**Documento de Arquitetura de Sistemas – FastTechFoods**

Daniel Rossi de Amorim

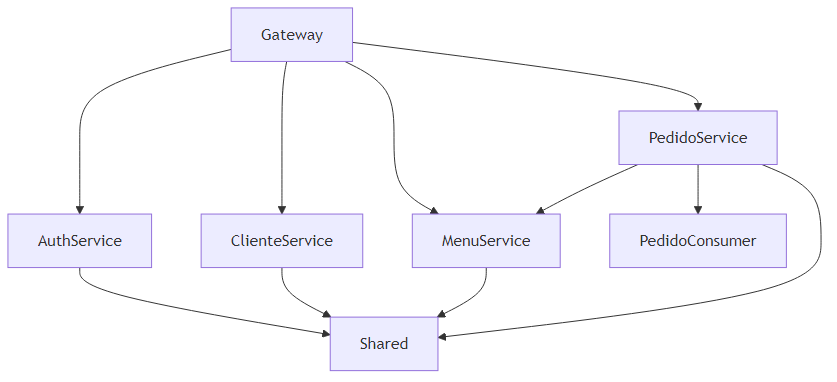
FASE: TechChallenge

Versão: 1.0 Data: 29 de July de 2025

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

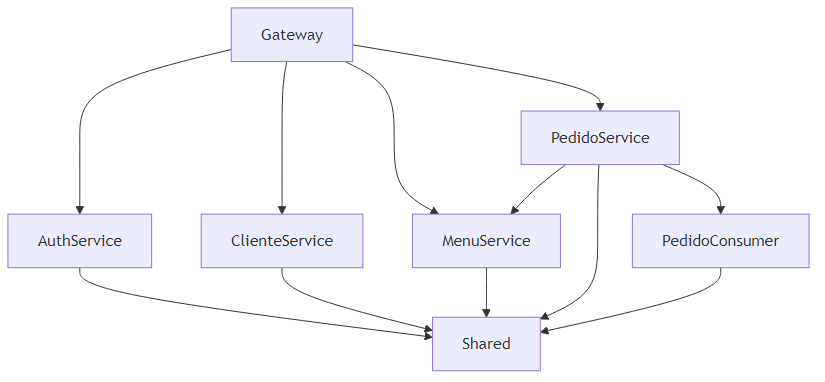
# 1. Introdução

1. Introdução  
  
Esta seção apresenta uma visão geral do sistema FastTechFoods, contextualizando o propósito do projeto, seu escopo e os principais objetivos da arquitetura proposta.  
  
1.1 Objetivo do Documento  
  
O objetivo deste documento é descrever a arquitetura do sistema FastTechFoods, detalhando os principais componentes, suas responsabilidades e as interações entre eles. Este documento serve como referência para desenvolvedores, arquitetos, analistas e demais stakeholders envolvidos no desenvolvimento, manutenção e evolução do sistema.  
  
1.2 Visão Geral do Sistema  
  
O FastTechFoods é um sistema composto por múltiplos serviços, cada um responsável por uma área funcional específica. O sistema foi projetado para atender demandas de escalabilidade, modularidade e facilidade de manutenção, utilizando uma abordagem baseada em microsserviços. Os principais serviços identificados no projeto são:  
  
- Serviço de autenticação (auth-service): Responsável pelo gerenciamento de autenticação e autorização de usuários.  
- Serviço de clientes (cliente-service): Responsável pelo cadastro, consulta e manutenção de informações de clientes.  
- Serviço de menu (menu-service): Responsável pelo gerenciamento dos itens de menu e pedidos.  
- Serviço de pedidos (pedido-service): Responsável pelo processamento e gerenciamento de pedidos realizados pelos clientes.  
- Gateway (FastTechFoods.Gateway): Responsável por centralizar o acesso dos usuários aos serviços internos do sistema.  
- PedidoConsumer: Responsável pelo consumo de eventos relacionados a pedidos, integrando-se com outros serviços conforme necessário.  
- Shared: Biblioteca compartilhada entre os serviços, contendo funcionalidades e modelos comuns.  
  
1.3 Escopo  
  
O escopo deste documento abrange a descrição dos componentes arquiteturais do FastTechFoods, suas responsabilidades, fluxos de interação e os principais pontos de integração entre os serviços. Não estão incluídos neste documento detalhes de implementação, testes ou aspectos operacionais.  
  
1.4 Público-Alvo  
  
Este documento destina-se a desenvolvedores, arquitetos de software, analistas de sistemas, gerentes de projeto e demais profissionais envolvidos no ciclo de vida do sistema FastTechFoods.  
  
1.5 Estrutura do Documento  
  
O documento está organizado em seções que abordam a visão geral da arquitetura, detalhamento dos componentes, fluxos de interação, decisões arquiteturais e considerações finais.  
  
A seguir, apresenta-se um diagrama simplificado dos principais componentes do sistema FastTechFoods e suas interações:



# 2. Arquitetura da Solução

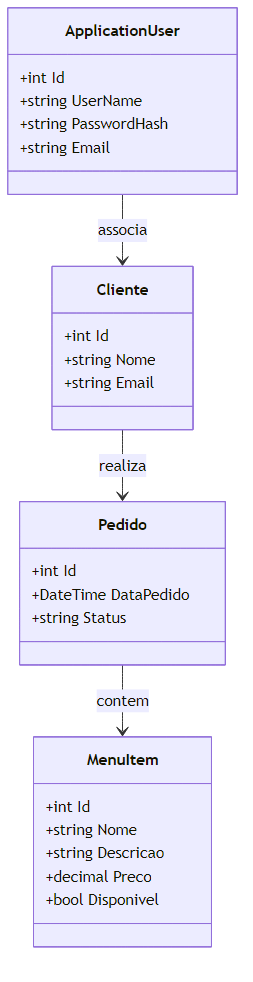
2. Arquitetura da Solução  
  
Esta seção descreve a arquitetura da solução do sistema FastTechFoods, detalhando os principais componentes, suas responsabilidades e as interações entre eles. O objetivo é fornecer uma visão clara e estruturada da composição do sistema, facilitando o entendimento técnico e o alinhamento entre as equipes envolvidas no projeto.  
  
