UNIVERSIDADE FEDERAL DE MATO GROSSO
CAMPUS UNIVERSITÁRIO DE VÁRZEA GRANDE
FACULDADE DE ENGENHARIA

Uma API REST para emissão de documentos fiscais

Alisson Cerutti

Orientadora: Prof.ª Dr.ª Gracyeli Santos Souza Guarienti

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Computação da FAENG/CUVG/UFMT (área de concentração: Engenharia de Computação) como parte dos requisitos para obtenção do título de Bacharel em Engenharia de Computação.

Cuiabá, MT, 26 de Outubro de 2023

Dados Internacionais de Catalogação na Fonte.

C418a Cerutti, Alisson.
Uma API REST para emissão de documentos fiscais [recurso eletrônico] / Alisson Cerutti - - Dados eletrônicos (1 arquivo : 62 f., il. color., pdf). - - 2023.

Orientadora: Gracyeli Santos Souza Guarienti.
TCC (graduação em Engenharia de Computação) - Universidade Federal de Mato Grosso, Instituto de Engenharia, Várzea Grande, 2023.
Modo de acesso: World Wide Web: https://bdm.ufmt.br.
Inclui bibliografia.

1. Typescript. 2.Node.js. 3.REST API. 4.NF-e. 5.CT-e. I. Guarienti, Gracyeli Santos Souza, *orientador*. II. Titulo

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

Uma API REST para emissão de documentos fiscais

Alisson Cerutti

Trabalho de Conclusão de Curso aprovado em 26 de Outubro de 2023 pela banca examinadora composta pelos seguintes membros:

Prof.ª Dr.ª Gracyeli Santos Souza Guarienti (orientadora) FAENG/CUVG/UFMT

Prof. Dr. Raoni Florentino da Silva Teixeira ..... FAENG/CUVG/UFMT

Prof. Dr. Gustavo Post Sabin ..... FAENG/CUVG/UFMT

Aos meus familiares, amigos, e professores pela dedicação e apoio durante a realização deste trabalho.

Agradecimentos

A minha orientadora, Prof.ª Dr.ª Gracyeli Santos Souza Guarienti, sou grato pela orientação e ajuda na revisão deste trabalho.

Aos amigos que fiz durante a graduação pelas sugestões e apoio durante o desenvolvimento deste trabalho.

À minha família e minha namorada pelo apoio durante esta jornada.

Resumo

Este trabalho apresenta o desenvolvimento de uma solução para a automação do cálculo de impostos e a geração de documentos fiscais por meio de uma REST API implementada em TypeScript com Node.js. A necessidade de simplificar e otimizar processos fiscais é cada vez mais evidente, considerando a complexidade das regulamentações fiscais em nosso cenário tributário. O objetivo deste projeto é oferecer uma solução abrangente e eficaz para empresas que necessitam calcular impostos e gerar documentos fiscais, incluindo NF-e, NFC-e e CT-e, bem como seus documentos auxiliares, DANFE e DACTE.

Palavras-chave: Typescript, Node.js, REST API, NF-e, CT-e.

Abstract

This work presents the development of a solution for the automation of tax calculation and the generation of tax documents through a REST API implemented in TypeScript with Node.js. The need to simplify and optimize tax processes is becoming increasingly evident, considering the complexity of tax regulations in our tax scenario. The objective of this project is to provide a comprehensive and effective solution for companies that need to calculate taxes and generate tax documents, including NF-e, NFC-e, and CT-e, as well as their accompanying documents, DANFE and DACTE.

Keywords: Typescript, Node.js, REST API, NF-e, CT-e.

