Logotipo, nome da empresa

Descrição gerada automaticamente

**FATEC COTIA**

**Desenvolvimento de Software Multiplataforma**

LÁZARO SANTOS

LEANDRO CARDOSO

CELSO SEBASTIÃO

JHONATHAM PEDROSO

**IES011 - Projeto Integrador I - Atividade 1**

Sistema Inteligente de Recomendação para Bares e Restaurantes - ABOA

**COTIA – SP**

**OUTUBRO / 2024**

Lista de figuras

|  |  |
| --- | --- |
| Figura 01 - Cadastro de Clientes e Restaurantes | 19 |
| Figura 02 - Autenticação Segura | 21 |
| Figura 03 - Busca por Restaurantes e Pratos | 23 |
| Figura 04 - Listagem de Restaurantes Baseados na Localização | 25 |
| Figura 05 - Modificação de Cardápios | 27 |
| Figura 06 - Exibição de Informações do Restaurante | 29 |
| Figura 07 - Realização de Pedidos de Pratos | 31 |
| Figura 08 - Exibição do Status do Pedido em Tempo Real | 33 |
| Figura 09 - Exibição de Restaurantes com Desconto e Sugestões Baseadas nas Preferências do Usuário | 35 |
| Figura 10 - Backlog-1°Sprint - 11 a 22 Set de 2024 | 38 |
| Figura 11 - 1°Sprint - 11 a 22 Set de 2024 | 39 |
| Figura 13 - 2°Sprint - 23 a 30 Set de 2024 | 40 |
| Figura 14 - 2°Sprint - 23 a 30 Set de 2024-Histórias e tarefas | 40 |
| Figura 15 - Backlog-3°Sprint - 06 a 12 Oct de 2024 | 40 |
| Figura 16 - 3°Sprint - 06 a 12 Oct de 2024 | 41 |
| Figura 17 - Timeline | 42 |

Sumário

[1. INTRODUÇÂO 3](#_Toc179706129)

[1.1. Problemática 3](#_Toc179706130)

[1.2. Questões a Serem Respondidas 4](#_Toc179706131)

[1.3. Para concluir este tópico 5](#_Toc179706132)

[1.4. Proposta 5](#_Toc179706133)

[2. DESAFIOS TÉCNICOS E FORMAS DE RESOLUÇÃO 6](#_Toc179706134)

[2.1. Métodos de Recomendação 6](#_Toc179706135)

[2.2. Google Places API 7](#_Toc179706136)

[2.3. Monetização: Destaque Pago para Restaurantes 8](#_Toc179706137)

[2.4. Métricas de Recomendação aos Restaurantes 9](#_Toc179706138)

[2.5. Evitar Superlotação dos Estabelecimentos 9](#_Toc179706139)

[3. TÉCNICAS DE ELICITAÇÃO DE REQUISITOS UTILIZADAS E SEUS RESULTADOS 10](#_Toc179706140)

[4. REQUISITOS DE USUÁRIOS 11](#_Toc179706141)

[5. REQUISITOS FUNCIONAIS E NÃO-FUNCIONAIS (MoSCoW) 15](#_Toc179706142)

[5.1. Requisitos Funcionais (RF) 15](#_Toc179706143)

[5.2. Requisitos Não-Funcionais (RNF) 16](#_Toc179706144)

[6. FERRAMENTAS E METODOLOGIAS UTILIZADAS 35](#_Toc179706145)

[6.1. Cerimônias/Cadências 37](#_Toc179706146)

[6.2. Prints 38](#_Toc179706147)

[6.3. Ferramentas utilizadas 42](#_Toc179706148)

[6.4. Justificativa de usar tais ferramentas 42](#_Toc179706149)

[7. CONCLUSÃO 44](#_Toc179706150)

# Introdução

Um aplicativo que transforma a experiência gastronômica dos utilizadores, proporcionando uma plataforma de pesquisa personalizada para bares e restaurantes. O sistema, ao utilizar informações de localização, preferências pessoais, condições meteorológicas e horários de funcionamento em tempo real, possibilita uma navegação intuitiva e eficaz, garantindo recomendações precisas, pedidos simplificados e integração de informações atualizadas, proporcionando uma experiência única e cativante para cada cliente.

## Problemática

Atualmente, muitos usuários de aplicativos de busca enfrentam dificuldades em encontrar estabelecimentos que atendam às suas necessidades específicas, levando em consideração não apenas a localização, mas também a disponibilidade, os horários de funcionamento e as condições ambientais. A experiência de busca muitas vezes se torna frustrante e ineficiente, resultando em escolhas subótimas.

Os usuários desejam encontrar estabelecimentos que estejam abertos e disponíveis conforme suas preferências. No entanto, a busca tradicional muitas vezes ignora fatores críticos, como:

* **Localização:** Embora a maioria dos aplicativos permita buscas baseadas em localização, muitos não oferecem resultados relevantes se o usuário não estiver na área específica. Isso limita a capacidade de descobrir novos estabelecimentos ou serviços próximos, especialmente em áreas menos conhecidas ou em deslocamentos.
* **Horário de Funcionamento:** Os horários de funcionamento dos estabelecimentos nem sempre são atualizados em tempo real. Isso pode resultar em frustrações significativas quando o usuário tenta visitar um local e encontra as portas fechadas. Além disso, variações em horários durante feriados ou eventos especiais são frequentemente negligenciadas.
* **Condições Climáticas:** O clima pode influenciar consideravelmente a escolha do estabelecimento. Por exemplo, em dias chuvosos, um usuário pode preferir um café aconchegante, enquanto em dias ensolarados, pode buscar um restaurante com área externa. O impacto das condições climáticas na experiência do cliente é frequentemente ignorado pelos sistemas de busca.
* **Fatores Adicionais:** Além dos fatores mencionados, existem diversos elementos que poderiam influenciar a escolha, como eventos locais (festivais, shows), avaliações de usuários, promoções específicas, ou mesmo a presença de amigos e familiares na área. Essas variáveis poderiam ser integradas para oferecer uma busca mais personalizada.

## Questões a Serem Respondidas

**Integração de Dados**

Como o software pode integrar dados de localização, horários de funcionamento e informações climáticas para oferecer recomendações mais precisas? Que APIs ou fontes de dados podem ser utilizadas para garantir que as informações sejam atualizadas e relevantes?

**Atualização de Informações**

Quais são os métodos para garantir que as informações de horário e clima sejam constantemente atualizadas e confiáveis? A implementação de uma verificação em tempo real ou parcerias com serviços de meteorologia podem ser soluções viáveis.

**Personalização da Busca**

De que maneira a personalização da busca pode ser aprimorada para incluir preferências do usuário e outros fatores contextuais? O que poderia ser feito para coletar e analisar os dados de comportamento do usuário de maneira ética e eficaz?

## Para concluir este tópico

A criação de um software que leve em conta a localização, horário de funcionamento e condições climáticas não só melhoraria significativamente a experiência do usuário, mas também aumentaria a relevância e a precisão das recomendações. Com essa abordagem, a busca por estabelecimentos se tornaria mais eficiente e adaptada às necessidades reais dos usuários, proporcionando uma experiência mais satisfatória e engajadora.

## Proposta

Criar uma experiência única para os clientes, facilitando o acesso a bares e restaurantes, permitindo que descubram novos lugares, façam pedidos com simplicidade e se sintam em casa em cada interação com o aplicativo. O sistema será um companheiro digital, guiando o usuário em sua jornada gastronômica, desde a descoberta do local até o pagamento.

**Funcionalidades**

* **Sugestões baseadas em preferências e humor:**  
  Recomendação de bares e restaurantes de acordo com clima, localização e preferências pessoais. Exemplo: em um dia frio, sugerir locais que ofereçam pratos quentes, como sopas ou fondues.
* **Exploração por categorias:**  
  Permitir ao usuário explorar por tipo de comida (italiana, japonesa, vegana) ou por clima do local (aconchegante, moderno, descontraído).
* **Exploração por proximidade:**  
  Mostrar estabelecimentos próximos ao usuário.
* **Recomendações da comunidade:**  
  Mostrar avaliações e recomendações de outros usuários que visitaram os locais.

**Benefícios para o Cliente**

* **Facilidade e rapidez:**  
  Simplicidade em cada etapa, do pedido ao pagamento.
* **Conexão personalizada:**O app "entende" as preferências do cliente, oferecendo uma experiência única e personalizada.
* **Recomendações precisas:**  
  O sistema mostra estabelecimentos alinhados ao estilo e humor do cliente.

