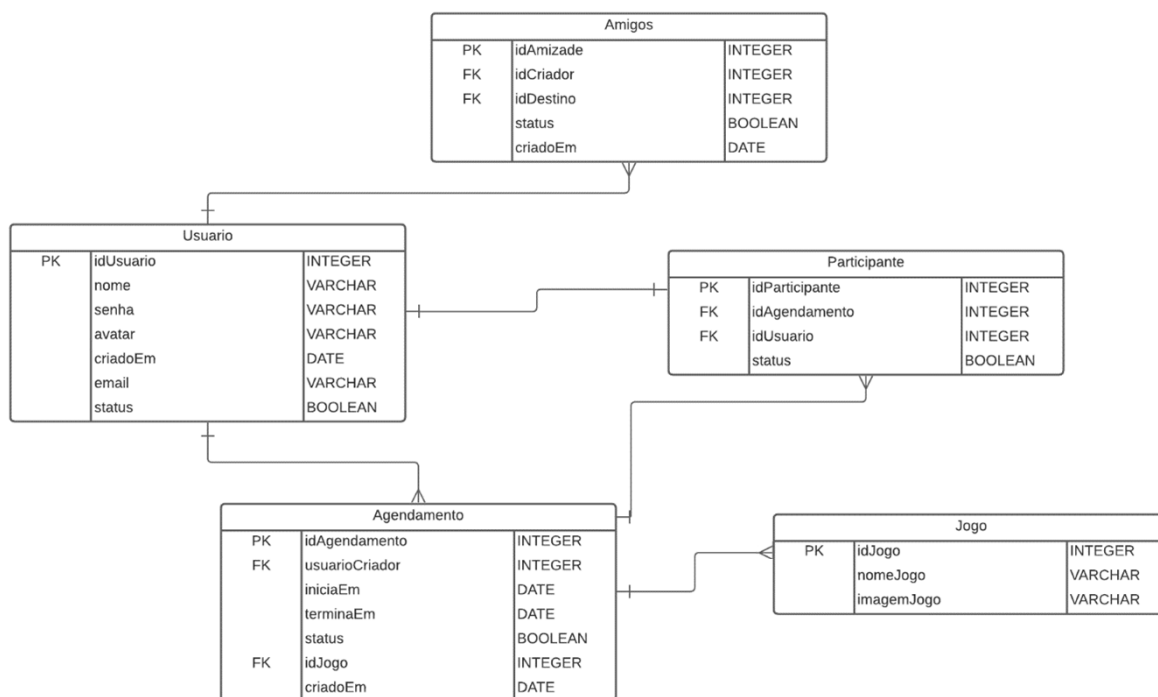
4 ANÁLISE E DESIGN

4.1 DIAGRAMA DE ENTIDADE RELACIONAMENTO

Neste tópico será abordado o Diagrama de Entidade Relacionamento que segundo a Danielle Oliveira (2023), instrutora de Data Science, o DER é a forma visual do Modelo de Entidade Relacionamento, que descreve os objetos da realidade em formato de entidades, que são ligadas por meio de relacionamentos através de suas propriedades.

Contextualizado o conceito, a figura 4.1 abaixo mostra o Diagrama de Entidade Relacionamento do sistema SyncFriends.

Figura 4.1 – Diagrama de Entidade Relacionamento



Fonte: Elaborado pelo autor, 2023.

Nesse diagrama, nota-se a existência de 5 entidades (Usuário, Agendamento, Jogo, Participante e Amigos), onde Usuário possui um relacionamento de um para muitos agendamentos, significando que um usuário pode criar diversos agendamentos, também possui relacionamento com Participante, delimitando que todo participante deverá ser um usuário através do relacionamento de um para um, e o último relacionamento de usuário é com amigos, demonstrando o relacionamento de um para muitos, ou seja, um usuário poderá ter diversos amigos.





Em agendamento, os relacionamentos são com Usuário, como explicado anteriormente, com Jogo, onde mostra que um agendamento pode ter somente um jogo, e por fim com participante, essa tabela servirá como apoio ao agendamento, nela terá os registros e as respostas dos participantes, que quando respondido, mudará o status do agendamento para ativo ou inativo.

4.2 DIAGRAMA DE ATIVIDADES

Neste tópico será abordado o diagrama de atividades, segundo o site da IBM (2021), esse diagrama serve como uma representação do comportamento do sistema baseado nas sequências de ações de um processo, são semelhantes a fluxogramas, porém podem mostrar fluxos alternativos de um determinado processo.

Demonstrarei a seguir, os seguintes diagramas de atividades que fazem parte do sistema:

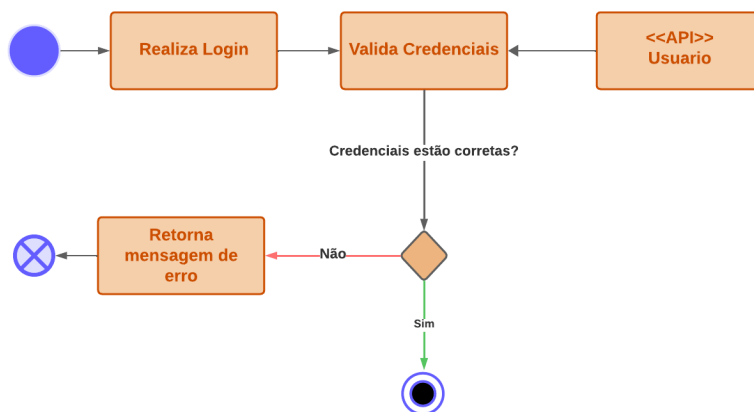
4.2.1 Realizar Login

Neste fluxo do diagrama (figura 4.2), o usuário colocará as credenciais de acesso e o sistema irá validar com os dados retornados da API interna do projeto, caso seja satisfeita as condições, o usuário completará o fluxo de login, caso não seja satisfeito, o sistema retornará uma mensagem de erro e encerrará o fluxo.





Figura 4.2 – Diagrama de Atividades: Realizar Login.

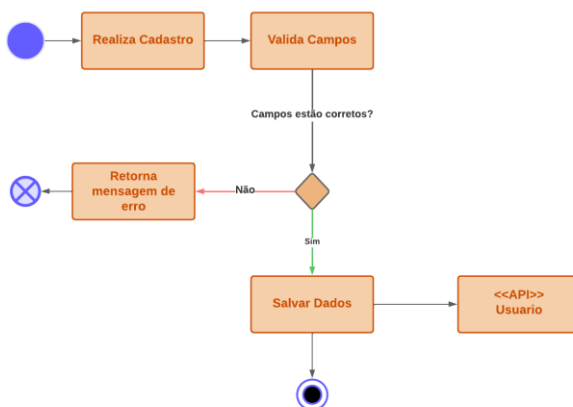


Fonte: Elaborado pelo autor, 2023.