Sumário

Sumário i

Lista de Figuras iii

Lista de Tabelas v

Lista de Listagens vii

Lista de Símbolos e Abreviaturas ix

1 Introdução 1

2 Referencial Teórico 3

2.1 Trabalhos relacionados 3

2.2 Procedimentos, materiais e equipamentos 4

2.2.1 Visual Studio Code 5

2.2.2 TypeScript 5

2.2.3 Node.js 6

2.2.4 Banco de dados 6

2.2.5 PrismaORM 6

2.2.6 API REST 7

2.2.7 Insomnia 7

2.2.8 Docker 7

2.2.9 Azure Data Studio 8

2.2.10 Jest 8

3 Materiais e métodos **9**

3.1 Casos de uso 10

3.2 Modelagem do banco de dados 13

3.3 Desenvolvimento 15

4 Resultados **21**

5 Conclusão **33**

Referências bibliográficas **34**

Anexo A Termo de Autorização do Autor **37**

Lista de Figuras

3.1 Caso de uso geral do sistema 10

3.2 Diagrama entidade-relacionamento. 13

3.3 Fluxo dos dados. 15

3.4 Exemplo de requisição. Fonte: SEFAZ 16

3.5 A arquitetura limpa. Fonte: Martin (2019) 17

3.6 Estrutura do projeto. 19

4.1 Exemplo de DANFE gerado pela API. 26

4.2 Exemplo de DACTE gerado pela API. 27

Lista de Tabelas

3.1 Caso de uso para fazer login. 11

3.2 Caso de uso para o endpoint de clientes. 11

3.3 Caso de uso para o endpoint de cálculo de impostos. 11

3.4 Caso de uso para o endpoint da NF-e. 12

3.5 Caso de uso para o endpoint da NFC-e. 12

3.6 Caso de uso para o endpoint da CT-e. 12

3.7 Descrição da tabela de usuários. 13

3.8 Descrição da tabela de estados. 13

3.9 Descrição da tabela de cidades. 14

3.10 Descrição da tabela de clientes. 14

3.11 Descrição da tabela de logs. 14

3.12 Descrição da tabela de nfes. 14

3.13 Descrição da tabela de nfces. 14

3.14 Descrição da tabela de ctes. 15

3.15 Descrição da tabela de eventos. 15

3.16 Endpoints da API. 20

Lista de Listagens

4.1 Exemplo de requisição em JSON para emissão de documento fiscal. . . . 22

4.2 Exemplo de retorno em JSON da emissão do documento fiscal. 26

4.3 Exemplo de requisição em JSON para cancelamento de documento fiscal. 28

4.4 Exemplo de retorno em JSON do cancelamento de documento fiscal. . . . 28

4.5 Exemplo de requisição em JSON para inutilizar numeração de documento fiscal. 29

4.6 Exemplo de retorno em JSON da inutilização da numeração do documento fiscal. 29

4.7 Exemplo de requisição em JSON para carta de correção do documento fiscal. 30

4.8 Exemplo de retorno em JSON da carta de correção do documento fiscal. . 30

4.9 Exemplo de requisição em JSON para cálculo de imposto. 31

4.10 Exemplo de retorno em JSON para cálculo de imposto. 31

Lista de Símbolos e Abreviaturas

API: Interface de Programação de Aplicação

CT-e: Conhecimento de Transporte Eletrônico

DACTE: Documento Auxiliar do Conhecimento de Transporte Eletrônico

DANFE: Documento Auxiliar da Nota Fiscal Eletrônica

ERP: Planejamento de recursos empresariais

HTML: Linguagem de Marcação de Hipertexto

JSON: Notação de Objeto JavaScript

MDF-e: Manifesto Eletrônico de Documentos Fiscais

NF-e: Nota Fiscal Eletrônica

NFC-e: Nota Fiscal do Consumidor Eletrônica

NFS-e: Nota Fiscal de Serviço Eletrônica

ORM: Mapeamento objeto-relacional

REST: Transferência de Estado Representacional

SEFAZ: Secretaria de Estado da Fazenda

SQL: Linguagem de Consulta Estruturada

XML: Linguagem de Marcação Extensível

FAENG: Faculdade de Engenharia

UFMT: Universidade Federal de Mato Grosso

Capítulo 1

Introdução

Com a crescente popularização dos microserviços, muitas empresas estão optando por essa abordagem em vez dos sistemas monolíticos. A proposta da arquitetura baseada em microserviços é criar sistemas mais flexíveis, escaláveis e de manutenção mais simples. Durante o desenvolvimento de um ERP (*Enterprise Resource Planning*), uma das partes mais complexas é a relacionada à gestão fiscal, abrangendo desde cálculos de impostos até a emissão de documentos fiscais, como NF-e, NFC-e e CT-e. Diante desse cenário, identificou-se a necessidade de criar um microserviço dedicado à geração e transmissão de documentos fiscais, atuando como intermediário entre a aplicação do cliente e a SEFAZ. O processo de desenvolvimento, que inclui a geração de XML e a comunicação com a SEFAZ, pode ser demorado. Portanto, surgiu a necessidade de criar uma API para facilitar essa operação. Para esse fim, foi desenvolvida uma API utilizando a plataforma Node.js, com uma arquitetura REST. Essa API possibilita que toda a comunicação do sistema ocorra por meio de JSON.

É esperado que com a API seja possível facilitar o desenvolvimento da parte fiscal desses ERPs e também sua manutenção, uma vez que com certa frequência existem atualizações de layout nos documentos fiscais. Para o desenvolvimento vamos utilizar as abordagens da arquitetura limpa, e tecnologias como TypeScript, Node.js, Prisma ORM, com o objetivo de criar uma aplicação robusta para atender todos os processos da emis-

são dos documentos fiscais e realizar os cálculos de impostos, como, PIS, COFINS, ICMS E IPI.

Capítulo 2

Referencial Teórico

O propósito deste capítulo é fornecer o referencial teórico que serviu como base para o desenvolvimento da aplicação de consumo de documentos fiscais. Esta aplicação foi construída utilizando tecnologias contemporâneas, como Node.js, Prisma ORM e TypeScript. A finalidade principal da aplicação desenvolvida neste trabalho é a realização de cálculos de impostos para um software de gestão, bem como a emissão de documentos fiscais.

2.1 Trabalhos relacionados

Esta seção visa apresentar alguns trabalhos relacionados realizados com linguagens e frameworks semelhantes com este trabalho. Thiago Vieira Puluceno (2012) em sua tese realizou um estudo de caso em uma API REST utilizando Node.js e comparou a aplicação com uma implementação idêntica desenvolvida na linguagem JAVA. Seus resultados foram muito interessantes, os comportamentos no geral foram muito parecidos com algumas diferenças, como o consumo de memoria que foi muito inferior por parte do Node.js e a sua capacidade de lidar com mais requisições por segundo e de resolvê-las com grande qualidade.

Bruno Rodrigues (2008) desenvolveu uma aplicação simples em JAVA para emissão de NF-e pouco tempo depois que se tornou obrigatória a emissão dos documentos fiscais, a aplicação foi desenvolvida ainda em seu primeiro layout, apesar de simples a aplicação conta com uma interface gráfica, consegue gerar e comunicar com o webservice da SEFAZ, e disponibila a opção para o usuário baixar o XML gerado.

Lucas Campos Jorge (2020) apresentou uma API REST para um sistema de monitoramento de redes ópticas, seu desenvolvimento se deu da necessidade de uma aplicação que centralizasse todos os recursos necessários para o gerenciamento das redes ópticas e que fosse confiável. Segundo o autor com base no design REST foi possível tornar API mais fácil de ser utilizada e aprimorada no futuro.

2.2 Procedimentos, materiais e equipamentos

No processo de desenvolvimento da tecnologia, dos endpoints e das regras relacionadas aos documentos fiscais, foram utilizadas diversas fontes de informação. Foram consultados os manuais e as notas técnicas disponíveis no site da SEFAZ e da Receita Federal para obter informações essenciais para o desenvolvimento da API. Além disso, foram analisadas as documentações das tecnologias utilizadas no desenvolvimento da API, como prisma, Node.js e TypeScript.

Também foi aproveitada a vasta quantidade de artigos online que abordam métodos e boas práticas para o desenvolvimento desse sistema. Esses recursos foram fundamentais para orientar e aprimorar o processo de desenvolvimento da solução.