2.1 Visão Geral  
  
O sistema FastTechFoods é composto por múltiplos serviços independentes, organizados em uma arquitetura de microsserviços. Cada serviço é responsável por um domínio específico do negócio, promovendo escalabilidade, manutenção facilitada e independência de desenvolvimento. A comunicação entre os serviços ocorre principalmente por meio de APIs HTTP e mensageria assíncrona.  
  
2.2 Componentes Principais  
  
A seguir, são apresentados os principais componentes do sistema:  
  
- Auth Service: Responsável pela autenticação e autorização de usuários, gerenciamento de credenciais e emissão de tokens JWT.  
- Cliente Service: Gerencia os dados dos clientes, incluindo cadastro, atualização e consulta de informações.  
- Menu Service: Responsável pelo gerenciamento dos itens do cardápio e pedidos, incluindo cadastro, consulta e atualização de itens e pedidos.  
- Pedido Service: Gerencia o ciclo de vida dos pedidos, desde a criação até o processamento, além de publicar eventos relacionados a pedidos.  
- Pedido Consumer: Serviço dedicado ao consumo de eventos de pedidos, permitindo o processamento assíncrono e integração com outros sistemas.  
- Gateway: Atua como ponto de entrada unificado para os clientes, roteando as requisições para os serviços apropriados e centralizando aspectos como autenticação e autorização.  
- Shared: Biblioteca compartilhada entre os serviços, contendo classes e utilitários comuns.  
  
2.3 Diagrama de Componentes  
  
A seguir, apresenta-se um diagrama simplificado dos principais componentes e suas interações:



2.4 Descrição dos Componentes  
  
Auth Service  
O Auth Service é responsável por autenticar usuários e emitir tokens JWT. Ele expõe endpoints para login e registro, validando credenciais e garantindo a segurança do acesso aos demais serviços.  
  
Cliente Service  
O Cliente Service gerencia as informações dos clientes, permitindo operações de cadastro, consulta e atualização. Ele garante a integridade dos dados dos clientes e oferece APIs para integração com outros serviços.  
  
Menu Service  
O Menu Service administra os itens do cardápio e os pedidos. Ele permite o cadastro, atualização e consulta de itens do menu, além de gerenciar os pedidos realizados pelos clientes.  
  
Pedido Service  
O Pedido Service é responsável pelo gerenciamento dos pedidos, desde a criação até o processamento. Ele publica eventos de pedidos criados, permitindo a integração assíncrona com outros componentes do sistema.  
  
Pedido Consumer  
O Pedido Consumer consome eventos de pedidos publicados pelo Pedido Service, processando-os de forma assíncrona. Este componente pode ser utilizado para integrações externas ou processamento adicional de pedidos.  
  
Gateway  
O Gateway centraliza o acesso ao sistema, roteando as requisições dos clientes para os serviços apropriados. Ele também pode implementar políticas de autenticação, autorização e controle de acesso.  
  
Shared  
A biblioteca Shared contém classes e utilitários comuns utilizados por diversos serviços, promovendo o reuso de código e padronização.  
  
2.5 Comunicação entre Componentes  
  
A comunicação entre os serviços ocorre de duas formas principais:  
  
- Comunicação síncrona via HTTP: Utilizada para operações de consulta e comandos diretos entre os serviços, principalmente através do Gateway.  
- Comunicação assíncrona via mensageria: Utilizada para eventos de pedidos, onde o Pedido Service publica eventos que são consumidos pelo Pedido Consumer.  
  
2.6 Considerações de Escalabilidade e Manutenção  
  
A arquitetura baseada em microsserviços permite que cada componente seja escalado de forma independente, conforme a demanda. Além disso, facilita a manutenção e evolução do sistema, pois cada serviço pode ser desenvolvido, testado e implantado de maneira autônoma.  
  
2.7 Resumo  
  
A arquitetura da solução FastTechFoods foi projetada para garantir modularidade, escalabilidade e facilidade de manutenção, utilizando princípios de microsserviços e comunicação eficiente entre os componentes. O diagrama apresentado ilustra as principais interações e responsabilidades de cada serviço dentro do ecossistema do sistema.

# 3. Modelo de Dados (Entidade-Relacionamento)

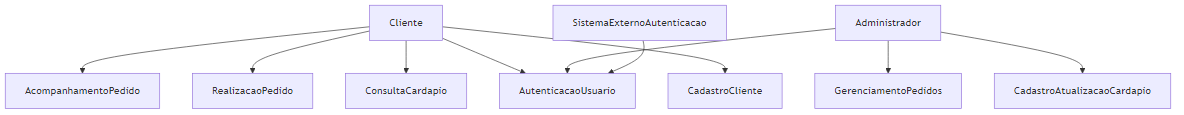
3. Modelo de Dados (Entidade-Relacionamento)  
  
Esta seção descreve o modelo de dados do sistema FastTechFoods, detalhando as principais entidades, seus atributos e os relacionamentos entre elas, conforme identificado nos arquivos do projeto. O objetivo é fornecer uma visão clara da estrutura de dados que suporta as funcionalidades do sistema.  
  
Entidades Principais  
  
1. ApplicationUser  
Representa o usuário autenticado do sistema, responsável pelo acesso e autenticação. Os atributos principais incluem identificador, nome de usuário, senha e outros dados de autenticação.  
  
2. Cliente  
Representa o cliente do sistema, contendo informações cadastrais relevantes para identificação e contato. Os atributos principais incluem identificador, nome, e-mail e outros dados pessoais.  
  
3. MenuItem  
Representa um item do cardápio disponível para pedidos. Os atributos principais incluem identificador, nome do item, descrição, preço e disponibilidade.  
  
4. Pedido  
Representa um pedido realizado por um cliente. Os atributos principais incluem identificador, data do pedido, status, referência ao cliente e aos itens do pedido.  
  
Relacionamentos  
  
- Um ApplicationUser pode estar associado a um Cliente, permitindo a autenticação e o gerenciamento de dados pessoais.  
- Um Cliente pode realizar vários Pedidos.  
- Um Pedido pode conter vários MenuItems, e um MenuItem pode estar presente em vários Pedidos, caracterizando um relacionamento muitos-para-muitos entre Pedido e MenuItem.  
  