Este sistema de gestão, focado no cliente, cria uma experiência intuitiva, personalizada e acolhedora para frequentadores de bares e restaurantes. Com recomendações inteligentes, um processo de pedido simplificado e uma navegação fluida, o usuário se sentirá valorizado. A integração de preferências pessoais, pagamento facilitado e recompensas transforma a jornada gastronômica, fazendo com que o app se torne um verdadeiro parceiro no dia a dia do cliente.

# Desafios Técnicos e Formas de Resolução

Este documento aborda os desafios técnicos na implementação de um sistema de recomendação de restaurantes, as formas de monetização através de destaque pago, e estratégias para evitar a superlotação dos estabelecimentos.

## Métodos de Recomendação

Para garantir a relevância das recomendações de restaurantes aos usuários, é necessário desenvolver algoritmos que considerem diversos fatores, como preferências pessoais, dados históricos e contexto atual do usuário. Os métodos de recomendação podem ser divididos em:

* **Filtragem Colaborativa:**
* **Baseada em Usuário**: Encontra similaridades entre usuários e recomenda restaurantes que outros com gostos semelhantes apreciaram.
* **Baseada em Item**: Recomenda restaurantes similares àqueles que o usuário já gostou.
* **Filtragem Baseada em Conteúdo:**

Utiliza características do restaurante, como tipo de cozinha, localização e faixa de preço, para oferecer recomendações.

* **Sistemas Híbridos:**

Combina os métodos anteriores para melhorar a precisão das recomendações, mitigando as limitações individuais de cada um.

* **Recomendações Contextuais:**

Considera fatores como horário, localização e preferências, adaptando as sugestões conforme o contexto atual do usuário.

* **Aprendizado de Máquina:** Algoritmos de machine learning são treinados com dados históricos para prever quais restaurantes podem agradar novos usuários.

**Solução Proposta**

Implementar um sistema híbrido que combine filtragem colaborativa, baseada em conteúdo e aprendizado de máquina, utilizando técnicas de análise de sentimento para aumentar a precisão das recomendações e garantir que os dados contextuais sejam considerados.

## Google Places API

A Google Places API oferece funcionalidades que podem enriquecer o sistema de recomendação com dados de localização e avaliações em tempo real.

* **Busca de Lugares**:

Localiza restaurantes com base na posição do usuário, aplicando filtros como tipo de cozinha e faixa de preço.

* **Detalhes de Lugares**:

Fornece informações detalhadas sobre o restaurante, como horários, avaliações e fotos.

* **Geocodificação**:

Converte endereços em coordenadas geográficas para identificar estabelecimentos próximos.

* **Autocompletar**:

Facilita a busca por restaurantes, sugerindo opções enquanto o usuário digita.

* **Avaliações e Classificações**:

Enriquece as recomendações com base na percepção dos clientes.

**Solução Proposta**

Integrar a Google Places API para complementar os dados de recomendação com informações de localização, horários e avaliações, proporcionando ao usuário uma experiência mais rica e contextualizada.

## Monetização: Destaque Pago para Restaurantes

Para que os restaurantes possam se destacar no sistema de recomendações, deve-se implementar um modelo de monetização que seja viável tanto para os estabelecimentos quanto para o sistema.

* **Modelo de Pagamento por Clique (PPC)**: Os restaurantes pagam uma taxa sempre que um cliente interage com o anúncio.
* **Assinatura Mensal**: Um modelo simples onde o estabelecimento paga uma taxa fixa para aparecer em destaque.
* **Leilão de Posição**: Estabelecimentos competem em um leilão para garantir destaque nas recomendações.
* **Pay-per-Recommendation**: Pagamento apenas quando o restaurante é recomendado diretamente ao usuário.

**Solução Proposta**

Propor um modelo híbrido que combine PPC e assinatura mensal, oferecendo a opção de leilão de posição para restaurantes que desejam destaque em períodos de alta demanda.

## Métricas de Recomendação aos Restaurantes

Os restaurantes precisam de visibilidade sobre o desempenho de suas recomendações pagas, para que possam ajustar suas estratégias.

* **Relatórios Periódicos**: Enviar dados semanais ou mensais sobre número de recomendações, cliques, impressões e comparações com concorrentes.
* **Dashboard em Tempo Real**: Oferecer um painel onde os restaurantes podem acompanhar o desempenho em tempo real.
* **Notificações de Desempenho**: Enviar notificações quando o estabelecimento atinge marcos importantes.

**Solução Proposta**

Desenvolver um sistema de relatórios periódicos e um dashboard com métricas em tempo real, possibilitando que os restaurantes ajustem suas campanhas conforme necessário.

## Evitar Superlotação dos Estabelecimentos

Recomendar restaurantes de forma controlada, levando em consideração a capacidade de cada estabelecimento e sua taxa de ocupação em tempo real.

* **Capacidade do Estabelecimento**: Solicitar essa informação diretamente no cadastro do restaurante ou estimar com base no tipo de estabelecimento e localização.
* **Monitorar Taxa de Ocupação**: Acompanhar a ocupação em tempo real e ajustar as recomendações conforme o local se aproxima da capacidade máxima.
* **Limites de Recomendações**: Definir um limite de recomendações que o sistema pode fazer em um determinado período.
* **Segmentação de Recomendações**: Diversificar as recomendações com base em perfis e horários, evitando enviar muitos clientes a um mesmo restaurante simultaneamente.

**Solução Proposta**

Implementar algoritmos que monitorem a capacidade e a ocupação dos estabelecimentos em tempo real, ajustando automaticamente o número de recomendações e diversificando as sugestões conforme necessário.

* 1. **Para concluir este tópico**

Os desafios técnicos de desenvolver um sistema de recomendação eficaz e sustentável incluem a implementação de algoritmos híbridos de recomendação, a integração com APIs de localização e a criação de um modelo de monetização viável. A comunicação eficiente das métricas de desempenho e a prevenção da superlotação de estabelecimentos são essenciais para garantir uma boa experiência para os usuários e promover o crescimento do sistema.

# TÉCNICAS DE ELICITAÇÃO DE REQUISITOS UTILIZADAS E SEUS RESULTADOS

No desenvolvimento do **Sistema Inteligente de Recomendação para Bares e Restaurantes**, utilizamos várias técnicas de **Elicitação de Requisitos** para garantir que as necessidades dos usuários e os objetivos do projeto fossem claramente compreendidos e documentados. Abaixo, estão as técnicas utilizadas e seus respectivos resultados:

**Brainstorming**

Uma das principais técnicas de elicitação foi o **Brainstorming**, que envolveu a reunião de todos os integrantes do time e potenciais usuários do sistema. Durante essas sessões, todos os participantes puderam fornecer ideias e sugestões livremente, promovendo uma discussão aberta. As ideias foram registradas, agrupadas por similaridade e posteriormente priorizadas de acordo com sua relevância e viabilidade técnica.

**Reuniões de Alinhamento com Stakeholders**

Além do Brainstorming, realizamos **reuniões formais com membros da equipe**, onde discutimos as expectativas dos usuários em relação ao sistema. Durante essas reuniões, foram levantadas questões relacionadas à usabilidade, tipos de recomendação, e como o sistema poderia integrar APIs externas para fornecer dados atualizados.

**Análise de Documentos**

Para complementar o levantamento dos requisitos, utilizamos a **análise de documentos** existentes, como manuais de APIs e guias de implementação de sistemas similares, para garantir que os requisitos fossem tecnicamente viáveis. A API do Google Places, por exemplo, foi estudada para compreender como ela poderia ser integrada ao sistema de recomendação.

Todas as atas de reuniões, documentos de análise e demais arquivos relacionados à Elicitação de Requisitos estão armazenados no repositório do projeto no GitHub, garantindo o acesso transparente e organizado de todos os integrantes da equipe e stakeholders.

# REQUISITOS DE USUÁRIOS

Os requisitos funcionais do sistema são descritos a partir da perspectiva dos usuários, utilizando o formato de user stories (histórias de usuário). Essa abordagem facilita a compreensão das necessidades e expectativas dos usuários em relação ao sistema, descrevendo as funcionalidades desejadas de maneira clara e focada no valor entregue. As user stories são estruturadas para representar o que o usuário espera realizar com o sistema, permitindo uma comunicação eficiente entre desenvolvedores, clientes e demais partes interessadas.