O objetivo inicial da API é o desenvolvimento dos documentos fiscais NF-e, NFC-e e CT-e. A NF-e é um documento de existência eletrônica desenvolvido para documentar uma operação de circulação de mercadoria. Entrou em vigor em 2006 e veio para substituir até então as notas emitidas nos modelos 1 e 1A.

A NFC-e é também um documento fiscal apenas digital, mas para o consumidor, compar-

tilha da mesma plataforma da NF-e, com algumas diferenças, como a não obrigatoriedade do preenchimento dos dados do comprador.

A CT-e é um documento também digital, porem sua emissão é feita por transportadoras, o objetivo da CT-e é gerar um controle dos transportes de cargas de uma localidade a outra, no documento é também informado as passagens por cada estado na rota do transporte. No CT-e podemos colocar diversas NF-e de mercadorias que vão ser transportadas no veículo ao qual foi emitida a CT-e.

A seguir serão descritas algumas tecnologias utilizadas durante o desenvolvimento da API.

2.2.1 Visual Studio Code

O Visual Studio Code é um editor de código-fonte desenvolvido pela Microsoft para Windows, Linux e macOS. Ele inclui suporte para depuração, controle de versionamento Git incorporado, realce de sintaxe, complementação inteligente de código, snippets e refatoração de código. É um editor eficiente e de fácil utilização, contém diversas extensões da comunidade, sejam para realçar o código de diversas formas como acesso a FTP diretamente do editor. Pelo fato de ser gratuito, rápido e muito customizável foi o editor escolhido para o desenvolvimento.

2.2.2 TypeScript

É uma linguagem de programação de código aberto desenvolvida pela Microsoft. É um superconjunto sintático estrito de JavaScript e adiciona tipagem estática opcional à linguagem. O TypeScript foi utilizado no projeto para garantir a qualidade do código e reduzir erros comuns ao trabalhar com certos tipos de dados. É uma linguagem que vem sendo amplamente utilizada na construção de alguns frameworks como Angular, React e recentemente o VueJS, também faz parte do código fonte do próprio *Visual Studio Code*.

2.2.3 Node.js

O Node.js é um software de código aberto, multiplataforma, baseado no interpretador V8 do Google e que permite a execução de códigos JavaScript fora de um navegador web. Uma das principais vantagens do Node.JS é o fato de que uma única thread é capaz de executar o código da aplicação, isso pelo fato de que sua execução é orientada a eventos. Como sua construção é robusta não são exigidos muitos recursos para o seu funcionamento. É um software amplamente utilizado pela indústria em empresas, como LinkedIn, PayPal e NASA.

2.2.4 Banco de dados

Utilizou-se o MySQL versão 8.0.27, que é um SGBD (Sistema de gerenciamento de banco de dados), que usa a linguagem SQL como interface. O MySQL é um gerenciador de banco de dados com suporte a inúmeros sistemas operacionais e muitas linguagens tem suporte nativo a esse SGBD. Foi escolhido para o desenvolvimento devido ao seu desempenho e segurança nos dados armazenados, na construção da API serão armazenados alguns dados sensíveis e de interesse apenas do cliente que vai utilizar, como o próprio documento fiscal emitido.

2.2.5 PrismaORM

O Prisma é um ORM que ajuda os desenvolvedores a criar aplicações mais rapidamente, fornece suporte a vários gerenciadores de banco de dados, como PostgreSQL, MySQL, SQLite, MongoDB, entre outros. O prisma foi lançado em 2017, e apesar de ser relativamente novo é um ORM bem robusto e que se posiciona como uma alternativa diferente do restante dos ORMs do mercado, um dos principais diferenciais do prisma é seu arquivo de schema onde é possível configurar todo o banco de dados, cada tabela e seus relacionamento e utilizando ferramentas do próprio prisma é possível criara o

banco de dados a partir desse schema. Existe ainda a opção de criar esse schema com base em uma base de dados já existe. Além disso, o Prisma suporta as linguagens de programação Javascript e Typescript.

2.2.6 API REST

É uma *interface* de programação de aplicações de transferência de estado representacional, comumente utilizada como arquitetura de transmissão de dados e mensagens para arquiteturas cliente-servidor e microserviço. Essa *interface* permite uma comunicação confiável e de alta performance em escala. Um dos seus benefícios é que você pode implementá-la e modifica-la com muita facilidade. Dentre as suas vantagens temos a separação entre o front-end e o back-end, essa separação torna mais ágil o desenvolvimento da aplicação, por utilizar o JSON como padrão de suas requisições torna o sistema multiplataforma uma vez que uma grande quantidade de aplicações consegue lidar com arquivos no formato JSON.

2.2.7 Insomnia

O Insomnia REST Client é uma ferramenta de teste de API que suporta aplicativos REST. Como a API vai ser desenvolvida apesar no back-end é primordial uma ferramenta para testarmos as requisições nos *end points*, a ferramenta foi escolhida pela sua facilidade de uso, tem uma interface muito simples e de fácil utilização, é uma ferramenta rápida e com uma boa performance.

2.2.8 Docker

Docker é um conjunto de produtos de plataforma como serviço que usam virtualização de nível de sistema operacional para entregar software em pacotes chamados contêineres. Os contêineres são isolados uns dos outros e agrupam seus próprios softwa-

res, bibliotecas e arquivos de configuração. Essa plataforma foi amplamente utiizada no desnvolvimento da API para facilitar no momento de levantar a API e para padronizar o ambiente de desenvolvimento.

2.2.9 Azure Data Studio

É uma ferramenta de análise de dados moderna, híbrida, multiplataforma e de software livre, projetada para simplificar o panorama de dados. O aplicativo foi escolhido para gerenciar o banco de dados durante o desenvolvimento. É um software rápido e de interface muito simples, é um software recomendado pela própria desenvolvedora para projetos em que seja necessário realizar buscas e inserções de forma rápida diretamente no banco de dados.