Diagrama Entidade-Relacionamento  
  
A seguir, apresenta-se o diagrama de classes Mermaid, representando as entidades e seus relacionamentos:



Observações  
  
- O relacionamento entre Pedido e MenuItem é do tipo muitos-para-muitos, podendo ser implementado por uma tabela associativa, caso necessário.  
- Os atributos listados são os principais identificados nos modelos presentes nos arquivos do projeto.  
- O modelo pode ser expandido conforme a evolução do sistema e a inclusão de novos requisitos.  
  
Este modelo de dados serve como base para o desenvolvimento, integração e manutenção do sistema FastTechFoods, garantindo a consistência e integridade das informações manipuladas.

# 4. Atores e Casos de Uso Principais

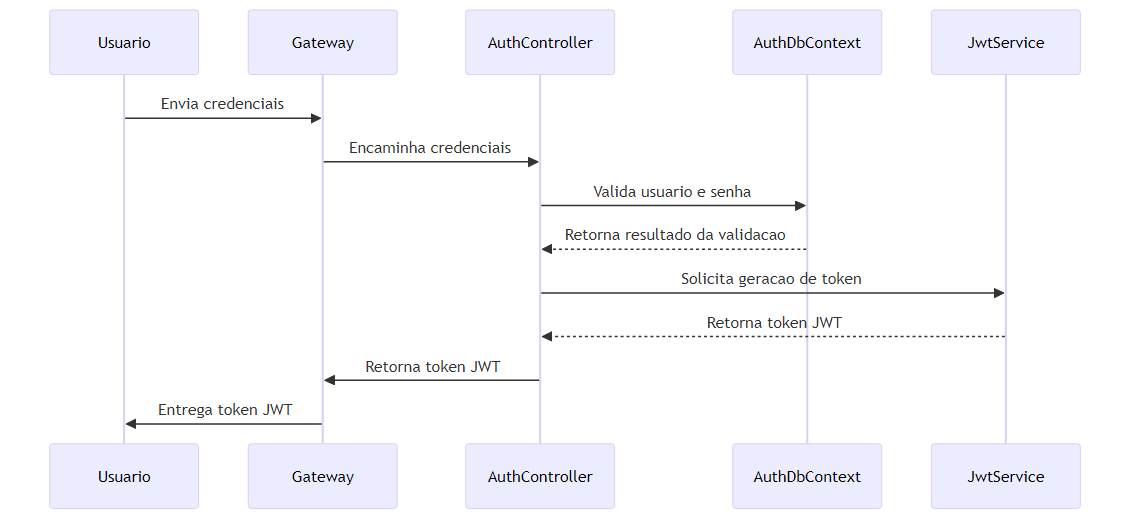
4. Atores e Casos de Uso Principais  
  
Esta seção descreve os principais atores que interagem com o sistema FastTechFoods e os casos de uso fundamentais identificados a partir dos componentes presentes no projeto.  
  
Atores  
  
1. Cliente  
Pessoa que utiliza o sistema para consultar o cardápio, realizar pedidos e acompanhar o status dos mesmos.  
  
2. Administrador  
Usuário responsável pela gestão do sistema, incluindo cadastro e atualização de itens do cardápio, além de gerenciamento de clientes.  
  
3. Sistema Externo de Autenticação  
Serviço responsável por autenticar usuários e fornecer tokens de acesso.  
  
4. Sistema de Pagamento (futuro ou externo)  
Sistema responsável pelo processamento de pagamentos dos pedidos realizados.  
  
Casos de Uso Principais  
  
1. Cadastro de Cliente  
Permite que um novo cliente se registre no sistema, fornecendo informações pessoais necessárias.  
  
2. Autenticação de Usuário  
Permite que clientes e administradores realizem login no sistema, recebendo um token de autenticação.  
  
3. Consulta de Cardápio  
Permite que o cliente visualize os itens disponíveis no cardápio.  
  
4. Cadastro e Atualização de Itens do Cardápio  
Permite que o administrador adicione, remova ou altere itens do cardápio.  
  
5. Realização de Pedido  
Permite que o cliente selecione itens do cardápio e realize um pedido.  
  
6. Acompanhamento de Pedido  
Permite que o cliente acompanhe o status do seu pedido após a realização.  
  
7. Gerenciamento de Pedidos  
Permite que o administrador visualize e gerencie os pedidos realizados.  
  
Fluxo dos Casos de Uso  
  
A seguir, um diagrama ilustrando a interação entre os principais atores e os casos de uso do sistema FastTechFoods:



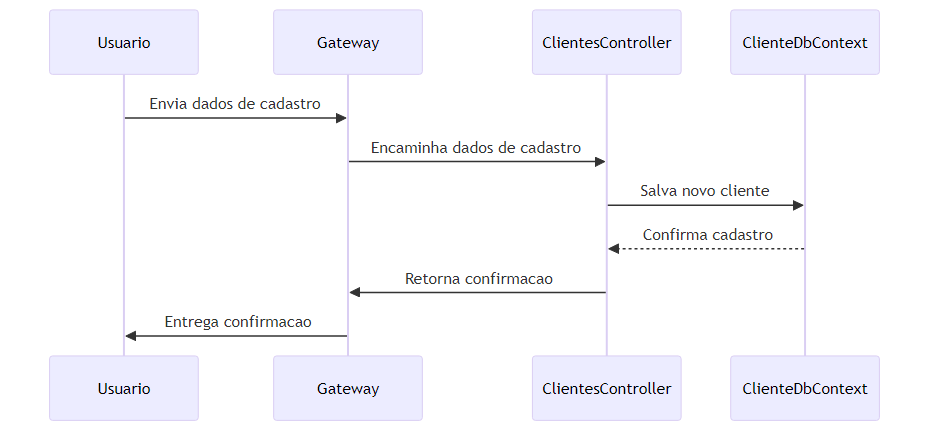
Descrição dos Casos de Uso  
  
Cadastro de Cliente  
O cliente fornece informações pessoais e cria uma conta no sistema. Os dados são armazenados no serviço de clientes.  
  