Cada história de usuário segue o formato padrão:

* Como [tipo de usuário], eu quero [realizar uma ação], para que [alcançar um objetivo].

Esta forma de descrição ajuda a entender o contexto e o propósito das funcionalidades do sistema, garantindo que todas as partes envolvidas tenham uma visão clara sobre os requisitos a serem atendidos

|  |  |  |
| --- | --- | --- |
| Identificador | Descrição | Prioridade |
| RF01 | Como administrador de um restaurante, eu quero realizar o cadastro do meu estabelecimento no sistema, para que eu possa gerenciar o cardápio, estoque e pedidos de maneira centralizada. | Must |
| RF01 | Como administrador de um restaurante, eu quero fornecer informações detalhadas sobre meu restaurante, como nome, endereço, tipo de comida e horários de funcionamento, para que os clientes possam encontrar e saber mais sobre meu estabelecimento. | Must |
| RF01 | Como cliente, eu quero me cadastrar no sistema com meus dados pessoais, para que eu possa acessar o cardápio e fazer pedidos diretamente pelo aplicativo. | Must |
| RF01 | Como cliente, eu quero salvar minhas preferências (como tipos de comida e estabelecimentos favoritos) no cadastro, para que o sistema possa me oferecer recomendações personalizadas. | Must |
| RF03 | *Como* administrador de um restaurante, *eu quero* que o sistema valide os dados inseridos no cadastro (como e-mail e telefone), *para que* eu tenha certeza de que as informações fornecidas estão corretas e atualizadas. | Must |
| RF03 | Como cliente, eu quero que meus dados pessoais sejam protegidos por uma senha segura e criptografada durante o cadastro, para que minhas informações estejam seguras e privadas. | Must |
| RF04 | Como cliente, eu quero buscar por restaurantes e pratos no sistema, para que eu encontre opções que atendam às minhas preferências alimentares e localização. | Must |
| RF05 | Como cliente, eu quero visualizar uma lista de restaurantes próximos com base na minha localização, para que eu possa escolher um local conveniente para fazer meu pedido. | Must |
| RF08 | *Como* administrador do restaurante, *eu quero* modificar o cardápio do meu estabelecimento no sistema, *para que* eu possa manter as informações de pratos atualizadas e refletir a disponibilidade em tempo real. | Must |
| RF09 | Como cliente, eu quero visualizar o cardápio completo de um restaurante, para que eu possa fazer meu pedido com base nas opções disponíveis. | Must |
| RF10 | Como cliente, eu quero ver as informaçõesdetalhadas de um restaurante, como formas de pagamento, horário de funcionamento, avaliação, categoria e distância, para que eu possa tomar uma decisão informada antes de fazer um pedido. | Must |
| RF13 | Como cliente, eu quero realizar pedidos de pratos diretamente pelo sistema, para que eu possa garantir que meu pedido seja enviado ao restaurante de forma rápida e eficiente. | Must |
| RF14 | Como cliente, eu quero acompanhar o status do meu pedido em tempo real (realizado, sendo preparado, pronto para retirada), para que eu saiba exatamente quando meu pedido estará pronto. | Must |
| RF15 | Como cliente, eu quero realizar o pagamento diretamente pelo aplicativo, para que eu possa concluir minha compra de forma rápida e segura, sem precisar manusear dinheiro ou cartões fisicamente. | Must |
| RF12 | Como cliente, eu quero receber sugestões de restaurantes com descontos e baseadas nas minhas preferências, para que eu possa descobrir novos lugares e aproveitar promoções especiais. | Must |

# **REQUISITOS FUNCIONAIS E NÃO-FUNCIONAIS** (MoSCoW)

Este relatório tem como objetivo documentar os requisitos funcionais do "Sistema de Gerenciamento para Restaurantes e Bares", utilizando a técnica de priorização MoSCoW para identificar e organizar as funcionalidades mais importantes para o desenvolvimento do sistema. Além disso, serão apresentados diagramas de casos de uso que ilustram as interações entre os atores e os principais requisitos funcionais classificados como essenciais ("Must Have").

## Requisitos Funcionais (RF)

* Must Have:
  + - RF001: O sistema deve permitir o cadastro de clientes e restaurantes.
    - RF003: O sistema deve permitir a autenticação segura (ex.: login, recuperação de senha).
    - RF004: O sistema deve permitir que o usuário busque por restaurantes e pratos.
    - RF005: O sistema deve listar restaurantes baseados na localização inserida.
    - RF008: O sistema deve permitir que os restaurantes modifiquem seus cardápios.
    - RF009: O sistema deve listar o cardápio completo de cada restaurante.
    - RF010: O sistema deve exibir informações do restaurante: formas de pagamento, horário de funcionamento, avaliação, categoria, distância.
    - RF013: O usuário deve ser capaz de realizar pedidos de pratos no restaurante.
    - RF014: O sistema deve exibir o status do pedido em tempo real: realizado, sendo preparado, pronto para retirada.
    - RF012: O sistema deve exibir restaurantes com desconto e sugestões baseadas nas preferências do usuário.
* Should Have:
  + - RF002: O usuário deve ser capaz de editar seus dados pessoais e endereço.
    - RF006: O sistema deve fornecer filtros avançados para melhorar a busca (ex.: por tipo de cozinha, distância, preço).
    - RF011: O sistema deve identificar a disponibilidade do restaurante.
    - RF016: O usuário deve ser capaz de avaliar os restaurantes.
    - RF017: O usuário deve ser capaz de favoritar restaurantes para acessos futuros.
* Could Have:
  + - RF007: O sistema deve permitir que o usuário limpe o histórico de busca.
    - RF018: O sistema deve exibir o histórico completo dos pedidos do usuário.
    - RF015: O sistema deve permitir que o usuário realize pagamentos pelo aplicativo.
* Won’t Have:
  + - RF020: O sistema deve oferecer um programa de fidelidade com acúmulo de pontos e resgate de prêmios.
    - RF021: O sistema deve enviar notificações push para promoções, status de pedido e novidades.
    - RF024: O sistema deve sugerir pratos/restaurantes com base no humor ou estado emocional do usuário.
    - RF025: O sistema deve integrar programas de recompensa de cartões de crédito (descontos/cashback).
    - RF026: O sistema deve funcionar no modo offline, permitindo que o usuário navegue sem conexão.
    - RF027: O sistema deve permitir que o cliente crie cardápios personalizados com pratos favoritos de diferentes restaurantes.

## Requisitos Não-Funcionais (RNF)

* Must Have:
  + - RNF001: O sistema deve estar disponível para as plataformas: Web, Android, iOS.
    - RNF002: O sistema deve se conectar com o sistema de localização do aparelho de usuário.
    - RNF004: O sistema deve ser responsável pela segurança e privacidade dos dados dos usuários.
    - RNF005: As operações críticas, como busca e carregamento de cardápios, devem ser processadas em tempo real, com respostas rápidas.
    - RNF006: O sistema deve fornecer feedback imediato ao usuário sobre o sucesso ou falha da transação.
* Should Have:
  + - RNF003: O sistema deve garantir alta disponibilidade.
    - RNF009: O sistema web deve ser compatível com os principais navegadores (Chrome, Firefox, Safari, Edge).
* Could Have:
  + - RNF007: O sistema deve ser fácil de manter e atualizar, permitindo a adição de novos recursos e correção de bugs sem afetar a experiência do usuário.
    - RNF008: O código deve seguir boas práticas de desenvolvimento, como modularidade e documentação clara.
    1. **RF001: Cadastro de Clientes e Restaurantes**

Descrição: O sistema permitirá que os usuários se cadastrem como clientes ou estabelecimentos, preenchendo os dados necessários.

Atores: Clientes e Estabelecimentos.

Pré-condições: O usuário deve ter acesso ao sistema.

Pós-condições: O usuário será cadastrado no sistema com sucesso.