2.2.10 Jest

É um framework de teste em JavaScript projetado para garantir a correção de qualquer código JavaScript. Esse framework foi escolhido pela sua simplicidade de configurar e por seu desempenho na realização dos testes. Com ele podemos verificar o *coverage* do código e com essas informações planejar novos testes para deixar o código menos suscetível a erros no futuro.

Capítulo 3

Materiais e métodos

Neste capítulo, serão abordados os recursos e as estratégias essenciais para iniciar o desenvolvimento do projeto, incluindo a modelagem do sistema e do banco de dados, bem como a descrição dos métodos empregados na implementação dos algoritmos da aplicação. Este capítulo também apresenta os materiais e métodos utilizados ao longo deste trabalho.

3.1 Casos de uso

Nesta sessão é apresentado os casos de uso do sistema, onde é demonstrada cada ação que o usuário e o cliente poderá realizar na API.

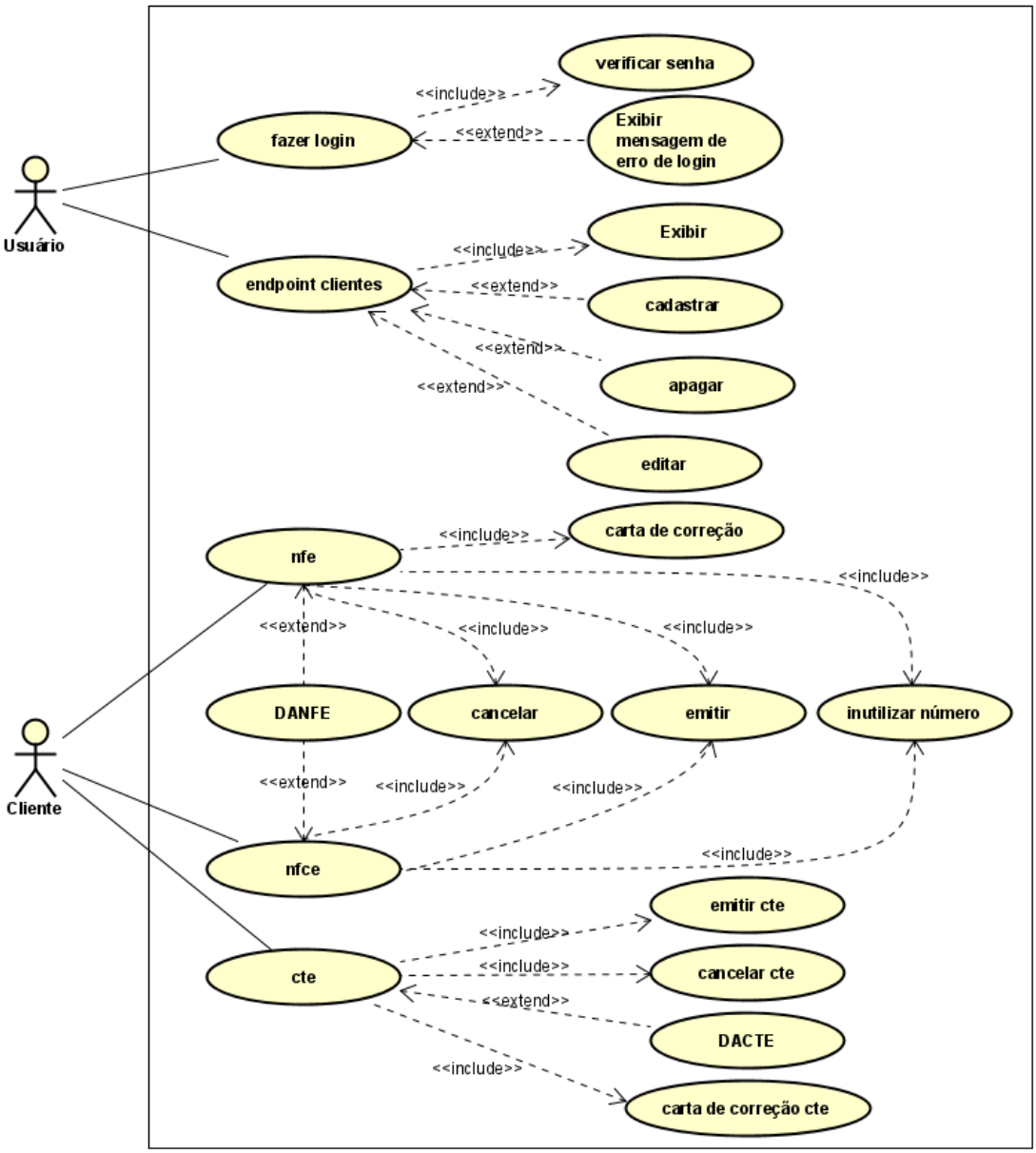


Figura 3.1: Caso de uso geral do sistema

Nome do Caso de Uso	Fazer login
Objetivo	Permitir o usuário acessar o sistema de gerenciamento.
Ator(es)	Usuário.
Pré-Condição	O usuário se autenticar no sistema de gerenciamento.
Cenário Principal	O ator preenche os dados de cadastro e confirma.
Cenário Alternativo.	Caso o usuário preencha os dados de forma errada, não terá acesso ao sistema de gerenciamento.

Tabela 3.1: Caso de uso para fazer login.

Nome do Caso de Uso	Endpoint cliente.
Objetivo	Permitir que o usuário visualize, altere, cadastre e exclua clientes do sistema de gerenciamento.
Ator(es)	Usuário.
Pré-Condição	1 - O usuário se autenticar no sistema de gerenciamento. 2 - O usuário requisitar o endpoint de clientes.
Cenário Principal	1 - O usuário visualiza os dados do cliente. 1 - O usuário cadastra um novo cliente. 2 - O usuário altera os dados de um cliente. 3 - O usuário exclui um cliente.
Cenário Alternativo.	Não realizar ação.

Tabela 3.2: Caso de uso para o endpoint de clientes.