4.2.2 Realizar Cadastro

Neste fluxo do diagrama (figura 4.3), o usuário é responsável por fazer seu cadastro no sistema, quando preenchido o formulário, o sistema irá validar os campos e caso estejam todos corretos, será salvo os dados através das API interna de usuários, caso algum esteja incorreto, o sistema retornará uma mensagem de erro e encerrará o fluxo.

Figura 4.3 – Diagrama de Atividades: Realizar Cadastro.



Fonte: Elaborado pelo autor, 2023.

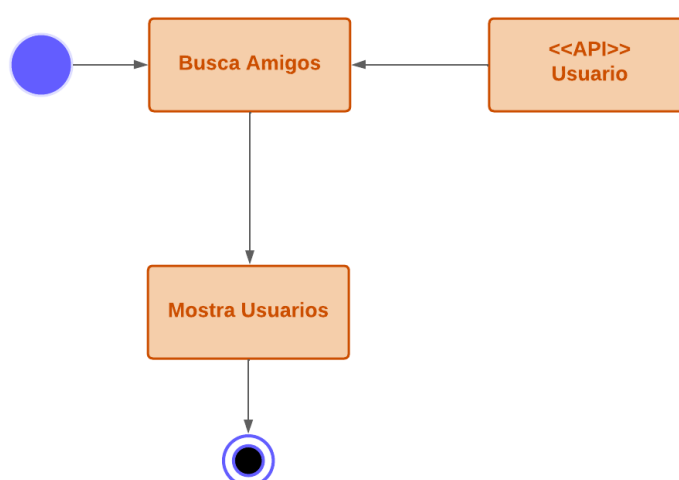




4.2.3 Busca Amigos

Neste fluxo (figura 4.4), o usuário entrará na aba “Buscar Amigos” e pesquisará o nome desejado, assim o sistema retornará todos os cadastros de usuários disponíveis com aquele nome.

Figura 4.4 – Diagrama de Atividades: Busca Amigos.



Fonte: Elaborado pelo autor, 2023.

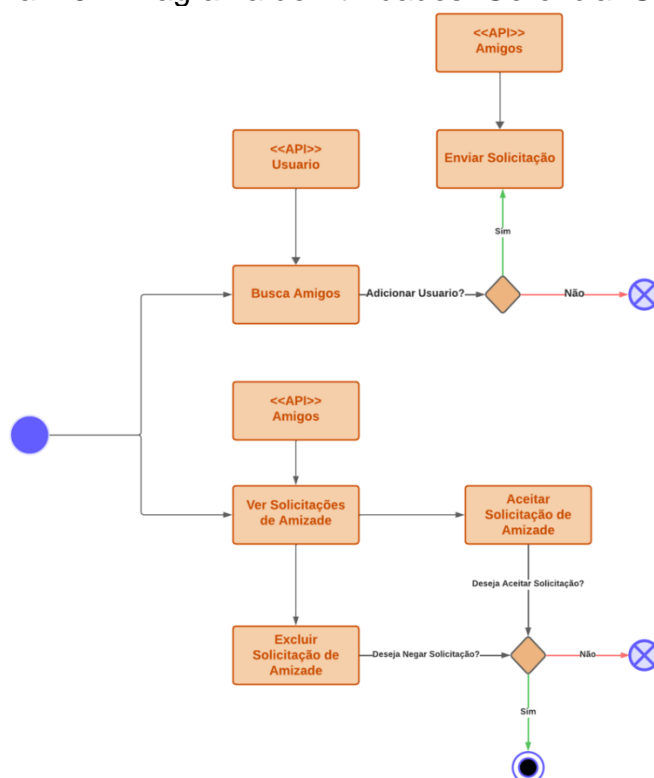
4.2.4 Gerencia Solicitações de Amizade

Neste fluxo (figura 4.5), o usuário poderá seguir por diferentes caminhos para obter o fim dele, começando pelo fluxo de “Adicionar Amigo”. O usuário, após a busca feita no tópico anterior, poderá enviar uma solicitação de amizade para o usuário selecionado, encerrando esse caminho. Para a rota de “Ver solicitações”, encontrará um novo fluxo de aceitar ou negar as solicitações de outros usuário pendentes para ele.





Figura 4.5 – Diagrama de Atividades: Gerenciar Solicitações de Amizade.

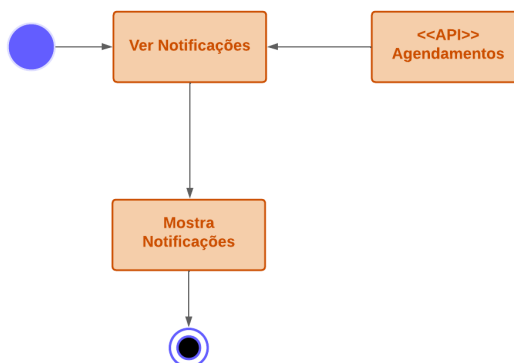


Fonte: Elaborado pelo autor, 2023.

4.2.5 Ver Notificações

Este fluxo (figura 4.6), demonstra o usuário consultando quais notificações existem para ele ver. Quando selecionada a aba de “Notificações”, o sistema irá mostrar automaticamente quais agendamentos estão disponíveis para aceitar e recusar.

Figura 4.6 – Diagrama de Atividades: Ver Notificações



Fonte: Elaborado pelo autor, 2023.