Autenticação de Usuário  
O cliente ou administrador fornece credenciais de acesso. O sistema de autenticação valida as informações e retorna um token de acesso.  
  
Consulta de Cardápio  
O cliente acessa a lista de itens disponíveis no cardápio, visualizando detalhes como nome, descrição e preço.  
  
Cadastro e Atualização de Itens do Cardápio  
O administrador gerencia os itens do cardápio, podendo adicionar novos itens, atualizar informações ou remover itens existentes.  
  
Realização de Pedido  
O cliente seleciona itens do cardápio e realiza um pedido, que é registrado no sistema de pedidos.  
  
Acompanhamento de Pedido  
Após realizar um pedido, o cliente pode consultar o status do mesmo até a finalização.  
  
Gerenciamento de Pedidos  
O administrador pode visualizar todos os pedidos realizados, acompanhar seu status e tomar ações administrativas necessárias.  
  
Esta estrutura garante que os principais fluxos de interação do sistema FastTechFoods estejam claramente definidos, facilitando o entendimento e a evolução do sistema.

# 5. Fluxos de Trabalho Principais

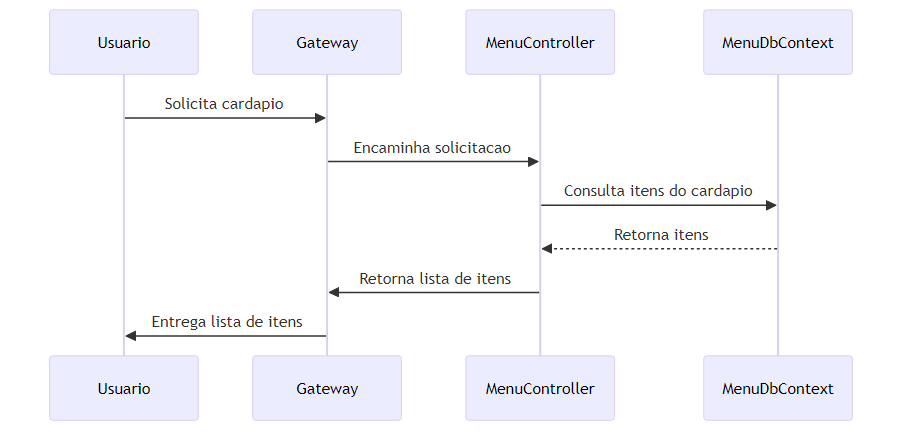
5. Fluxos de Trabalho Principais  
  
Esta seção descreve os principais fluxos de trabalho do sistema FastTechFoods, detalhando as etapas envolvidas em cada processo central, os serviços participantes e a interação entre os componentes do sistema.  
  
5.1 Fluxo de Autenticação de Usuário  
  
O fluxo de autenticação permite que um usuário acesse o sistema de forma segura. O processo envolve a validação das credenciais e a emissão de um token JWT para acesso aos demais serviços.  
  
Etapas do fluxo:  
1. O usuário envia suas credenciais (usuário e senha) para o AuthController do auth-service.  
2. O AuthController valida as credenciais utilizando o AuthDbContext.  
3. Se as credenciais forem válidas, o JwtService gera um token JWT.  
4. O AuthController retorna o token JWT ao usuário.  
  
Diagrama do fluxo de autenticação:



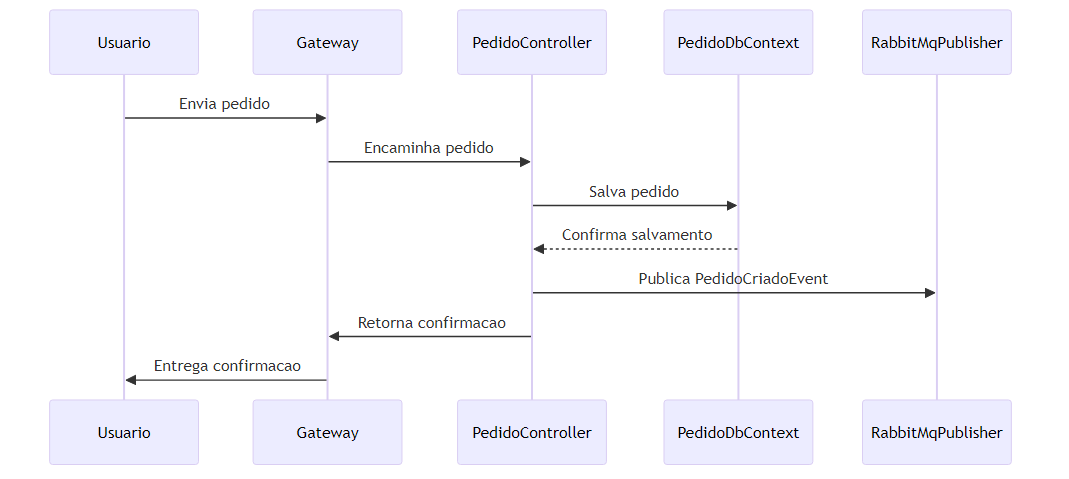
5.2 Fluxo de Cadastro de Cliente  
  
O fluxo de cadastro permite que um novo cliente seja registrado no sistema.  
  
Etapas do fluxo:  
1. O usuário envia os dados de cadastro para o ClientesController do cliente-service.  
2. O ClientesController utiliza o ClienteDbContext para persistir os dados do novo cliente.  
3. O serviço retorna a confirmação do cadastro ao usuário.  
  
Diagrama do fluxo de cadastro de cliente:



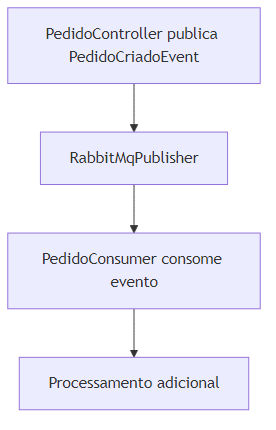
5.3 Fluxo de Consulta de Cardápio  
  
O fluxo de consulta de cardápio permite que o usuário visualize os itens disponíveis para pedido.  
  