Entradas:

Nome completo (Cliente ou Estabelecimento), E-mail, Senha, Confirmação de senha, Tipo de perfil (Cliente ou Estabelecimento), Telefone, Endereço (para estabelecimentos)

Fluxo Principal:

1. O usuário acessa o sistema.

2. O sistema apresenta uma tela com os campos necessários para o cadastro.

3. O usuário preenche os seguintes dados:

* Nome completo
* E-mail
* Senha e confirmação de senha
* Tipo de perfil (Cliente ou Estabelecimento)
* Telefone (opcional para cliente, obrigatório para estabelecimento)
* Endereço (obrigatório para estabelecimento)

4. O sistema valida os dados preenchidos.

5. Caso a validação seja bem-sucedida, o sistema exibe a mensagem “Cadastro efetuado com sucesso”.

Subfluxos:

[SB001] Campos Obrigatórios não preenchidos:

1. No passo 3 do fluxo principal, se o usuário deixar de preencher algum campo obrigatório, o sistema exibe a mensagem “Campos obrigatórios não preenchidos”.

2. O usuário permanece na mesma tela de cadastro para corrigir as informações.

3. O subfluxo termina, e o sistema retorna ao passo 3 do fluxo principal.

[SB002] Usuário já cadastrado:

1. Se o e-mail do usuário já estiver cadastrado, o sistema exibe a mensagem “Usuário já cadastrado”.

2. O subfluxo termina, e o sistema retorna ao passo 3 do fluxo principal.

Diagrama, Esquemático

Descrição gerada automaticamente

Figura 01 - Cadastro de Clientes e Restaurantes

* + 1. **RF003:** **Autenticação Segura**

Descrição: O sistema deve permitir a autenticação segura dos usuários, incluindo funcionalidades de login e recuperação de senha.

Atores: Clientes e Estabelecimentos.

Pré-condições: O usuário deve ter uma conta previamente cadastrada no sistema.

Pós-condições: O usuário será autenticado no sistema e terá acesso às funcionalidades correspondentes ao seu perfil.

Entradas:

E-mail, Senha

Fluxo Principal de Login:

1. O usuário acessa a página de login do sistema.

2. O sistema apresenta uma tela com campos para preenchimento:

* E-mail
* Senha

3. O usuário preenche os campos com suas credenciais.

4. O sistema valida as credenciais informadas.

5. Se as credenciais estiverem corretas, o sistema redireciona o usuário para a página inicial.

Subfluxos:

[SB001] Credenciais inválidas:

1. Se o usuário informar um e-mail ou senha incorretos, o sistema exibe a mensagem “Credenciais inválidas”.

2. O usuário permanece na tela de login para tentar novamente.

3. O subfluxo termina e retorna ao passo 2 do fluxo principal.

[SB002] Recuperação de senha:

1. O usuário clica na opção “Esqueci minha senha”.

2. O sistema solicita o e-mail associado à conta.

3. O usuário informa o e-mail e solicita a recuperação.

4. O sistema envia um e-mail de recuperação com instruções para redefinir a senha.

5. O sistema exibe a mensagem “Instruções de recuperação enviadas para o seu e-mail”.

Fluxo Alternativo para Recuperação de Senha:

1. Se o e-mail informado não estiver cadastrado, o sistema exibe a mensagem “E-mail não cadastrado”.

2. O usuário retorna à tela de recuperação de senha para tentar novamente ou voltar à tela de login.

Diagrama

Descrição gerada automaticamente

Figura 02 - Autenticação Segura

* + 1. **RF004: Busca por Restaurantes e Pratos**

Descrição: O sistema deve permitir que o usuário busque por restaurantes e pratos disponíveis, facilitando a navegação e a escolha de opções.

Atores: Clientes.

Pré-condições: O usuário deve estar autenticado no sistema.

Pós-condições: O sistema exibirá os resultados da busca conforme os critérios especificados pelo usuário.

Entradas:

Termo de busca (nome do restaurante ou prato), Filtros opcionais (categoria de prato, distância, preço, avaliações)

Fluxo Principal:

1. O usuário acessa a página de busca no sistema.

2. O sistema apresenta uma barra de pesquisa e filtros disponíveis.

3. O usuário insere o termo de busca e, se desejar, aplica filtros adicionais.

4. O usuário clica no botão de busca.

5. O sistema processa a solicitação e pesquisa na base de dados.

6. O sistema exibe os resultados correspondentes à busca, incluindo:

* Nome do restaurante
* Pratos disponíveis
* Avaliações
* Distância (se aplicável)

7. O usuário pode clicar em um restaurante ou prato para obter mais informações.

Subfluxos:

[SB001] Nenhum resultado encontrado:

1. Se a busca não retornar resultados, o sistema exibe a mensagem “Nenhum resultado encontrado”.

2. O usuário pode optar por modificar o termo de busca ou aplicar filtros diferentes.

3. O subfluxo termina e retorna ao passo 3 do fluxo principal.

[SB002] Busca inválida:

1. Se o usuário inserir um termo de busca inválido (por exemplo, caracteres especiais ou vazio), o sistema exibe a mensagem “Termo de busca inválido”.

2. O usuário permanece na tela de busca para corrigir o termo.

3. O subfluxo termina e retorna ao passo 2 do fluxo principal.

Diagrama

Descrição gerada automaticamente

Figura 03 - Busca por Restaurantes e Pratos

* + 1. **RF005: Listagem de Restaurantes Baseados na Localização**

Descrição: O sistema deve listar restaurantes disponíveis com base na localização inserida pelo usuário, permitindo uma navegação mais eficiente.

Atores: Clientes.

Pré-condições: O usuário deve estar autenticado no sistema.

Pós-condições: O sistema exibirá uma lista de restaurantes disponíveis na localização especificada pelo usuário.

Entradas:

Localização (endereço, cidade ou coordenadas geográficas), Filtros opcionais (categoria de cozinha, faixa de preço)

Fluxo Principal:

1. O usuário acessa a página de listagem de restaurantes.

2. O sistema apresenta um campo para inserção da localização e opções de filtro.

3. O usuário insere a localização desejada e, se desejar, aplica filtros adicionais.

4. O usuário clica no botão de busca.

5. O sistema processa a solicitação e pesquisa na base de dados.

6. O sistema exibe a lista de restaurantes correspondentes à localização, incluindo:

* Nome do restaurante
* Endereço
* Distância da localização inserida
* Categoria de cozinha
* Avaliações

7. O usuário pode clicar em um restaurante para obter mais informações.

Subfluxos:

[SB001] Nenhum restaurante encontrado:

1. Se a pesquisa não retornar restaurantes, o sistema exibe a mensagem “Nenhum restaurante encontrado na localização especificada”.

2. O usuário pode optar por modificar a localização ou ajustar os filtros.

3. O subfluxo termina e retorna ao passo 3 do fluxo principal.

[SB002] Localização inválida:

1. Se o usuário inserir uma localização inválida (por exemplo, caracteres especiais ou vazios), o sistema exibe a mensagem “Localização inválida”.

2. O usuário permanece na tela de listagem para corrigir a localização.

3. O subfluxo termina e retorna ao passo 2 do fluxo principal.

Diagrama

Descrição gerada automaticamente

Figura 04 - Listagem de Restaurantes Baseados na Localização

**1.1.8 RF008: Modificação de Cardápios**

Descrição: O sistema deve permitir que os restaurantes atualizem e modifiquem seus cardápios, garantindo que as informações estejam sempre atualizadas.

Atores: Estabelecimentos (administradores dos restaurantes).

Pré-condições: O usuário deve estar autenticado como estabelecimento no sistema.

Pós-condições: O cardápio do restaurante será atualizado conforme as modificações realizadas.

Entradas:

ID do restaurante, Itens do cardápio a serem adicionados, removidos ou editados (nome, descrição, preço, categoria)

Fluxo Principal:

1. O administrador do restaurante acessa a página de gerenciamento de cardápio no sistema.

2. O sistema apresenta a lista atual dos itens do cardápio.

3. O administrador pode escolher uma das seguintes ações:

* Adicionar um novo item ao cardápio
* Editar um item existente
* Remover um item do cardápio

4. Se o administrador optar por adicionar um item:

* O sistema apresenta um formulário para preenchimento dos dados do novo item (nome, descrição, preço, categoria).
* O administrador preenche os dados e clica em "Adicionar".

5. Se o administrador optar por editar um item:

* O sistema apresenta os detalhes do item selecionado.
* O administrador realiza as modificações necessárias e clica em "Salvar".

6. Se o administrador optar por remover um item:

* O sistema solicita a confirmação da remoção.
* O administrador confirma e o item é removido do cardápio.