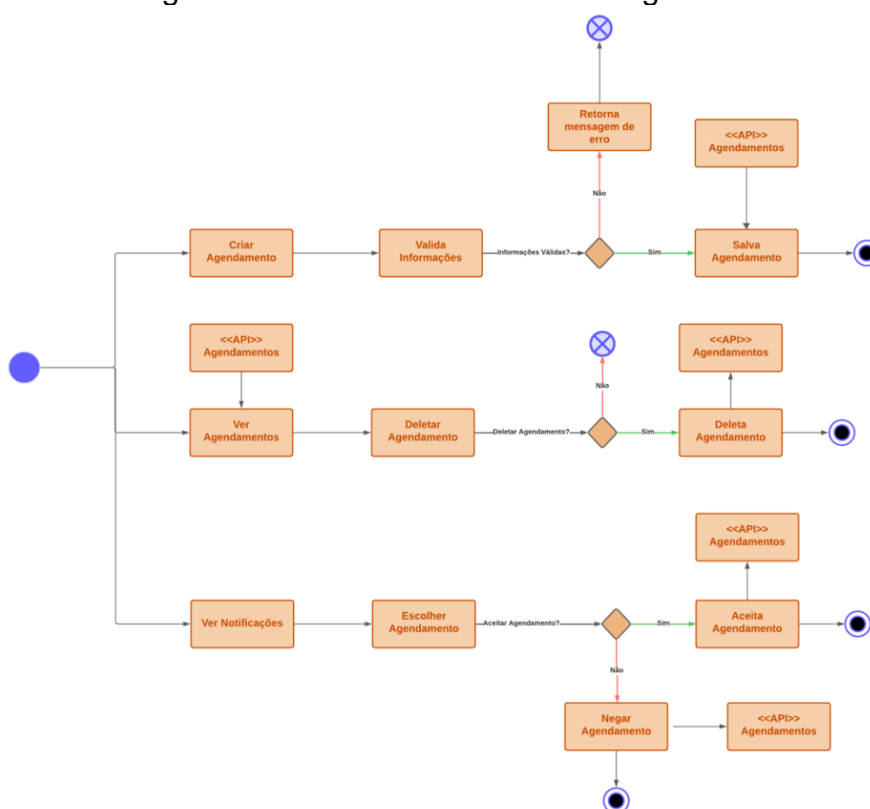




4.2.6 Gerenciar Agendamentos

Neste diagrama (figura 4.7), o usuário pode seguir diferentes caminhos para concluí-lo, começando com “Criar Agendamento”, onde o usuário preencherá todas as informações para marcá-lo, o sistema validará se estão pertinentes as regras de agendamento, e então salvar através da API interna de agendamentos. Outro caminho será o de “Ver Agendamentos”, onde o sistema automaticamente informa todos os agendamentos para o usuário, caso o usuário os delete, será enviada uma chamada para a API de agendamentos informando a deleção. Por fim, através das notificações o usuário poderá aceitar ou negar um agendamento, feito por um de seus amigos, através da API de agendamentos.

Figura 4.7 – Diagrama de Atividades: Gerenciar Agendamentos.



Fonte: Elaborado pelo autor, 2023.

4.3 APPLICATION PROGRAMMING INTERFACE

Application Programming Interface ou API, segundo o site da Amazon (2023), pode ser explicado como um contrato de serviço entre duas aplicações, onde define como as duas irão se comunicar utilizando solicitações e respostas.





Funcionam com uma estrutura cliente e servidor, onde é feita uma solicitação do cliente para o servidor, que responde com os dados solicitados, existem quatro formas de funcionamento de uma API, dependendo do seu uso:

- APIs SOAP
- APIs RPC
- APIs WebSocket
- APIs REST

A **API SOAP** funciona usando o Simple Object Access Protocol (Protocolo de Acesso a Objetos Simples), onde a troca de mensagens entre servidor e cliente é feito através de arquivos XML.

A **API RPC** ou conhecida como Remote Procedure Calls (Chamadas de Procedimento Remoto), funcionam com a conclusão de uma função ou procedimento que esteja no servidor e então ele devolve os dados para o cliente.

A **API WebSocket**, é um método que utiliza objetos JSON para trafegar informações, oferecendo suporte à comunicação bidirecional entre cliente e servidor.

E por fim, a **API REST**, funciona a partir da padronização de métodos como GET, POST, DELETE, entre outros. Assim, o cliente e servidor podem usar a transferências de dados a partir do protocolo HTTP. Sua principal característica é a ausência de estado, significando que, a resposta do servidor virá em uma informação simples e não em uma representação gráfica, aumentando assim o desempenho da aplicação.

4.3.1 Rotas da Aplicação

A aplicação SyncFriends faz o uso de uma API REST interna feita em NodeJS, demonstrado na figura 4.8, para consumir e inserir dados no banco de dados, e este tópico abordará quais são elas. Todas as rotas ligam-se em um endereço eletrônico fixo “http://localhost:3333/”, o que muda é o endpoint, ou seja, o final do endereço eletrônico.





Figura 4.8 – Todas as Rotas.

```
const usersController = require('./controller/UsuariosController')
const friendsController = require('./controller/AmigosController')
const gamesController = require('./controller/JogosController')
const schedulingController = require('./controller/AgendamentoController')
const participantController = require('./controller/ParticipantesController')
const emailSender = require('./email/sendingEmail')

router.get("/", (require, response) => {
  response.send("Root Application.")
})

// Rotas de Usuario
router.get("/users", usersController.getAllUsersController)
router.get("/users/:id", usersController.getUserByIdController)
router.post("/auth/user", usersController.authUserLoginController)
router.post("/users/create", upload.single('avatar'), usersController.addUserController)
router.delete("/users/delete/:id", usersController.deleteUserController)
router.put("/users/put/:id", usersController.updatedUserController)
router.post("/users/search/:id", usersController.getUsersForSearchController)
router.use("/uploads", express.static(path.resolve('uploads')))

// Rotas de Amigos
router.post("/friends/create", friendsController.addAmigoController)
router.post("/friends/verify", friendsController.verifyAmigosController)
router.post("/friends/list", friendsController.listAmigosController)
router.post("/friends/request", friendsController.listRequestController)
router.post("/friends/request/response/:id", friendsController.AcceptRequestController)

// Rota de Jogos
router.get("/game", gamesController.getAllJogosController)
router.post("/game/create", upload.single('imagemJogo'), gamesController.addJogoController)

// Rota de Agendamento
router.post("/scheduling/create", schedulingController.addAgendamentoController)
router.get("/scheduling/:id", schedulingController.getAgendamentoByIdController)
router.get("/scheduling/next/:id", schedulingController.getNearAgendamentoByIdController)
router.get("/scheduling/last/:id", schedulingController.getLastAgendamentoByIdController)
router.get("/scheduling/verify/:id", schedulingController.verifyStatusAgendamentoController)
router.delete("/scheduling/delete/:id", schedulingController.deleteAgendamentoController)

// Rota de Participante
router.post("/participant/create", participantController.addParticipantesController)
router.post("/participant", participantController.getParticipantesController)
router.get("/participant/schedule/:id", participantController.getSchedulingByParticipantsController)
router.get("/participant/schedule/pending/:id", participantController.getPendingSchedulingByParticipantsController)
router.put("/participant/schedule/response/:id", participantController.putPendingSchedulingController)

// Rota de Envio de E-mail
router.post("/sending/email", emailSender.sendingEmails)

module.exports = router
```

Fonte: Elaborado pelo autor, 2023.

4.3.1.1 Rotas de Usuários

Estas rotas (figura 4.9), tratam de todos os métodos do usuário, como por exemplo, listar todos os usuários cadastrados, criar usuário, autenticar login de usuário, entre outros.