Nome do Caso de Uso	Cálculo de impostos.
Objetivo	Permitir que o cliente realize cálculos tributários.
Ator(es)	Cliente.
Pré-Condição	1 - O cliente ser cadastrado e ter uma hash de acesso a API. 2 - O cliente requisitar o endpoint de cálculo de impostos.
Cenário Principal	1 - O cliente envia os dados para o cálculo do imposto.
Cenário Alternativo.	Não realizar ação.

Tabela 3.3: Caso de uso para o endpoint de cálculo de impostos.

Nome do Caso de Uso	NF-e
Objetivo	Permitir ao cliente emitir, cancelar, gerar carta de correção, inutilizar uma numeração e gerar o DANFE.
Ator(es)	Cliente.
Pré-Condição	1 - O cliente ser cadastrado e ter uma hash de acesso a API. 2 - O cliente requisitar o endpoint da NF-e.
Cenário Principal	1 - O cliente envia os dados para a emissão da NF-e. 2 - O cliente envia os dados para cancelamento da NF-e. 3 - O cliente envia os dados para carta de correção. 4 - O cliente envia os número para inutilizar. 5 - O cliente envia a NF-e que deseja gerar DANFE.
Cenário Alternativo.	Não realizar ação.

Tabela 3.4: Caso de uso para o endpoint da NF-e.

Nome do Caso de Uso	NFC-e
Objetivo	Permitir ao cliente emitir, cancelar, inutilizar uma numeração e gerar o DANFE.
Ator(es)	Cliente.
Pré-Condição	1 - O cliente ser cadastrado e ter uma hash de acesso a API. 2 - O cliente requisitar o endpoint da NFC-e.
Cenário Principal	1 - O cliente envia os dados para a emissão da NFC-e. 2 - O cliente envia os dados para cancelamento da NFC-e. 3 - O cliente envia os número para inutilizar. 4 - O cliente envia a NFC-e que deseja gerar DANFE.
Cenário Alternativo.	Não realizar ação.

Tabela 3.5: Caso de uso para o endpoint da NFC-e.

Nome do Caso de Uso	CT-e
Objetivo	Permitir ao cliente emitir, cancelar, gerar carta de correção e gerar o DACTE.
Ator(es)	Cliente.
Pré-Condição	1 - O cliente ser cadastrado e ter uma hash de acesso a API. 2 - O cliente requisitar o endpoint da CT-e.
Cenário Principal	1 - O cliente envia os dados para a emissão da CT-e. 2 - O cliente envia os dados para cancelamento da CT-e. 3 - O cliente envia os dados para carta de correção. 4 - O cliente envia a CT-e que deseja gerar DACTE.
Cenário Alternativo.	Não realizar ação.

Tabela 3.6: Caso de uso para o endpoint da CT-e.

3.2 Modelagem do banco de dados

Nesta sessão é apresentado a modelagem do banco de dados utilizado neste trabalho.

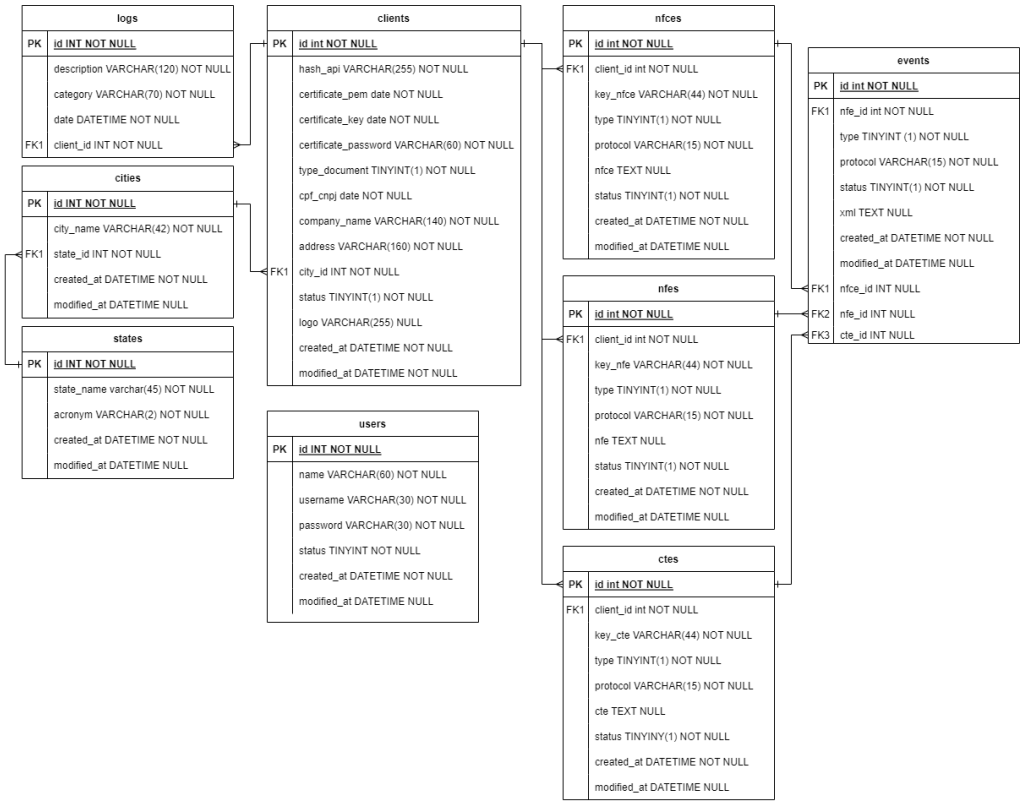


Figura 3.2: Diagrama entidade-relacionamento.

A figura 3.2 representa o diagrama Entidade-Relacionamento que foi implementado no sistema de gerenciamento.

Nome da tabela	Usuários
Objetivo	Utilizada para armazenamento de usuários que utilizarão a parte administrativa da API.

Tabela 3.7: Descrição da tabela de usuários.

Nome da tabela	Estados
Objetivo	Utilizada para armazenamento dos estados.
Relacionamento	1:N

Tabela 3.8: Descrição da tabela de estados.

Nome da tabela	Cidades
Objetivo	Utilizada para armazenamento das cidades.
Relacionamento	N:1

Tabela 3.9: Descrição da tabela de cidades.