Etapas do fluxo:  
1. O usuário solicita a lista de itens do cardápio ao MenuController do menu-service.  
2. O MenuController consulta o MenuDbContext para obter os itens disponíveis.  
3. O serviço retorna a lista de itens ao usuário.  
  
Diagrama do fluxo de consulta de cardápio:



5.4 Fluxo de Criação de Pedido  
  
O fluxo de criação de pedido permite que o usuário realize um pedido de itens do cardápio.  
  
Etapas do fluxo:  
1. O usuário seleciona itens e envia o pedido ao PedidoController do pedido-service.  
2. O PedidoController utiliza o PedidoDbContext para registrar o novo pedido.  
3. O PedidoController publica um evento PedidoCriadoEvent via RabbitMqPublisher.  
4. Outros serviços interessados podem consumir o evento para processamentos adicionais.  
  
Diagrama do fluxo de criação de pedido:



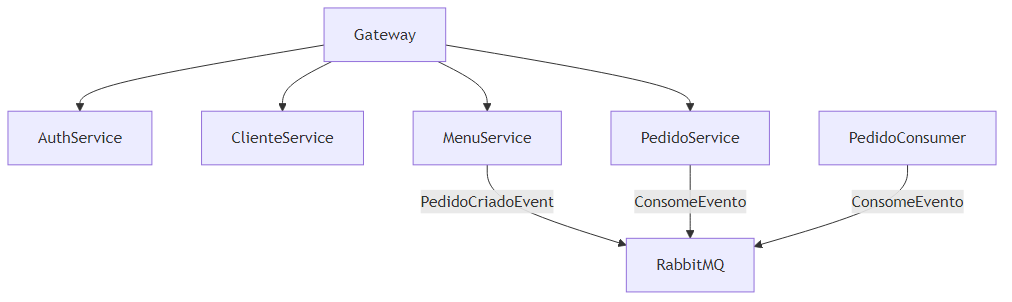
5.5 Fluxo de Processamento de Evento de Pedido Criado  
  
Este fluxo descreve como o evento de pedido criado é processado por consumidores interessados.  
  
Etapas do fluxo:  
1. O RabbitMqPublisher publica o evento PedidoCriadoEvent.  
2. O PedidoConsumer consome o evento e executa o processamento necessário.  
  
Diagrama do fluxo de processamento de evento:



Estes fluxos de trabalho principais garantem a integração entre os serviços do FastTechFoods, promovendo uma experiência eficiente e segura para os usuários finais.

# 6. Integrações

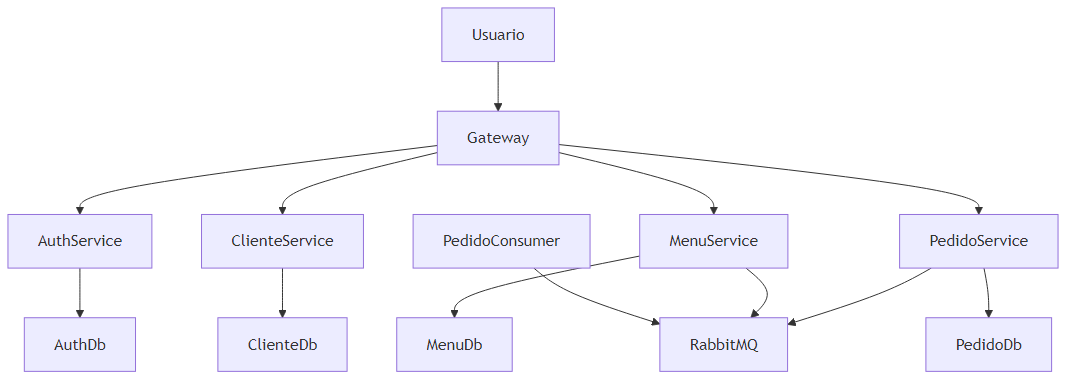
6. Integrações  
  
Esta seção descreve as integrações entre os componentes do sistema FastTechFoods, detalhando como os serviços interagem entre si e com sistemas externos, bem como os mecanismos de comunicação utilizados.  
  
6.1 Integração entre Serviços Internos  
  
O sistema FastTechFoods é composto por múltiplos serviços, cada um responsável por uma área funcional específica. As integrações entre esses serviços são fundamentais para garantir o fluxo de informações e a orquestração dos processos de negócio.  
  
6.1.1 Serviço de Autenticação (auth-service)  
  
O serviço de autenticação é responsável pelo gerenciamento de usuários e autenticação via JWT. Ele expõe endpoints para registro e login de usuários, que são consumidos pelos demais serviços e pelo gateway.  
  
6.1.2 Serviço de Clientes (cliente-service)  
  
O serviço de clientes gerencia os dados dos clientes. Ele expõe endpoints REST para cadastro, consulta e atualização de informações de clientes. O acesso a esses endpoints é protegido por autenticação JWT, fornecida pelo auth-service.  
  
6.1.3 Serviço de Menu (menu-service)  
  
O serviço de menu gerencia os itens do cardápio e os pedidos realizados. Ele expõe endpoints REST para consulta de itens do menu e criação de pedidos. Além disso, publica eventos relacionados a pedidos para integração assíncrona com outros serviços.  
  
6.1.4 Serviço de Pedidos (pedido-service)  
  
O serviço de pedidos é responsável pelo processamento e gerenciamento dos pedidos realizados pelos clientes. Ele consome eventos de pedidos criados, publicados pelo menu-service, e pode publicar novos eventos para integração com outros componentes.  
  
6.1.5 Gateway (FastTechFoods.Gateway)  
  
O gateway atua como ponto de entrada unificado para os clientes do sistema, roteando as requisições para os serviços internos apropriados. Ele consome os endpoints expostos pelos serviços de autenticação, clientes, menu e pedidos.  
  
6.2 Integração Assíncrona  
  
Para garantir escalabilidade e desacoplamento, o sistema utiliza integração assíncrona baseada em eventos para comunicação entre alguns serviços. O mecanismo de mensageria identificado é o RabbitMQ.  
  
6.2.1 Publicação de Eventos  
  
O menu-service publica eventos do tipo PedidoCriadoEvent quando um novo pedido é criado. Esses eventos são enviados para o RabbitMQ, permitindo que outros serviços interessados possam processá-los de forma assíncrona.  
  