Figura 4.9 – Rotas de Usuário.

```
// Rotas de Usuario
router.get("/users", usersController.getAllUsersController)
router.get("/users/:id", usersController.getUserByIdController)
router.post("/auth/user", usersController.authUserLoginController)
router.post("/users/create", upload.single('avatar'), usersController.addUserController)
router.delete("/users/delete/:id", usersController.deleteUserController)
router.put("/users/put/:id", usersController.updatedUserController)
router.post("/users/search/:id", usersController.getUsersForSearchController)
router.use("/uploads", express.static(path.resolve('uploads')))
```

Fonte: Elaborado pelo autor, 2023.

Na criação do usuário junto a uma foto de perfil (figura 4.10), é utilizada a biblioteca de upload “Multer”, onde criptografa, em uma base de 16 dígitos hexadecimais, o nome do arquivo e o transfere para uma pasta na raiz chamada “uploads”.

Figura 4.10 – Código Upload de Imagens

```
const multer = require('multer')
const upload = multer({ storage: storage })
const path = require('path')
```

Fonte: Elaborado pelo autor, 2023.

Para conseguir esse resultado, necessita-se da seguinte codificação :

Figura 4.11 – Arquivo de configuração para Upload de Imagens

```
const multer = require('multer')
const path = require('path')
const crypto = require('crypto')

const storage = multer.diskStorage({
  destination: function (req, file, cb) {
    cb(null, path.resolve('uploads'))
  },
  filename: function (req, file, cb) {
    crypto.randomBytes(16, (err, hash) => {
      if (err) cb(err)
      const fileName = `${hash.toString("hex")}-${file.originalname}`
      cb(null, fileName)
    })
  }
})

module.exports = {
  storage
}
```

Fonte: Elaborado pelo autor, 2023.





Após isso, coloca-se o código como um middleware da rota (figura 4.12), significando que quando chamada, é executado um script de validação e codificação da imagem.

Figura 4.12 – Rota com Middleware de Upload de Imagens

```
router.post("/users/create", upload.single('avatar'), usersController.addUserController)
```

Fonte: Elaborado pelo autor, 2023.

4.3.1.2 Rotas de Amigos

Essas rotas (figura 4.13), referem-se a parte de amizades do sistema, onde ligará usuários que se tornam amigos por uma tabela no banco de dados chamada “Amigos”, nelas serão relacionados dois usuários, um criador da amizade e outro que é o destino do convite, além de um status mostrando se está ativa ou não. Oferecendo também, acesso aos métodos de listar, criar e verificar uma amizade.

Figura 4.13 – Rotas de Amigos.

```
// Rotas de Amigos
router.post("/friends/create", friendsController.addAmigoController)
router.post("/friends/verify", friendsController.verifyAmigosController)
router.post("/friends/list", friendsController.listAmigosController)
router.post("/friends/request", friendsController.listRequestController)
router.post("/friends/request/response/:id", friendsController.AcceptRequestController)
```

Fonte: Elaborado pelo autor, 2023.

4.3.1.3 Rotas de Jogos

Essas rotas (figura 4.14), serão responsáveis pela criação e a consulta de todos os jogos disponíveis para agendamento no sistema.

Figura 4.14 – Rotas de Jogos.

```
// Rota de Jogos
router.get("/game", gamesController.getAllJogosController)
router.post("/game/create", upload.single('imagemJogo'), gamesController.addJogoController)
```

Fonte: Elaborado pelo autor, 2023.

Também seguindo a premissa do upload de fotos dos usuários através da biblioteca “Multer”.

4.3.1.4 Rotas de Agendamentos





Essas rotas (figura 4.15), serão responsáveis por todo o sistema de agendamento, nelas têm-se a criação, deleção, verificação, além de ter a consulta de dados do painel de controle de último jogo e de próximo jogo.

Figura 4.15 – Rotas de Agendamentos.

```
// Rota de Agendamento
router.post("/scheduling/create", schedulingController.addAgendamentoController)
router.get("/scheduling/:id", schedulingController.getAgendamentoByIdController)
router.get("/scheduling/next/:id", schedulingController.getNearAgendamentoByIdController)
router.get("/scheduling/last/:id", schedulingController.getLastAgendamentoByIdController)
router.get("/scheduling/verify/:id", schedulingController.verifyStatusAgendamentoController)
router.delete("/scheduling/delete/:id", schedulingController.deleteAgendamentoController)
```

Fonte: Elaborado pelo autor, 2023.

4.3.1.5 Rotas de Participantes

Estas rotas (figura 4.16), serão responsáveis por criar e registrar os participantes de um determinado agendamento, todos eles constituídos de usuários que aceitaram ou negaram o registro de um agendamento.

Figura 4.16 – Rotas de Participantes.

```
// Rota de Participante
router.post("/participant/create", participantController.addParticipantesController)
router.post("/participant", participantController.getParticipantesController)
router.get("/participant/schedule/:id", participantController.getSchedulingByParticipantsController)
router.get("/participant/schedule/pending/:id", participantController.getPendingSchedulingByParticipantsController)
router.put("/participant/schedule/response/:id", participantController.putPendingSchedulingController)
```

Fonte: Elaborado pelo autor, 2023.

4.3.1.6 Rota de Envio de Email

Essa rota (figura 4.17), faz o envio de e-mail na criação de um agendamento para os participantes que foram colocados como participantes, nele é informado o jogo, a data e quem os convidou, assim os usuários podem ficar cientes e respondê-lo.

Figura 4.17 – Rota de Envio de E-mail.

```
// Rota de Envio de E-mail
router.post("/sending/email", emailSender.sendingEmails)
```

Fonte: Elaborado pelo autor, 2023.





5 MANUAL DO USUÁRIO

Nesse capítulo, aborda-se o visual das telas desenvolvidas na aplicação, demonstrando as principais funcionalidades do projeto, bem como sua identidade visual e design.