Nome da tabela	Clientes
Objetivo	Utilizada para armazenamento dos clientes que serão adicionados pelo usuário e utilizarão as funcionalidades da API.
Relacionamento	1:N

Tabela 3.10: Descrição da tabela de clientes.

Nome da tabela	Logs
Objetivo	Utilizada para armazenamento de atividades que serão realizadas pelo cliente.
Relacionamento	1:N

Tabela 3.11: Descrição da tabela de logs.

Nome da tabela	NFes
Objetivo	Utilizada para armazenamento de NF-e que serão emitidas pelo cliente.
Relacionamento	1:N

Tabela 3.12: Descrição da tabela de nfes.

Nome da tabela	NFCes
Objetivo	Utilizada para armazenamento de NFC-e que serão emitidas pelo cliente.
Relacionamento	1:N

Tabela 3.13: Descrição da tabela de nfces.

Nome da tabela	CTes
Objetivo	Utilizada para armazenamento de CT-e que serão emitidas pelo cliente.
Relacionamento	1:N

Tabela 3.14: Descrição da tabela de ctes.

Nome da tabela	Eventos
Objetivo	Utilizada para armazenar os eventos da NF-e, NFC-e e CT-e, tais como: carta de correção, inutilização e cancelamento.
Relacionamento	1:N

Tabela 3.15: Descrição da tabela de eventos.

3.3 Desenvolvimento

Nesta sessão é apresentado o desenvolvimento do sistema, conceituando alguns pontos do código da aplicação, referenciando-os de forma sucinta para melhor entendimento do projeto.

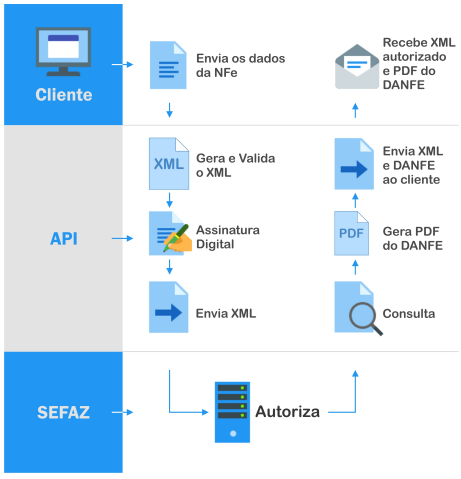


Figura 3.3: Fluxo dos dados.

Na figura 3.3 é apresentado o fluxo de dados da API, onde ela recebe do cliente os dados do documento fiscal, gera e valida os dados, assina com o certificado digital o documento e faz o envio para a SEFAZ. Uma vez autorizada pela SEFAZ, a API realiza a consulta do documento, gera o DANFE e retorna para o cliente o XML e o DANFE.

Na figura 3.4 temos um exemplo de solicitação para a emissão de um documento fiscal junto à SEFAZ. Essa solicitação é feita de forma assíncrona. Primeiramente, enviamos os dados à SEFAZ e, em resposta, recebemos um protocolo. Esse protocolo é essencial, pois será utilizado para futuras consultas no webservice da SEFAZ a fim de recuperar os documentos fiscais emitidos.

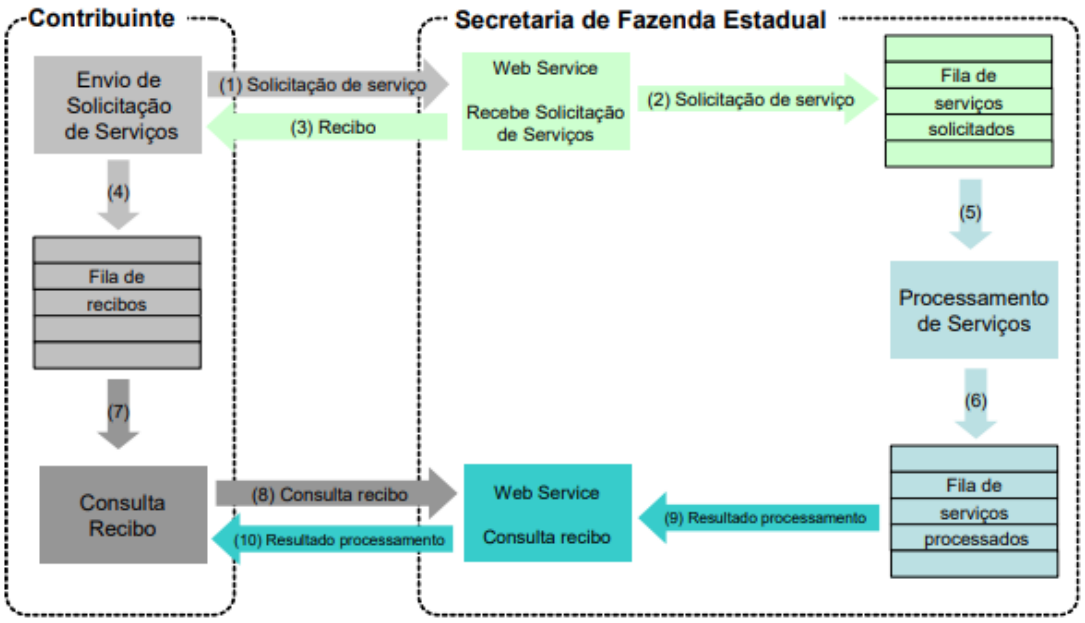


Figura 3.4: Exemplo de requisição. Fonte: SEFAZ

Durante o desenvolvimento da API, foi adotado a Arquitetura Limpa, uma abordagem proposta por Robert C. Martin, que visa padronizar e organizar o código, promovendo sua reusabilidade e independência de tecnologia. Esta arquitetura é baseada nos princípios SOLID apresentados em seu livro "Clean Architecture: A Craftsman's Guide to Software Structure". Durante a implementação também foram utilizados os princípios do SOLID.