7. O sistema exibe a mensagem “Cardápio atualizado com sucesso” após cada ação realizada.

Subfluxos:

[SB001] Campos obrigatórios não preenchidos:

1. Se o administrador tentar adicionar ou editar um item sem preencher os campos obrigatórios, o sistema exibe a mensagem “Campos obrigatórios não preenchidos”.

2. O administrador permanece na tela de modificação para corrigir os dados.

3. O subfluxo termina e retorna ao passo correspondente do fluxo principal.

[SB002] Item não encontrado para edição/remover:

1. Se o administrador tentar editar ou remover um item que não existe, o sistema exibe a mensagem “Item não encontrado”.

2. O subfluxo termina e retorna à lista de itens do cardápio.

Diagrama

Descrição gerada automaticamente

Figura 05 - Modificação de Cardápios

**1.1.10 RF010: Exibição de Informações do Restaurante**

Descrição: O sistema deve exibir informações detalhadas sobre os restaurantes, incluindo formas de pagamento, horário de funcionamento, avaliação, categoria e distância, para auxiliar os usuários na escolha.

Atores: Clientes.

Pré-condições: O usuário deve estar autenticado no sistema e ter realizado uma busca ou seleção de um restaurante.

Pós-condições: O sistema exibirá as informações detalhadas do restaurante selecionado.

Entradas:

ID do restaurante (ou outro identificador correspondente)

Fluxo Principal:

1. O usuário acessa a lista de restaurantes ou realiza uma busca específica.

2. O usuário seleciona um restaurante da lista.

3. O sistema processa a solicitação e recupera as informações do restaurante.

4. O sistema exibe as seguintes informações detalhadas:

* Nome do restaurante
* Formas de pagamento aceitas
* Horário de funcionamento
* Avaliação (média de avaliações de clientes)
* Categoria (tipo de cozinha)
* Distância em relação à localização do usuário

5. O usuário pode optar por fazer um pedido, ver o cardápio ou retornar à lista de restaurantes.

Subfluxos:

[SB001] Restaurante não encontrado:

1. Se o ID do restaurante não corresponder a nenhum registro no sistema, o sistema exibe a mensagem “Restaurante não encontrado”.

2. O subfluxo termina e retorna à lista de restaurantes ou à tela anterior.

[SB002] Erro ao recuperar informações:

1. Se ocorrer um erro durante a recuperação das informações (por exemplo, problemas de conexão), o sistema exibe a mensagem “Erro ao carregar informações do restaurante. Tente novamente mais tarde”.

2. O subfluxo termina e retorna à lista de restaurantes ou à tela anterior.

Diagrama, Esquemático

Descrição gerada automaticamente

Figura 06 - Exibição de Informações do Restaurante

**1.1.13 RF013: Realização de Pedidos de Pratos**

Descrição: O sistema deve permitir que o usuário realize pedidos de pratos diretamente no restaurante selecionado.

Atores: Clientes.

Pré-condições: O usuário deve estar autenticado no sistema e ter selecionado um restaurante.

Pós-condições: O pedido será registrado no sistema e enviado ao restaurante para processamento.

Entradas:

Seleção do prato (ID do prato), Quantidade, Observações/opcionais (por exemplo, sem ingredientes específicos), Informações de pagamento (cartão, voucher, etc.)

Fluxo Principal:

1. O usuário acessa o cardápio do restaurante selecionado.

2. O sistema exibe a lista de pratos disponíveis com preços e descrições.

3. O usuário escolhe um prato, define a quantidade desejada e, se necessário, adiciona observações (opcionais).

4. O usuário clica no botão "Adicionar ao pedido".

5. O sistema apresenta um resumo do pedido (itens, preços, total).

6. O usuário revisa o pedido e escolhe uma forma de pagamento (cartão, voucher, etc.).

7. O usuário confirma o pedido.

8. O sistema registra o pedido e envia a solicitação para o restaurante.

9. O sistema exibe a mensagem “Pedido realizado com sucesso” e um número de pedido para acompanhamento.

Subfluxos:

[SB001] Falta de itens no pedido:

1. Se o usuário tentar confirmar o pedido sem ter adicionado itens, o sistema exibe a mensagem “Nenhum item no pedido”.

2. O subfluxo termina e retorna ao passo 2 do fluxo principal.

[SB002] Falha no pagamento:

1. Se ocorrer um erro no processamento do pagamento (cartão inválido, saldo insuficiente, etc.), o sistema exibe a mensagem “Erro no pagamento. Tente novamente”.

2. O subfluxo termina e retorna ao passo 6 do fluxo principal para o usuário escolher outro método de pagamento.

[SB003] Restaurante indisponível:

1. Se o restaurante não estiver aceitando pedidos no momento (fora do horário de funcionamento, por exemplo), o sistema exibe a mensagem “Restaurante indisponível para pedidos no momento”.

2. O subfluxo termina e retorna à lista de restaurantes ou cardápio.

Diagrama

Descrição gerada automaticamente

Figura 07 - Realização de Pedidos de Pratos

**1.1.14 RF014: Exibição do Status do Pedido em Tempo Real**

Descrição:

O sistema deve permitir que o usuário acompanhe o status do pedido em tempo real, informando as etapas desde a realização até o momento em que está pronto para retirada.

Atores: Clientes e estabelecimentos (restaurantes).

Pré-condições:

• O usuário deve ter realizado um pedido.

• O restaurante deve ter registrado e iniciado o processamento do pedido.

Pós-condições:

O sistema exibirá o status atualizado do pedido em tempo real.

Entradas:

ID do pedido (gerado no momento da realização do pedido)

Fluxo Principal:

1. O usuário acessa a seção “Meus Pedidos” no sistema.

2. O sistema exibe a lista de pedidos em andamento.

3. O usuário seleciona o pedido que deseja acompanhar.

4. O sistema exibe o status atual do pedido, com possíveis estados:

* Realizado
* Sendo preparado
* Pronto para retirada

5. O sistema atualiza o status em tempo real conforme o restaurante muda o estado do pedido.

6. Quando o pedido está pronto, o sistema exibe a mensagem “Pedido pronto para retirada” e, se aplicável, instruções de retirada.

Subfluxos:

[SB001] Nenhum pedido em andamento:

1. Se o usuário não tiver nenhum pedido em andamento, o sistema exibe a mensagem “Nenhum pedido em andamento no momento”.

2. O subfluxo termina e retorna à tela inicial de pedidos.

[SB002] Falha na atualização do status:

1. Se houver um problema técnico na comunicação com o restaurante (conexão ou erro no servidor), o sistema exibe a mensagem “Erro ao atualizar status. Tente novamente”.

2. O subfluxo termina e retorna ao passo 4 do fluxo principal, onde o sistema tenta novamente atualizar o status.

Diagrama

Descrição gerada automaticamente

Figura 08 - Exibição do Status do Pedido em Tempo Real

**1.1.12 RF012: Exibição de Restaurantes com Desconto e Sugestões Baseadas nas Preferências do Usuário**

Descrição: O sistema deve exibir uma lista de restaurantes com descontos e sugestões personalizadas, levando em conta as preferências do usuário (como tipo de comida, restaurantes visitados anteriormente, avaliações, etc.).

Atores: Clientes.

Pré-condições:

• O usuário deve estar autenticado no sistema.

• O sistema deve ter registrado preferências do usuário ou histórico de interações.

Pós-condições:

O sistema exibe uma lista de restaurantes com descontos aplicáveis e sugestões personalizadas para o usuário.

Entradas:

Preferências do usuário (coletadas previamente), Informações sobre descontos em restaurantes (ofertas ativas), Localização atual ou preferida do usuário (opcional)

Fluxo Principal:

1. O usuário acessa a seção de recomendações e descontos no aplicativo.

2. O sistema exibe duas listas:

* Restaurantes com descontos disponíveis.
* Sugestões personalizadas com base nas preferências e histórico do usuário.

3. O usuário seleciona um restaurante de uma das listas.

4. O sistema exibe informações detalhadas sobre o restaurante, como:

* Nome
* Desconto aplicado (se aplicável)
* Formas de pagamento
* Avaliações
* Distância

5. O usuário pode optar por ver o cardápio, fazer um pedido, ou voltar à lista de recomendações/descontos.

Subfluxos:

[SB001] Nenhuma recomendação ou desconto disponível:

1. Se o sistema não encontrar restaurantes com desconto ou não tiver informações suficientes para sugerir restaurantes com base nas preferências do usuário, ele exibe a mensagem “Nenhuma recomendação disponível no momento”.