6.2.2 Consumo de Eventos  
  
O pedido-service e o componente PedidoConsumer consomem eventos PedidoCriadoEvent do RabbitMQ. Isso permite que o processamento de pedidos seja realizado de forma desacoplada do serviço de menu.  
  
6.3 Integração com Sistemas Externos  
  
Não foram identificadas integrações com sistemas externos nos arquivos analisados.  
  
6.4 Diagrama de Integração  
  
O diagrama a seguir ilustra as integrações entre os principais componentes do sistema FastTechFoods, destacando os fluxos de comunicação síncrona (REST) e assíncrona (eventos via RabbitMQ).



Legenda:  
- Gateway: FastTechFoods.Gateway  
- AuthService: auth-service  
- ClienteService: cliente-service  
- MenuService: menu-service  
- PedidoService: pedido-service  
- PedidoConsumer: PedidoConsumer  
- RabbitMQ: Broker de mensagens para eventos assíncronos  
  
6.5 Considerações de Segurança  
  
Todas as integrações REST entre os serviços são protegidas por autenticação JWT, fornecida pelo auth-service. O acesso aos eventos no RabbitMQ deve ser restrito aos serviços autorizados, garantindo a integridade e confidencialidade das mensagens.  
  
6.6 Resumo  
  
As integrações do FastTechFoods são realizadas principalmente via APIs REST autenticadas e eventos assíncronos via RabbitMQ, promovendo desacoplamento, escalabilidade e segurança entre os componentes do sistema.

# 7. Tecnologias Utilizadas

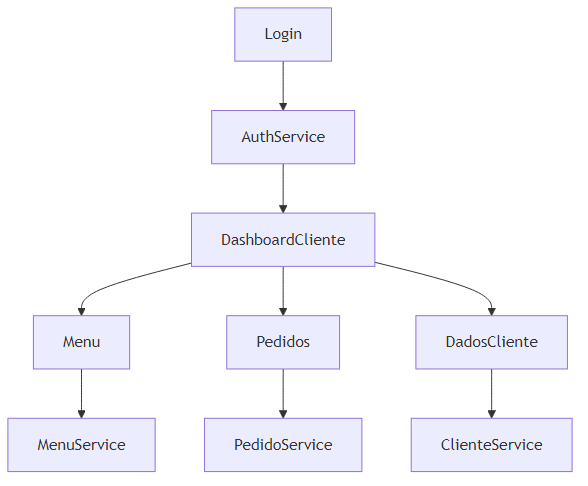
Seção 7. Tecnologias Utilizadas  
  
Esta seção descreve detalhadamente as tecnologias empregadas no desenvolvimento do sistema FastTechFoods, considerando os componentes identificados nos arquivos do projeto.  
  
Plataforma e Linguagem de Programação  
  
O sistema FastTechFoods é desenvolvido utilizando a plataforma .NET, com a linguagem de programação C#. Todos os serviços, incluindo autenticação, gerenciamento de clientes, menu, pedidos e gateway, são implementados como aplicações .NET.  
  
Frameworks e Bibliotecas  
  
O projeto utiliza o ASP.NET Core para a construção de APIs RESTful e aplicações web. Os controladores presentes nos serviços Auth, Cliente, Menu e Pedido indicam o uso deste framework para a exposição de endpoints HTTP e manipulação de requisições.  
  
Para a persistência de dados, é utilizado o Entity Framework Core, que provê o mapeamento objeto-relacional (ORM) entre as classes de domínio e os bancos de dados relacionais. A presença de classes de contexto (DbContext) e arquivos de migração em cada serviço confirma o uso desta tecnologia.  
  
Mensageria  
  
O sistema emprega o RabbitMQ como solução de mensageria para comunicação assíncrona entre serviços. Isso é evidenciado pela presença de classes relacionadas à publicação e consumo de eventos, como RabbitMqPublisher e PedidoCriadoEvent, além do projeto PedidoConsumer, responsável pelo processamento de mensagens.  
  
Estrutura de Microsserviços  
  
O FastTechFoods adota uma arquitetura baseada em microsserviços, com cada domínio funcional implementado como um serviço independente. Os serviços identificados são:  
  
- Auth Service: responsável pela autenticação e autorização de usuários.  
- Cliente Service: responsável pelo gerenciamento de dados de clientes.  
- Menu Service: responsável pela gestão de itens de menu e pedidos.  
- Pedido Service: responsável pelo processamento de pedidos.  
- PedidoConsumer: responsável pelo consumo de eventos relacionados a pedidos.  
- FastTechFoods Gateway: responsável pelo roteamento de requisições e centralização do acesso aos serviços.  
  
Banco de Dados  
  
Cada serviço possui seu próprio contexto de banco de dados, indicando o uso de bancos de dados relacionais independentes para cada domínio. O Entity Framework Core é utilizado para o gerenciamento das migrações e do acesso aos dados.  
  
Testes Automatizados  
  
O projeto inclui testes automatizados, conforme evidenciado pelo arquivo AuthControllerTests.cs, indicando a utilização de frameworks de teste compatíveis com .NET para garantir a qualidade e a robustez do sistema.  
  
Diagrama de Fluxo das Tecnologias Utilizadas



Legenda:  
- Usuario: Usuário final do sistema  
- Gateway: FastTechFoods Gateway  
- AuthService: Serviço de autenticação  
- ClienteService: Serviço de clientes  
- MenuService: Serviço de menu  
- PedidoService: Serviço de pedidos  
- PedidoConsumer: Consumidor de eventos de pedidos  
- RabbitMQ: Sistema de mensageria  
- AuthDb, ClienteDb, MenuDb, PedidoDb: Bancos de dados de cada serviço  
  
Resumo  
  
O sistema FastTechFoods utiliza tecnologias modernas e consolidadas para o desenvolvimento de aplicações distribuídas, com ênfase em microsserviços, APIs RESTful, mensageria assíncrona e persistência de dados relacional. O uso do .NET, ASP.NET Core, Entity Framework Core e RabbitMQ garante escalabilidade, modularidade e facilidade de manutenção para o sistema.