Na figura 3.5 é apresentado a essência da arquitetura limpa, onde as setas entre os círculos exemplificam a direção por onde a informação deve correr dentro da arquitetura. Essa abordagem tem como objetivo tornar a aplicação mais robusta e com um padrão de comunicação bem definido, onde cada camada se comunica com camadas especificas.

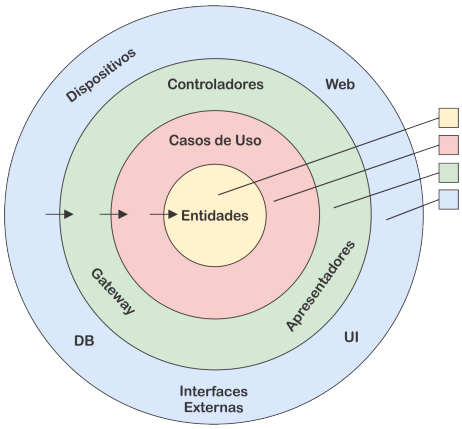


Figura 3.5: A arquitetura limpa. Fonte: Martin (2019)

Para manter essa estrutura utilizamos o SOLID, que são 5 princípios baseados em POO para organização da aplicação, são eles:

Single Responsiblity Principle (Princípio da responsabilidade única): Esse princípio especifica que uma classe deve ser especializada em um único assunto e possuir apenas uma responsabilidade dentro do software, ou seja, a classe deve ter uma única tarefa ou ação para executar.

Open-Closed Principle (Princípio Aberto-Fechado): Objetos ou entidades devem estar abertos para extensão, mas fechados para modificação, portanto, quando novos comportamentos e recursos precisam ser adicionados a aplicação, devemos estender e não alterar o código fonte original.

Liskov Substitution Principle (Princípio da substituição de Liskov): Essa regra diz que se você tem duas classes e ao passá-las para um código, nada precisa ser alterado nesse código, então elas são subtipos uma da outra.

Interface Segregation Principle (Princípio da Segregação da Interface): Essa regra define que um classe não deve ser forçada a implementar interfaces e métodos que

não irão utilizar, definindo assim que é melhor criar uma interface mais especifica ao invés de ter uma única interface genérica.

Dependency Inversion Principle (Princípio da inversão da dependência): Segundo Robert C. Martin podemos definir esse principio como, módulos de alto nível não devem depender de módulos de baixo nível. Ambos devem depender da abstração. E a abstração não deve depender de detalhes. Detalhes devem depender de abstrações.

No exemplo da figura 3.6, é demonstrado a estrutura utilizada para o desenvolvimento do projeto. A API leva em consideração os conceitos da arquitetura limpa, e em seu desenvolvimento quando utilizado bibliotecas de terceiros, foram criados adapters para desacoplar a dependência da API dessas bibliotecas e tornar fácil a manutenção futuramente.

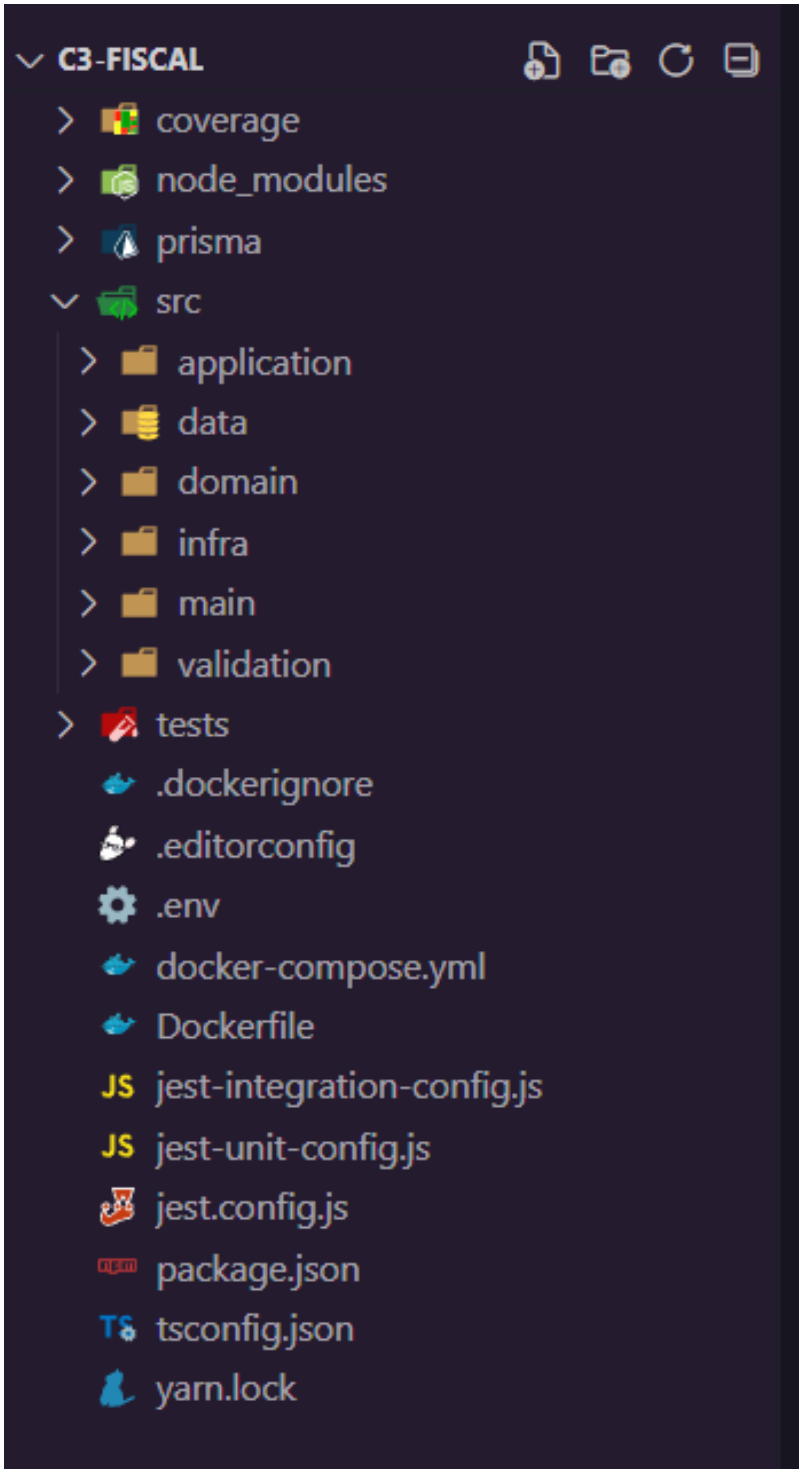


Figura 3.6: Estrutura do projeto.