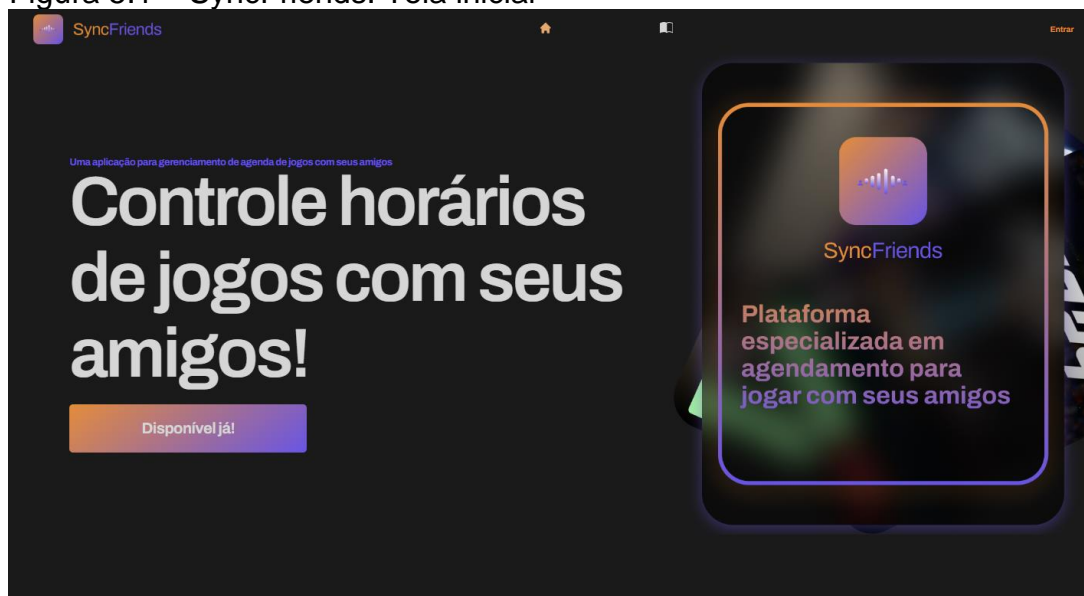
5.1 INTERFACES DO USUÁRIO

Neste tópico, pode-se ver como ficou todas as telas que o usuário terá acesso na aplicação, o foco principal de cada uma é fornecer ao usuário um fluxo intuitivo e de fácil utilização.

5.1.1 Página Inicial

Na Figura 5.1, tem-se a página inicial do projeto como uma “Landing Page”, isto é, uma página de apresentação que tem como objetivo principal converter visitantes em clientes. Nela há o botão de entrada do sistema, e no menu na parte superior há dois ícones, um da página inicial e outro da página informativa “Sobre”.

Figura 5.1 – SyncFriends: Tela inicial



Fonte: Elaborado pelo autor, 2023.

5.1.2 Sobre





A interface “Sobre” na Figura 5.2, refere-se as principais informações do projeto e do autor, referenciando seu objetivo. Neste está descrito quem o desenvolveu e para qual fim foi utilizado, além de especificar para qual instituição de ensino o sistema foi apresentado.

Figura 5.2 – SyncFriends: Sobre



Fonte: Elaborado pelo autor, 2023.

5.1.3 Entrar

Na Figura 5.3, mostra-se o método de entrada do sistema, feito através de credenciais pré-cadastrados, caso ainda não tenha as chaves de acesso, será necessário clicar no botão “Cadastrar” em laranja abaixo do botão “Entrar” em roxo.





Figura 5.3 – SyncFriends: Entrar

Fonte: Elaborado pelo autor, 2023.

5.1.4 Cadastro

Na Figura 5.4, tem-se o cadastro do usuário, caso ainda não tenha sido cadastrado no sistema, com os campos de nome, e-mail, senha, confirmação de senha e avatar, que será a imagem do usuário no sistema. Após preenchidos, o usuário clica no botão roxo “Cadastrar” e o sistema retornará se o cadastro foi efetivado, após o cadastro, o usuário pode voltar a tela de entrada através do botão laranja escrito “Voltar”.

Figura 5.4 – SyncFriends: Cadastro

Fonte: Elaborado pelo autor, 2023.