2. O subfluxo termina e retorna à tela principal do aplicativo.

[SB002] Erro ao carregar recomendações ou descontos:

1. Se ocorrer algum erro ao tentar carregar as listas de restaurantes (como problemas de conexão), o sistema exibe a mensagem “Erro ao carregar recomendações. Tente novamente mais tarde”.

2. O subfluxo termina e retorna ao passo 2 do fluxo principal, onde o sistema tenta novamente.

Diagrama

Descrição gerada automaticamente

Figura 09 - Exibição de Restaurantes com Desconto e Sugestões Baseadas nas Preferências do Usuário

# **FERRAMENTAS E METODOLOGIAS UTILIZADAS**

Scrum - Justificativa para a utilização da metodologia Scrum no projeto

Embora nosso projeto seja acadêmico e envolva um time relativamente pequeno de quatro pessoas, optamos por utilizar a metodologia Scrum devido às suas vantagens em termos de organização, melhoria incremental e colaboração. A adoção de uma metodologia ágil como o Scrum nos proporciona diversos benefícios para o desenvolvimento do nosso sistema de recomendação e gestão de bares e restaurantes:

Organização clara e definição de papéis: O Scrum nos oferece uma estrutura clara, com papéis bem definidos e eventos regulares que facilitam a comunicação entre os membros da equipe. Isso ajuda a manter todos alinhados em relação aos objetivos e prazos, evitando confusões sobre responsabilidades.

Entregas incrementais e foco em melhorias contínuas: Ao invés de esperar por um longo período para entregar o produto final, o Scrum promove a entrega constante de incrementos de valor ao projeto. Isso nos permite testar funcionalidades, receber feedback antecipado e realizar ajustes de forma iterativa, garantindo que o projeto evolua de maneira eficiente e alinhada às expectativas do professor e do grupo.

Autonomia e colaboração entre os participantes: Com a divisão do trabalho em sprints, cada integrante do time tem maior autonomia para gerenciar suas tarefas e contribuir com suas habilidades. Ao mesmo tempo, o Scrum incentiva a colaboração contínua, especialmente através das daily meetings (weekly mettings em nosso caso), que facilitam a resolução de problemas e promovem um ambiente de aprendizado e cooperação.

Transparência e adaptação: Um dos princípios fundamentais do Scrum é a transparência, com todas as tarefas sendo visíveis e rastreáveis por meio de ferramentas como o Jira, o que facilita a gestão do trabalho e o acompanhamento do progresso. Além disso, como a metodologia ágil prioriza a adaptação, estamos sempre preparados para revisar requisitos e ajustar o planejamento conforme novas ideias e necessidades surgem.

Entrega de valor frequente: O uso de sprints curtos nos permite entregar partes funcionais do sistema continuamente, o que oferece maior visibilidade do progresso ao longo do desenvolvimento. Essa entrega constante garante que cada sprint adicione valor real ao projeto, evitando o risco de acumular trabalho para um prazo final e promovendo a sensação de realização entre os participantes.

## Cerimônias/Cadências

Para implementar a metodologia ágil Scrum no desenvolvimento do nosso projeto, realizamos as seguintes cerimônias e cadências, fundamentais para manter o alinhamento e a organização da equipe:

Weekly Meetings (Reuniões Semanais de Planejamento):

As reuniões semanais foram realizadas para revisar o Product Backlog e planejar as atividades para a próxima sprint. Durante essas reuniões, a equipe discutiu as prioridades e estimou o tempo necessário para completar cada tarefa.

A equipe utilizou essas reuniões como ponto de partida para definir os objetivos da sprint, distribuindo as tarefas conforme as capacidades individuais e as entregas planejadas. Cada membro teve a oportunidade de compartilhar seus avanços, obstáculos e expectativas, assegurando que todos estivessem alinhados.

Essas reuniões serviram também como sessões de refinamento do backlog, ajustando e reorganizando as tarefas conforme novos requisitos e necessidades surgiam.

Sprint Timebox:

As sprints foram organizadas em ciclos de 1 a 2 semanas, variando conforme a complexidade das entregas e os prazos estabelecidos.

Ao início de cada sprint, a equipe estabeleceu metas claras e mensuráveis para serem alcançadas dentro do período estipulado. No final de cada sprint, uma revisão foi realizada para verificar o que foi completado e o que precisaria ser ajustado para as sprints seguintes.

Daily Scrum (Reuniões Diárias):

Embora optássemos por não realizar as daily meetings todos os dias, quando necessário, fizemos check-ins rápidos para acompanhar o progresso e resolver qualquer impedimento que surgisse. Essas reuniões curtas garantiram que todos os membros da equipe estivessem informados sobre os avanços e ajudaram a manter a comunicação fluida.

Sprint Retrospective:

Ao final de cada sprint, realizamos a cerimônia de Retrospectiva, onde a equipe refletiu sobre o que funcionou bem, o que poderia ser melhorado, e quais ajustes seriam necessários para os próximos ciclos. Essa cerimônia foi essencial para promover o aprendizado contínuo e melhorar nossos processos de trabalho.

Durante a retrospectiva, focamos principalmente na identificação de obstáculos que pudessem ter atrapalhado o andamento da sprint, assim como em sugestões para melhorar a colaboração e eficiência nas próximas fases.

## Prints

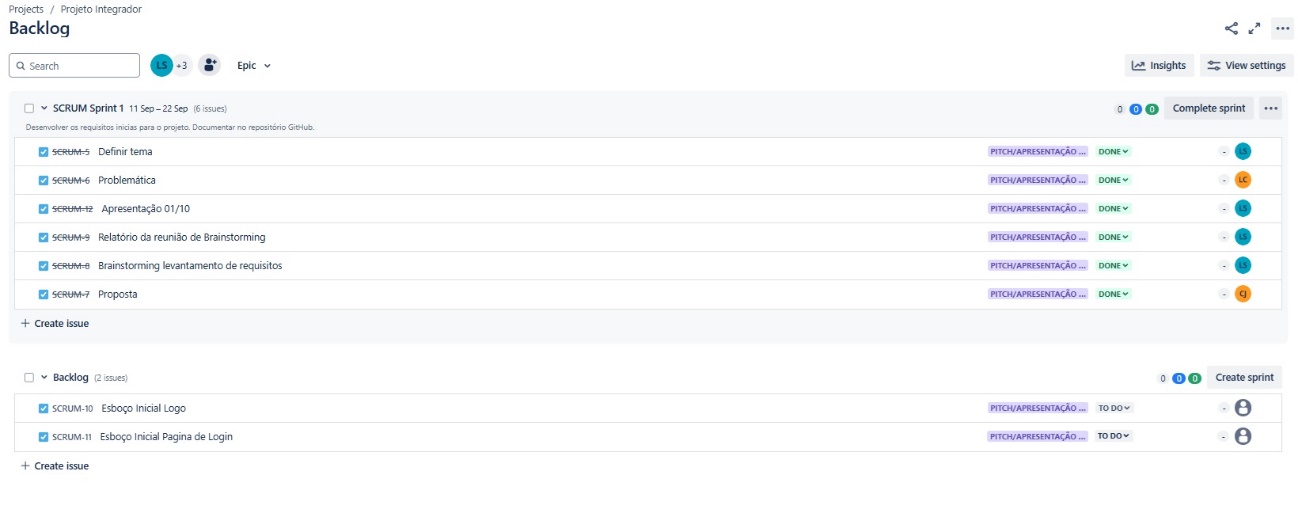


Figura 10 - Backlog-1°Sprint - 11 a 22 Set de 2024

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Figura 11 - 1°Sprint - 11 a 22 Set de 2024

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Figura 12 - Backlog-2°Sprint - 23 a 30 Set de 2024