# 8. Interface e Dashboards

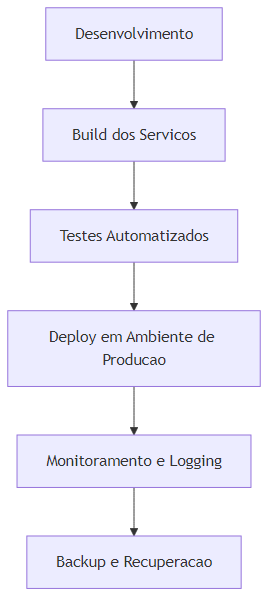
8. Interface e Dashboards  
  
Esta seção descreve as interfaces e dashboards do sistema FastTechFoods, detalhando os principais pontos de interação do usuário, os fluxos de navegação e a integração com os serviços do backend. O objetivo é garantir clareza sobre como os usuários acessam e utilizam as funcionalidades do sistema, bem como como as informações são apresentadas visualmente.  
  
8.1. Visão Geral das Interfaces  
  
O sistema FastTechFoods utiliza uma arquitetura baseada em microsserviços, com um gateway centralizando o acesso dos usuários. As interfaces são compostas por páginas web acessíveis via navegador, servidas pelo componente FastTechFoods.Gateway. As principais interfaces identificadas são:  
  
- Tela de Login e Registro de Usuário  
- Dashboard do Cliente  
- Dashboard de Pedidos  
- Dashboard de Menu  
  
8.2. Tela de Login e Registro  
  
A tela de login permite que usuários existentes autentiquem-se no sistema. O registro de novos usuários é realizado por meio de um formulário que coleta informações básicas, como nome, e-mail e senha. Ambas as operações interagem com o auth-service, responsável pela autenticação e autorização.  
  
8.3. Dashboard do Cliente  
  
Após autenticação, o usuário é direcionado ao dashboard do cliente, onde pode visualizar e atualizar seus dados cadastrais. As informações são obtidas do cliente-service, que gerencia o cadastro de clientes.  
  
8.4. Dashboard de Pedidos  
  
O dashboard de pedidos apresenta ao usuário uma lista de pedidos realizados, com detalhes como status, itens, valores e datas. O usuário pode criar novos pedidos, visualizar o histórico e acompanhar o andamento. As operações de consulta e criação de pedidos são realizadas por meio do pedido-service.  
  
8.5. Dashboard de Menu  
  
O dashboard de menu exibe os itens disponíveis para pedido, com informações como nome, descrição e preço. O usuário pode navegar pelo menu, selecionar itens e adicioná-los ao pedido. Os dados são fornecidos pelo menu-service.  
  
8.6. Fluxo de Navegação  
  
O fluxo de navegação entre as interfaces é centralizado pelo gateway, que direciona as requisições dos usuários para os serviços apropriados. O diagrama a seguir ilustra o fluxo principal de navegação do usuário:



8.7. Integração com Serviços  
  
Cada dashboard ou interface realiza chamadas ao backend por meio do gateway, que encaminha as solicitações para os microsserviços correspondentes. O gateway atua como ponto único de entrada, garantindo segurança e roteamento adequado.  
  
8.8. Considerações de Usabilidade  
  
As interfaces são projetadas para serem intuitivas, responsivas e acessíveis, visando proporcionar uma experiência de usuário eficiente. As informações são apresentadas de forma clara, com feedback visual para ações do usuário, como confirmações de pedido e mensagens de erro em caso de falhas de autenticação ou validação.  
  
8.9. Resumo  
  
As interfaces e dashboards do FastTechFoods são organizados para facilitar o acesso dos usuários às principais funcionalidades do sistema, integrando-se de forma transparente com os microsserviços de autenticação, cadastro de clientes, menu e pedidos. O gateway centraliza o fluxo de navegação e garante a comunicação eficiente entre frontend e backend.

# 9. Implantação (Considerações Gerais)

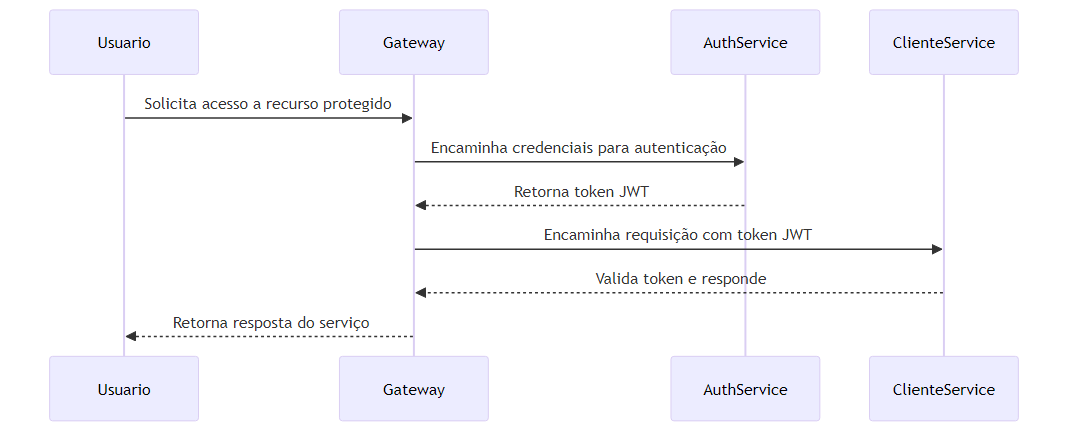
9. Implantação (Considerações Gerais)  
  
Esta seção apresenta as considerações gerais para a implantação do sistema FastTechFoods, abordando aspectos relevantes para garantir uma transição eficiente do ambiente de desenvolvimento para o ambiente de produção, bem como a operação contínua e estável dos serviços.  
  
Ambiente de Implantação  
  
O sistema FastTechFoods é composto por múltiplos serviços independentes, cada um responsável por uma funcionalidade específica. Os serviços identificados são: auth-service, cliente-service, menu-service, pedido-service, FastTechFoods.Gateway e PedidoConsumer. Cada serviço deve ser implantado em ambientes isolados, preferencialmente utilizando contêineres para facilitar o gerenciamento, escalabilidade e portabilidade.  
  