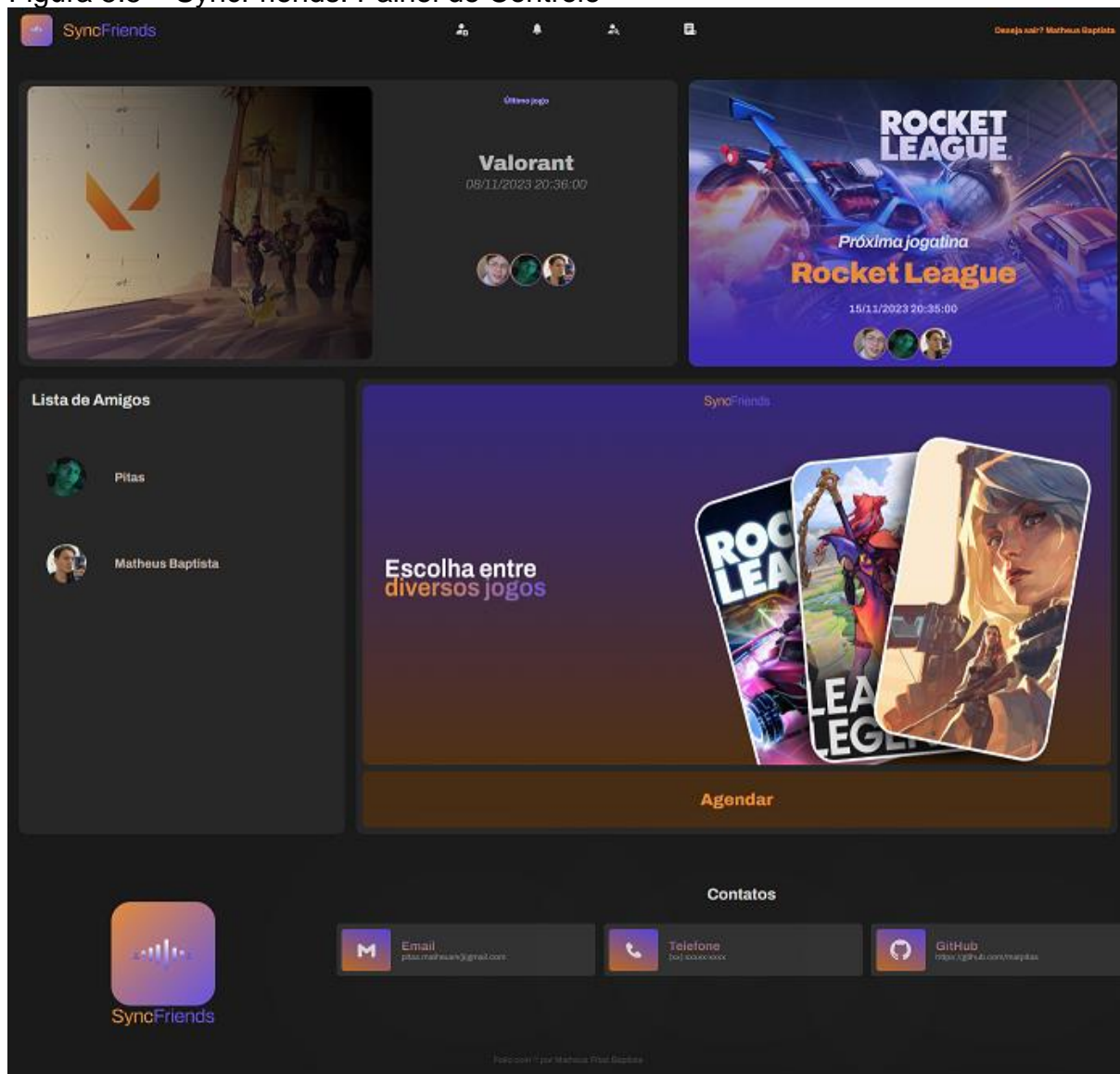




5.1.5 Painel de Controle

Na Figura 5.5, é demonstrado como fica o painel de controle com todos os dados preenchidos como: Lista de amigos, último jogo, Próximo jogo. Há também o menu que mostra os ícones das interfaces: Solicitações de Amizade, Notificações, Buscar Amigos e Listar Agendamentos. E no fim do menu, nota-se um texto com o nome do usuário que será o botão de saída do sistema.

Figura 5.5 – SyncFriends: Painel de Controle



Fonte: Elaborado pelo autor, 2023.

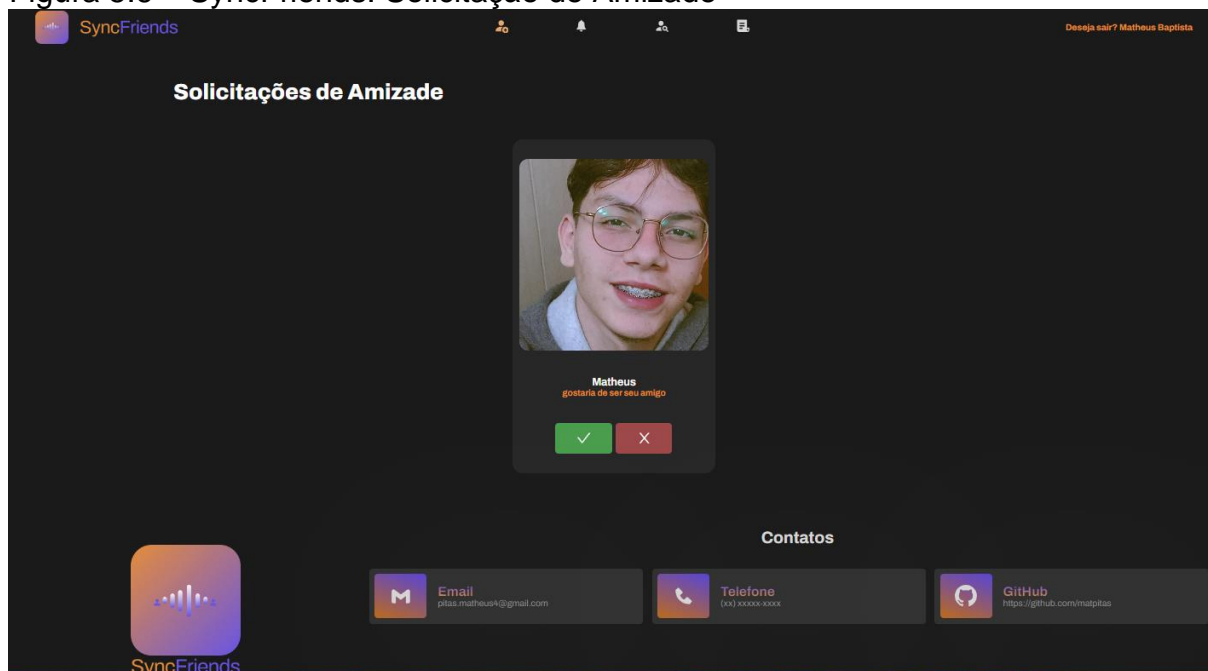
5.1.6 Solicitações de Amizade





Na Figura 5.6, mostra como é a tela de solicitações de amizade quando há uma solicitação pendente de outro usuário, há abaixo da imagem do avatar, os botões de aceitar ou recusar amizade, que será uma escolha do usuário.

Figura 5.6 – SyncFriends: Solicitação de Amizade

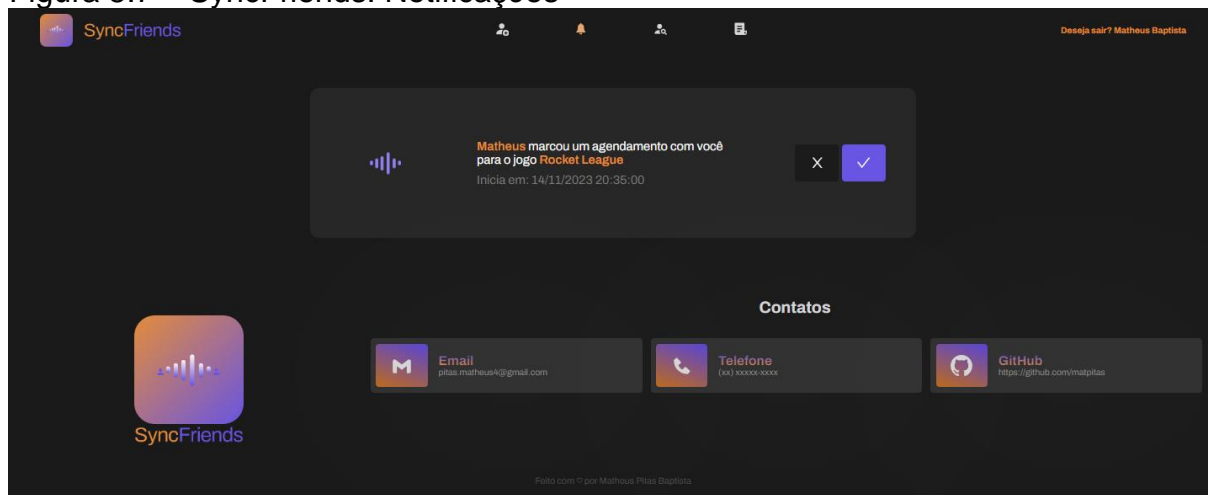


Fonte: Elaborado pelo autor, 2023.

5.1.7 Notificações

A tela de notificações demonstrada na Figura 5.7, tem como objetivo mostrar a solicitação de agendamento feita por um amigo do usuário, e nessa notificação fica dois botões, um para aceitar e outro para negar a participação no agendamento.

Figura 5.7 – SyncFriends: Notificações



Fonte: Elaborado pelo autor, 2023.