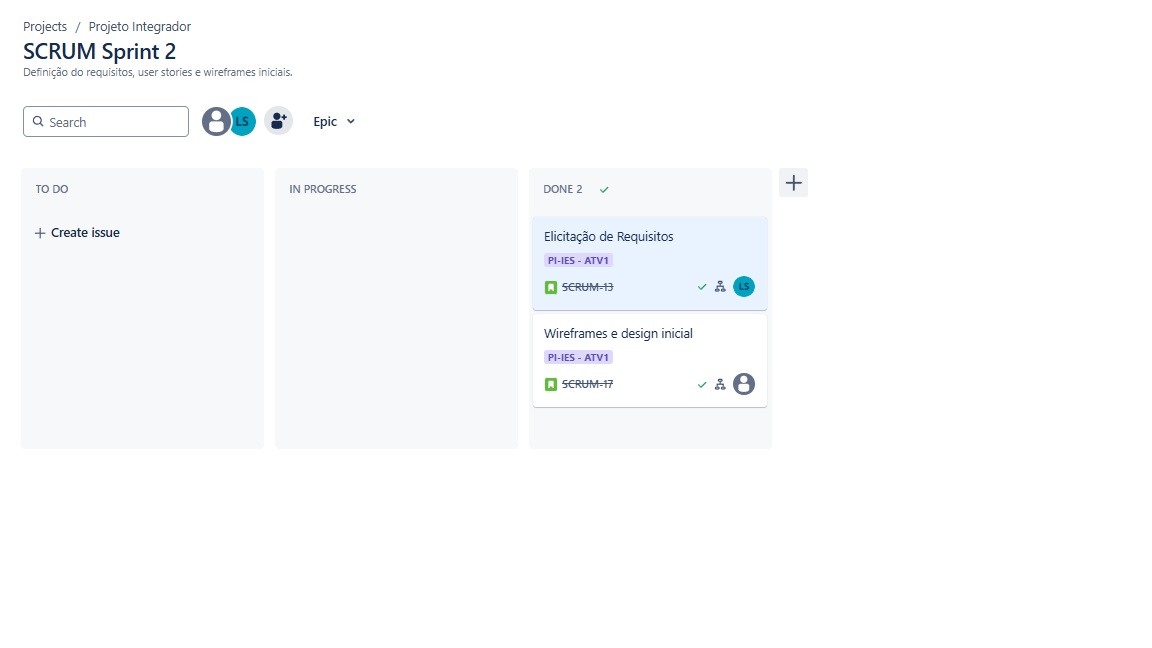


Figura 13 - 2°Sprint - 23 a 30 Set de 2024

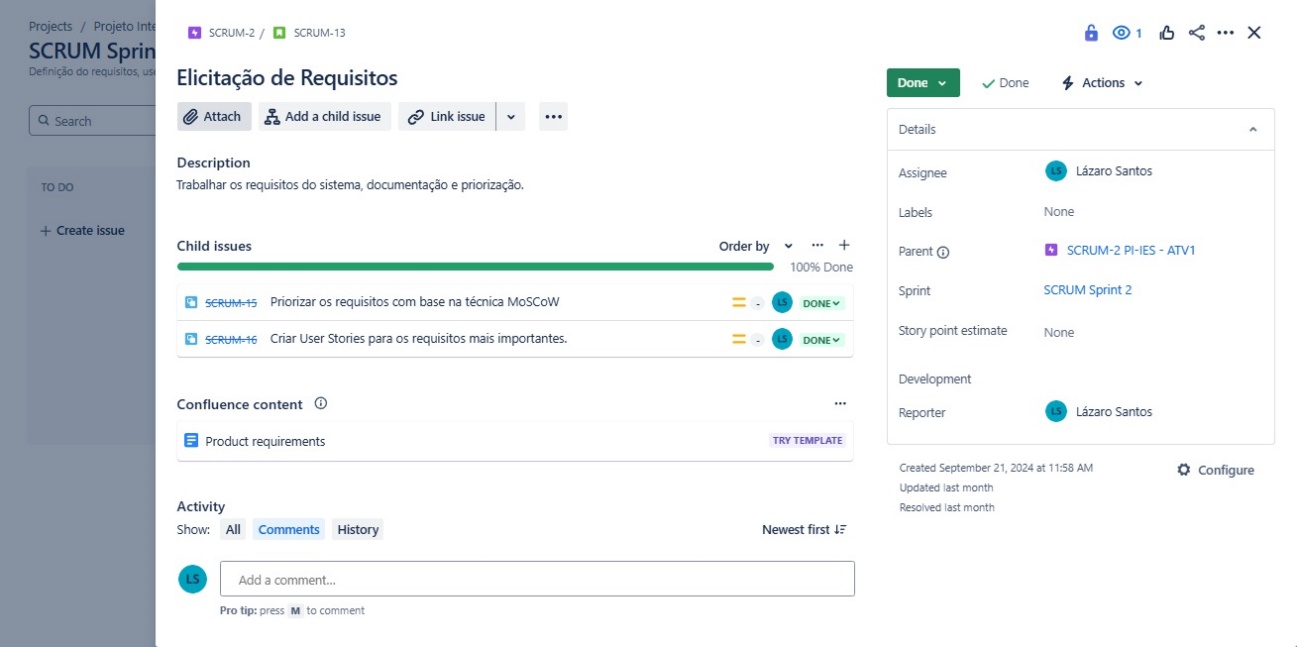


Figura 14 - 2°Sprint - 23 a 30 Set de 2024-Histórias e tarefas

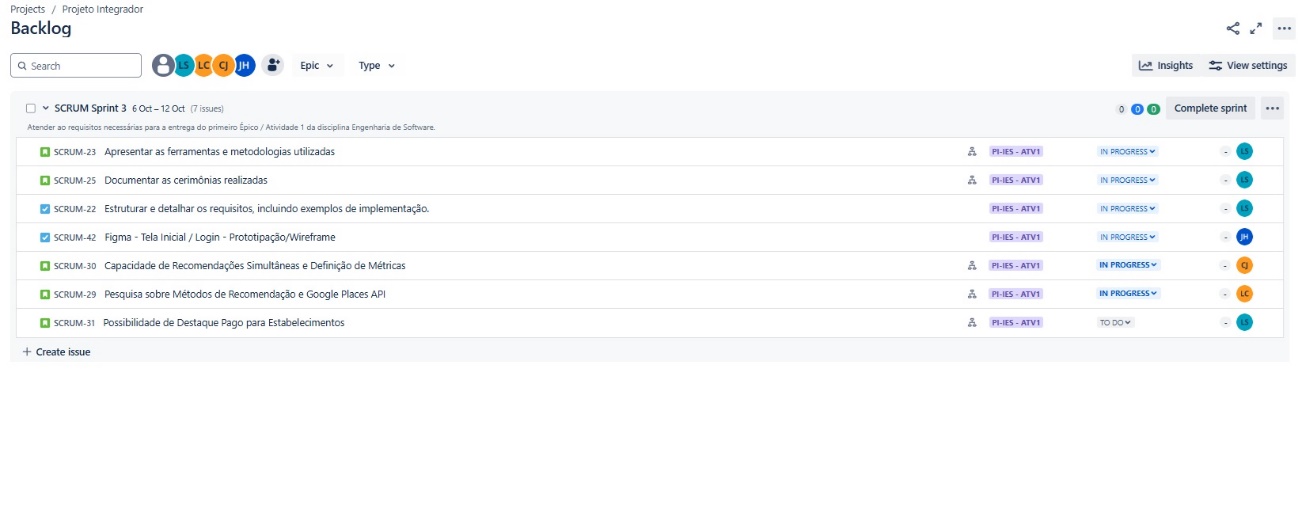


Figura 15 - Backlog-3°Sprint - 06 a 12 Oct de 2024

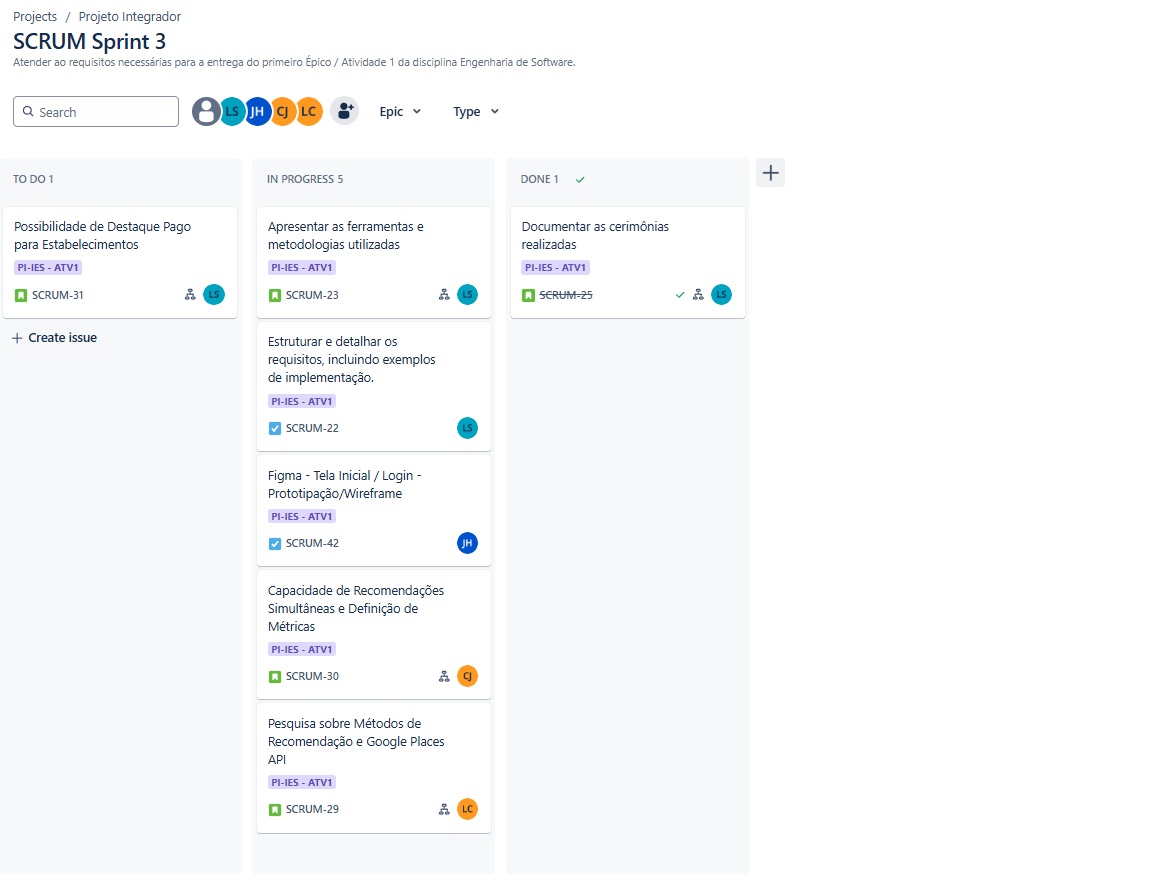


Figura 16 - 3°Sprint - 06 a 12 Oct de 2024

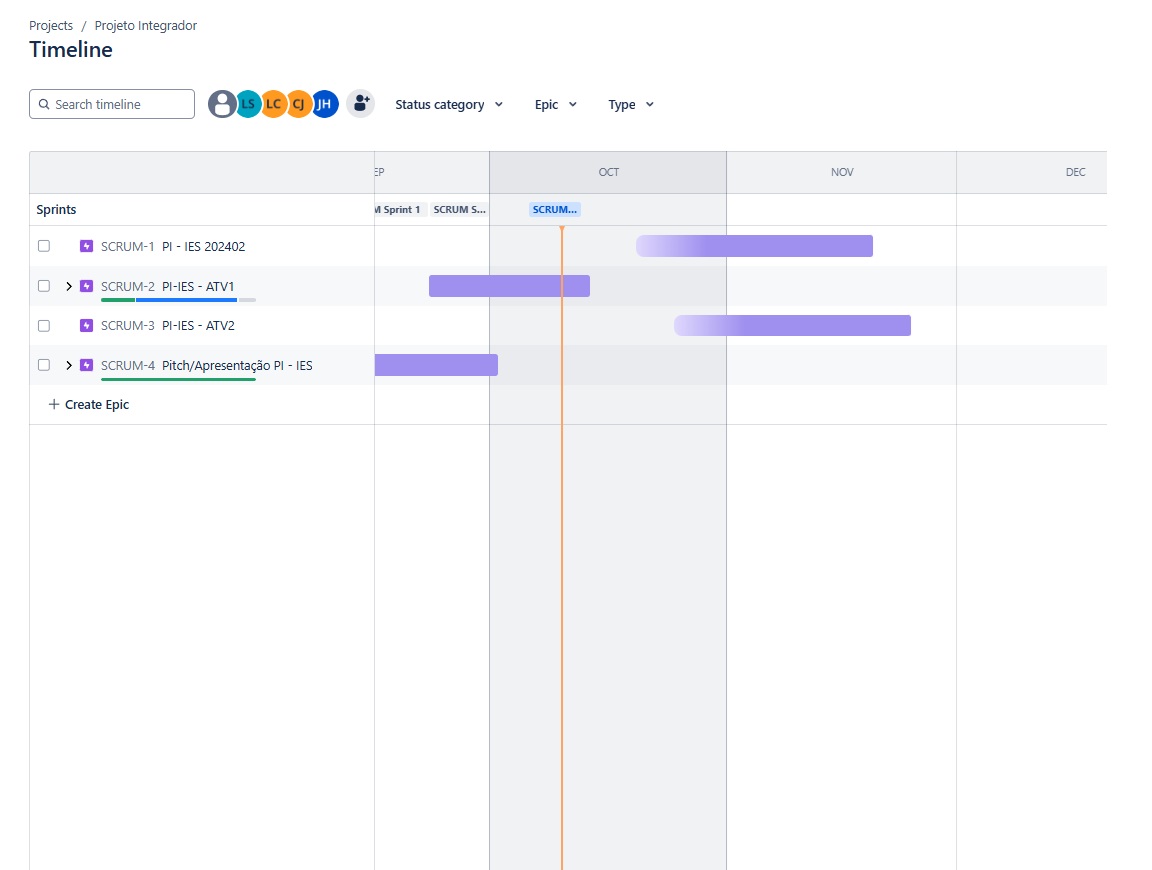


Figura 17 - Timeline

## Ferramentas utilizadas

• Jira

• Teams

• GitHub

## Justificativa de usar tais ferramentas

**Jira**

O Jira foi escolhido como a principal ferramenta de gestão do projeto devido à sua flexibilidade e integração com a metodologia ágil Scrum. Ele nos permitiu organizar o Product Backlog, planejar e acompanhar as tarefas ao longo das sprints, e realizar o monitoramento constante do progresso. Com o Jira, a equipe conseguiu dividir as atividades em tarefas claras e acompanhar facilmente os avanços, prazos e responsabilidades, facilitando a visibilidade geral do projeto.

Benefícios: Visualização clara do progresso, organização das sprints, atribuição de tarefas e priorização de backlog. Além disso, o Jira permite gerar relatórios que ajudam a entender a eficiência da equipe e identificar áreas para melhorias.

**Microsoft Teams**

O Microsoft Teams foi utilizado como ferramenta de comunicação da equipe. A escolha do Teams se deu por sua capacidade de integrar chat, chamadas de vídeo e compartilhamento de documentos em uma única plataforma. Isso facilitou as reuniões semanais, a troca rápida de informações e a colaboração remota entre os membros da equipe.