Configuração de Serviços  
  
Cada serviço deve ser configurado com variáveis de ambiente apropriadas para conexão com bancos de dados, filas de mensagens e outros recursos externos. Recomenda-se o uso de arquivos de configuração separados para ambientes de desenvolvimento, homologação e produção, garantindo que informações sensíveis, como strings de conexão e segredos, não sejam expostas.  
  
Banco de Dados  
  
Cada serviço possui seu próprio contexto de banco de dados, conforme identificado pelos arquivos de contexto e migrações. A implantação deve garantir que os bancos de dados estejam acessíveis e que as migrações sejam aplicadas antes do início dos serviços, evitando inconsistências de esquema.  
  
Comunicação entre Serviços  
  
A comunicação entre os serviços ocorre por meio de APIs HTTP e, no caso do pedido-service e PedidoConsumer, por meio de mensageria (RabbitMQ). É fundamental garantir que os endpoints estejam acessíveis e que as filas de mensagens estejam corretamente configuradas e disponíveis no ambiente de produção.  
  
Orquestração e Monitoramento  
  
Recomenda-se o uso de ferramentas de orquestração de contêineres, como Kubernetes ou Docker Compose, para gerenciar o ciclo de vida dos serviços, facilitar o balanceamento de carga e permitir escalabilidade horizontal. Além disso, é importante implementar soluções de monitoramento e logging centralizado para acompanhar o desempenho dos serviços e identificar rapidamente eventuais falhas.  
  
Disponibilidade e Recuperação  
  
Os serviços devem ser implantados em ambientes redundantes, sempre que possível, para garantir alta disponibilidade. Estratégias de backup e recuperação de dados devem ser definidas para os bancos de dados de cada serviço, minimizando o risco de perda de informações em caso de falhas.  
  
Segurança  
  
A implantação deve considerar práticas de segurança, como o uso de conexões seguras (HTTPS), restrição de acesso a portas e endpoints sensíveis, e gerenciamento adequado de credenciais. O auth-service deve ser configurado para proteger os fluxos de autenticação e autorização.  
  
Atualizações e Rollback  
  
O processo de implantação deve prever mecanismos para atualização contínua dos serviços, com possibilidade de rollback em caso de falhas. Recomenda-se o uso de pipelines de integração e entrega contínua (CI/CD) para automatizar o processo de build, teste e deploy.  
  
Fluxo Geral de Implantação  
  
A seguir, um diagrama exemplificando o fluxo geral de implantação dos serviços do FastTechFoods:



Considerações Finais  
  
A implantação do FastTechFoods deve ser cuidadosamente planejada para garantir a integridade, disponibilidade e segurança dos serviços. A adoção de práticas modernas de DevOps, automação e monitoramento contínuo é fundamental para o sucesso operacional do sistema.

# 10. Segurança e Considerações Finais

10. Segurança e Considerações Finais  
  
Segurança  
  
O sistema FastTechFoods adota práticas de segurança em múltiplas camadas para proteger dados sensíveis, garantir a integridade das operações e mitigar riscos de ataques. A seguir, são detalhados os principais aspectos de segurança considerados na arquitetura:  
  
Autenticação e Autorização  
  
O serviço auth-service é responsável pelo gerenciamento de autenticação e autorização dos usuários. Ele utiliza mecanismos robustos para validação de credenciais e geração de tokens JWT, garantindo que apenas usuários autenticados possam acessar recursos protegidos. O controle de acesso é realizado por meio de roles e permissões, restringindo operações conforme o perfil do usuário.  
  
Proteção de Dados  
  
Os dados dos usuários, clientes, pedidos e itens de menu são armazenados em bancos de dados isolados por serviço, reduzindo a superfície de ataque. O acesso aos dados é restrito aos serviços responsáveis, evitando exposição desnecessária de informações. Recomenda-se o uso de criptografia em repouso e em trânsito para proteger dados sensíveis, especialmente informações de autenticação e identificação pessoal.  
  
Comunicação Segura  
  
A comunicação entre os serviços é realizada por meio de APIs REST e mensageria (RabbitMQ). Recomenda-se o uso de HTTPS para todas as APIs, garantindo a confidencialidade e integridade dos dados trafegados. Para a mensageria, recomenda-se a configuração de autenticação e autorização no broker, além de canais seguros.  
  
Validação e Sanitização de Dados  
  
Todos os pontos de entrada dos serviços, especialmente controladores e endpoints públicos, devem realizar validação rigorosa dos dados recebidos. Isso previne ataques como injeção de SQL, XSS e outros vetores comuns. Os DTOs utilizados nos serviços são projetados para receber apenas os campos necessários, reduzindo o risco de exposição de dados indevidos.  
  
Gerenciamento de Dependências e Atualizações  
  
Os serviços devem manter suas dependências atualizadas, aplicando correções de segurança regularmente. O uso de ferramentas automatizadas para detecção de vulnerabilidades em bibliotecas e frameworks é recomendado.  
  
Monitoramento e Auditoria  
  
A arquitetura prevê a implementação de logs centralizados e monitoramento de eventos relevantes, como tentativas de acesso não autorizado, falhas de autenticação e operações críticas. Esses registros devem ser protegidos contra alterações e acessos não autorizados, permitindo auditoria e resposta a incidentes.  
  
Considerações Finais  
  
A arquitetura do FastTechFoods foi projetada para ser modular, escalável e resiliente, facilitando a manutenção e evolução do sistema. A separação de responsabilidades entre os serviços permite isolar falhas e aplicar medidas de segurança específicas para cada contexto.  
  
A adoção de práticas recomendadas de segurança, aliada ao monitoramento contínuo e à atualização constante dos componentes, contribui para a robustez e confiabilidade do sistema. Recomenda-se a realização periódica de testes de segurança, revisões de código e auditorias para identificar e corrigir possíveis vulnerabilidades.  
  
A seguir, um diagrama simplificado ilustra o fluxo de autenticação e autorização no sistema:



O compromisso com a segurança e a observância das melhores práticas são fundamentais para o sucesso e a confiança dos usuários no FastTechFoods.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Daniel Rossi de Amorim