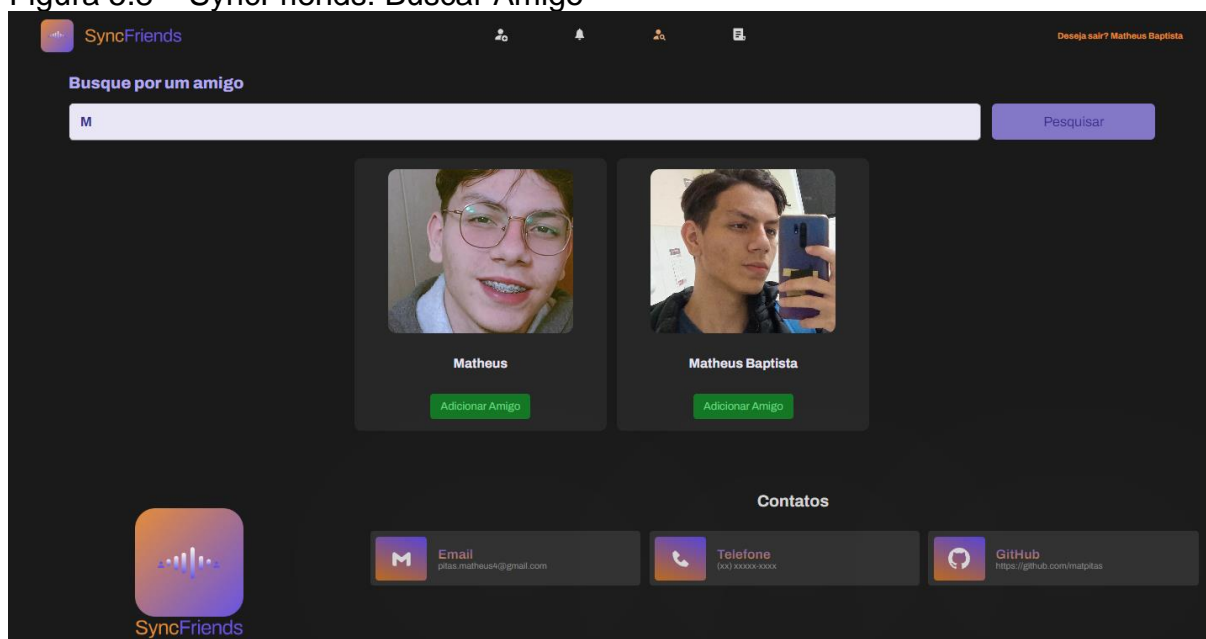




5.1.8 Buscar Amigos

Na Figura 5.8, mostra como fica a tela de busca de amigos, após o usuário ter feito a consulta no campo de busca, o sistema retorna todos os usuários com aquele começo de texto e assim o usuário pode escolher enviar solicitação ou não para uma das pessoas mostradas.

Figura 5.8 – SyncFriends: Buscar Amigo



Fonte: Elaborado pelo autor, 2023.

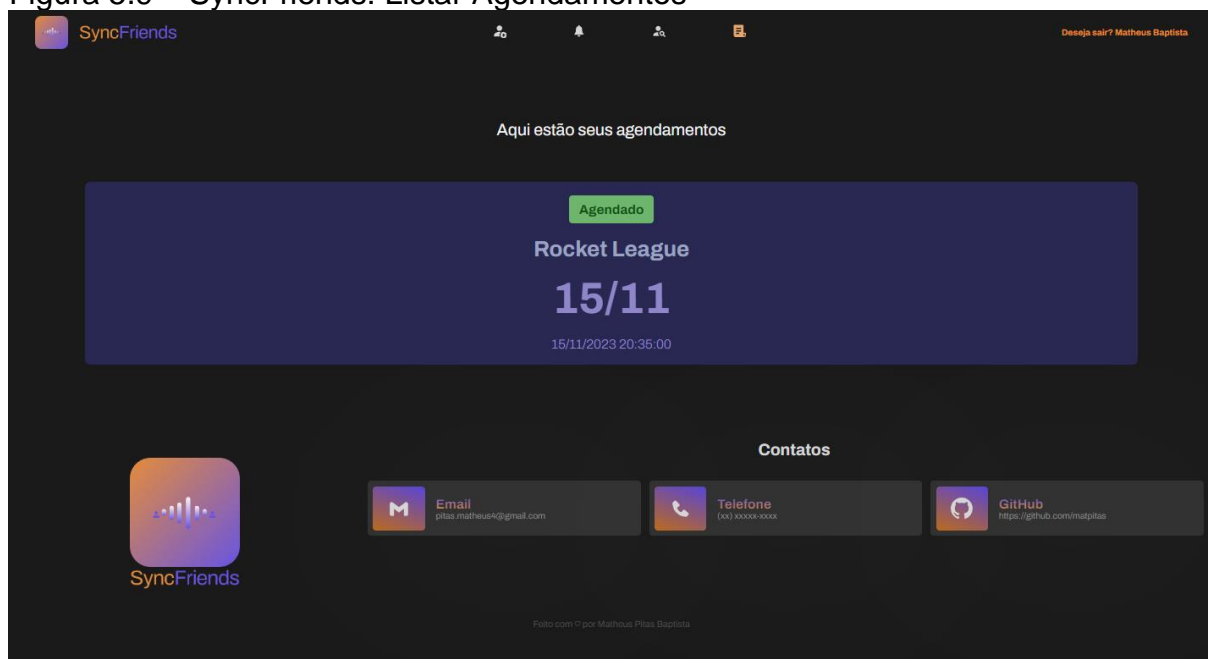
5.1.9 Listar Agendamentos

Como demonstrado abaixo, nessa interface do sistema, o usuário pode ver todos os agendamentos que estão relacionados a ele e que ainda não finalizaram.





Figura 5.9 – SyncFriends: Listar Agendamentos



Fonte: Elaborado pelo autor, 2023.

5.1.10 Criar Agendamento

Ainda na tela do painel de controle, tem a opção de agendar a partir do botão em laranja “Agendar”, quando clicado o sistema redireciona o usuário para o preenchimento das informações do agendamento. Após o preenchimento de todas as informações, o sistema faz a validação dos dados e retorna ao usuário uma resposta de criação ou de erro.





Figura 5.10 – SyncFriends: Criar Agendamento

Conecte AMIGOS **Agende Horários**

Agendamento

Jogo
Escolha Uma Opção

Horario de Inicio **Horario de Termino**
dd/mm/aaaa --:-- dd/mm/aaaa --:--

Amigos

☐ Pibe

☐ Mathous Baptista

Agendar

Fonte: Elaborado pelo autor, 2023.