Benefícios: Comunicação rápida e eficiente, facilidade para organizar reuniões virtuais, compartilhar arquivos e colaborar em tempo real, permitindo uma comunicação fluida entre todos.

**GitHub**

O GitHub foi escolhido como ferramenta de versionamento de código, garantindo que o desenvolvimento do projeto fosse centralizado e que todos os membros da equipe pudessem colaborar simultaneamente. O uso do GitHub permitiu o controle de versões, facilitando o acompanhamento de alterações, criação de branches, e revisão de código.

Benefícios: Controle de versões, colaboração remota entre os desenvolvedores, facilidade para gerenciar o código de forma organizada e garantir a integridade das versões do sistema. Além disso, o GitHub permitiu armazenar documentos e entregas relacionadas ao projeto.

# CONCLUSÃO

Este relatório apresentou uma análise abrangente sobre a implementação de um sistema de recomendação para restaurantes, destacando os métodos de recomendação, a integração com a Google Places API e as estratégias para evitar a superlotação dos estabelecimentos. Além disso, discutiu a viabilidade de um modelo de monetização baseado em recomendações pagas, assim como as melhores práticas para comunicar o desempenho das indicações aos restaurantes. As soluções propostas visam não apenas aprimorar a experiência do usuário, mas também garantir que os estabelecimentos possam gerenciar efetivamente a demanda, promovendo um ambiente equilibrado e sustentável para todos os envolvidos.

Todos os arquivos referentes ao projeto estão disponíveis no repositório:

[lvplazaro/PI-IES0202402: Repositório para o projeto Integrador da disciplina Engenharia de Software 1 - FATEC COtia (github.com)](https://github.com/lvplazaro/PI-IES0202402)