6 CONSIDERAÇÕES FINAIS

Em conclusão, o desenvolvimento de uma aplicação de agendamentos voltada para o nicho de jogos eletrônicos marca um novo segmento de mercado, que pode ser explorado com maior estudo e tempo. A capacidade de unir os controles de horário com o lazer de jogar com os amigos proporciona aos usuários um melhor gerenciamento da vida pessoal e da rotina, o uso da ferramenta traz uma visão simplificada ao usuário de quanto tempo os jogos têm influenciado no dia a dia.

A aplicação ao oferecer aos usuários de jogos virtuais um fluxo de agendamento, contribui para o crescimento pessoal, na perspectiva de entregar ao usuário um aspecto macro de como a organização do seu lazer pode influenciar em atividades cotidianas e na sua saúde mental, para que assim, não sinta que o tempo está sendo distorcido por essas atividades.

Sabe-se que uma aplicação nunca é finalizada completamente, está sempre em constante evolução, porém, para a conclusão desse trabalho, é necessário que haja um ponto de término temporário, e diante dessa perspectiva, há muitas funcionalidades que ainda podem ser implementadas no sistema como a Edição de Perfil, Filtragem de agendamentos, Gráficos Analíticos, Integrações com aplicativos de agendamento e jogos, Painel administrativo, entre outros. Além das funcionalidades, a alocação do sistema para um domínio público será, com certeza, algo essencial para a utilização futura da aplicação.

A utilização de tecnologias modernas como ReactJS e NodeJS para a codificação do projeto ajudaram-no a ser estruturado e com padrão de código. Além do auxílio de bibliotecas como Multer, CookiesJs, Axios, Express e outras, que fizeram o processo de desenvolvimento ser rápido e otimizado. Do ponto de vista de aprendizado, nesse projeto houve a conclusão de uma aplicação funcional em tecnologias que há um recém começo de estudo, mas que não impediu que fosse finalizado.

Em suma, o sistema cumpre os objetivos proposto de agendamento e controle de horário de jogos multijogadores com os amigos, ainda que possa ser implementado melhorias, o fluxo básico de agendamento foi concluído.





REFERÊNCIAS BIBLIOGRÁFICAS

BASSETTE, FERNANDA. Quase 30% dos adolescentes brasileiros fazem uso problemático de videogame, aponta estudo da USP. **Revista Crescer**.2022. Disponível em: <<https://www.ip.usp.br/site/noticia/quase-30-dos-adolescentes-brasileiros-fazem-uso-problematico-de-videogame-aponta-estudo-da-usp/#:~:text=De%20acordo%20com%20Luiza%2C%20o,%2C%20sintomas%20emocionais%2C%20entre%20outros>>. Acesso em: 19 set. 2023.

GOOGLE. **Google Agenda**. 2023. Disponível em: <<https://calendar.google.com/calendar/u/0/r/month/2023/11/1>>. Acesso em: 01 nov. 2023.

IBM. **Diagramas de Caso de Uso**. 05 mar. 2021. Disponível em: <<https://www.ibm.com/docs/pt-br/rsm/7.5.0?topic=diagrams-use-case>>. Acesso em: 01 nov. 2023.

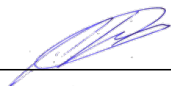
IBM. **Diagramas de Atividades**. 02 mar. 2021. Disponível em: <<https://www.ibm.com/docs/pt-br/rational-soft-arch/9.7.0?topic=diagrams-activity>>. Acesso em: 12 nov. 2023.

KOERICH, BRUNA ROSSI, E MELISSA DE MATTOS PIMENTA. **Temporalidades juvenis e impactos do contexto pandêmico**. 2023. Disponível em: <<https://www.scielo.br/j/civitas/a/ncFzbbrBt3pCzLxvnTkYdJN/?lang=pt>>. Acesso em 19 set. 2023.

OLIVEIRA, DANIELLE. **MER e DER: Definições, Banco de Dados e Exemplos**.18 set. 2023. Disponível em: <<https://www.alura.com.br/artigos/mer-e-der-funcoes>>. Acesso em: 01 nov. 2023.



Página de assinaturas



Anderson Pazin
264.548.978-85
Signatário



Julio Lieira
080.769.668-41
Signatário











Luiz Silva
265.257.308-05
Signatário



Thiago Patricio
364.421.478-60
Signatário

HISTÓRICO

19 dez 2023 11:00:53		Anderson Pazin criou este documento. (E-mail: anderson.pazin@fatec.sp.gov.br, CPF: 264.548.978-85)
19 dez 2023 11:00:53		Anderson Pazin (E-mail: anderson.pazin@fatec.sp.gov.br, CPF: 264.548.978-85) visualizou este documento por meio do IP 177.95.133.194 localizado em Guaratingueta - Sao Paulo - Brazil
19 dez 2023 11:00:55		Anderson Pazin (E-mail: anderson.pazin@fatec.sp.gov.br, CPF: 264.548.978-85) assinou este documento por meio do IP 177.95.133.194 localizado em Guaratingueta - Sao Paulo - Brazil
20 dez 2023 18:12:56		Thiago Seti Patricio (E-mail: thiago.patricio@fatec.sp.gov.br, CPF: 364.421.478-60) visualizou este documento por meio do IP 177.21.143.255 localizado em Araçatuba - Sao Paulo - Brazil
20 dez 2023 18:13:02		Thiago Seti Patricio (E-mail: thiago.patricio@fatec.sp.gov.br, CPF: 364.421.478-60) assinou este documento por meio do IP 177.21.143.255 localizado em Araçatuba - Sao Paulo - Brazil
20 dez 2023 15:40:19		Julio Fernando Lieira (E-mail: julio.lieira3@fatec.sp.gov.br, CPF: 080.769.668-41) visualizou este documento por meio do IP 189.126.178.186 localizado em Lins - Sao Paulo - Brazil
20 dez 2023 15:40:27		Julio Fernando Lieira (E-mail: julio.lieira3@fatec.sp.gov.br, CPF: 080.769.668-41) assinou este documento por meio do IP 189.126.178.186 localizado em Lins - Sao Paulo - Brazil
20 dez 2023 15:55:43		Luiz Fernando De Oliveira Silva (E-mail: luiz.silva169@fatec.sp.gov.br, CPF: 265.257.308-05) visualizou este documento por meio do IP 201.182.123.228 localizado em Lins - Sao Paulo - Brazil



20 dez 2023

15:55:59



Luiz Fernando De Oliveira Silva (E-mail: luiz.silva169@fatec.sp.gov.br, CPF: 265.257.308-05) assinou este documento por meio do IP 201.182.123.228 localizado em Lins - Sao Paulo - Brazil