Endereço	Verbo HTTP	Nome	Funcionalidade
/user/login	POST	Login API	Consulta se o usuário e senha informados podem acessar a API.
/clients	GET	Listar Clientes	Retorna a lista com os clientes cadastrados.
/client/	POST	Cadastrar Cliente	Endpoint para cadastro de novo cliente.
/client/	PUT	Alterar Cliente	Endpoint para alterar os dados do cliente.
/client/	DELETE	Delete Cliente	Endpoint para deletar um cliente.
/tax-calculation/icms	POST	Calcula ICMS	Endpoint para o cálculo do imposto ICMS.
/tax-calculation/pis	POST	Calcula PIS	Endpoint para o cálculo do imposto PIS.
/tax-calculation/cofins	POST	Calcula COFINS	Endpoint para o cálculo do imposto COFINS.
/tax-calculation/ipi	POST	Calcula IPI	Endpoint para o cálculo do imposto IPI.
/nfe/send	POST	Emitir NF-e	Endpoint para realizar a emissão da NF-e.
/nfe/event/correction-letter	POST	Carta Correção NF-e	Endpoint para gerar a carta de correção da NF-e.
/nfe/event/cancellation	POST	Cancelar NF-e	Endpoint para realizar o cancelamento da NF-e.
/nfe/event/unusability	POST	Inutilizar Númeração NF-e	Endpoint para realizar a inutilização de uma númeração da NF-e.
/nfce/send	POST	Emitir NFC-e	Endpoint para realizar a emissão da NFC-e
/nfce/event/cancellation	POST	Cancelar NFC-e	Endpoint para realizar o cancelamento da NFC-e
/nfce/event/unusability	POST	Inutilizar Númeração NFC-e	Endpoint para realizar inutilização de uma númeração da NFC-e
/cte/send	POST	Emitir CT-e	Endpoint para realizar a emissão da CT-e
/cte/event/cancellation	POST	Cancelar CT-e	Endpoint para cancelar a CT-e
/cte/event/correction-letter	POST	Carta Correção CT-e	Endpoint para realizar a carta de correção da CT-e

Tabela 3.16: Endpoints da API.

Na tabela 3.16 é demonstrado todos os endpoints disponibilizados na API. Foi amplamente utilizado os métodos HTTP para definir as ações do usuário, como GET, POST, PUT e DELETE, que é um dos princípios do REST.

Capítulo 4

Resultados

Este capítulo apresenta os resultados da implementação na prática do projeto desenvolvido utilizando as ferramentas do capítulo 3.

Evento de emissão

No exemplo da Listagem 4.1, ilustramos uma solicitação em formato JSON que o cliente envia à API para iniciar o processo de emissão de um documento fiscal. Para a emissão do documento é necessário o envio de certos dados, como informações sobre a nota, remetente, destinatário, produtos, tributação, os totais dos valores e pagamento. Em seguida na Listagem 4.2 descrevemos a resposta da API, que contém as informações detalhadas do documento fiscal emitido.

Listing 4.1: Exemplo de requisição em JSON para emissão de documento fiscal.

```
{
  "infNFe":{
    "ide": {
      "cUF": 51
      "cNF": 22123,
      "natOp": "VENDA DE MERCADORIA ADQUIRIDA DE TERCEIROS",
      "mod": 55,
      "serie": "001",
      "nNF": "001",
      "dhEmi": "2023-08-26T20:10:29.420Z",
      "dhSaiEnt": "2023-08-26T20:10:29.420Z",
      "tpNF": 1,
      "idDest": 2,
      "cMunFG": 4106902,
      "tpImp": 1,
      "tpEmis": 1,
      "cDV": 3,
      "tpAmb": 2,
      "finNFe": 1,
      "indFinal": 0,
      "indPres": 9,
      "indIntermed": 0
    },
    "emit": {
      "xNome": "TESTE EMPRESA LTDA",
      "IE": "54226455231",
      "CNPJ": "17415382000102",
      "iest": null,
      "enderEmit": {
        "xLgr": "Rua de teste",
        "nro": "684",
        "xBairro" : "Centro Sul",
        "xCpl": "",
        "xMun": "Sorriso",
```

```
        "UF": "MT",
        "CEP": "78892096",
        "fone": "(66) 99999-9999",
        "email": "teste@teste.com"
      }
    },
    "dest": {
      "xNome": "Teste Destinatario",
      "CPF": "00000000000",
      "IE": "52451644",
      "enderDest": {
        "xLgr": "Teste de rua 2",
        "nro": "489",
        "xMun": "Sorriso",
        "UF": "MT",
        "xBairro" : "Residencial de Teste",
        "CEP": "78892096",
        "fone": "(66) 99888-8888"
      }
    },
    "det": [
      {
      "prod": {
        "cProd": "90",
        "cEAN": "SEM GTIN",
        "xProd": "TESTE DE PRODUTO 1",
        "NCM": "87112010",
        "CFOP": "4102",
        "uCom": "UN",
        "qCom": 1,
        "vUnCom": 10000,
        "vProd": "10000",
        "cEANTrib": "SEM GTIN",
        "uTrib": "UN",
        "qTrib": 1,
        "vUnTrib": 10000,
        "indTot": 1
      },
      "imposto": {
        "ICMS": {
          "ICMS20": {
            "orig": "1",
            "CST": "20",
            "modBC": "3",
            "pRedBC": "95.00",
            "vBC": "500",
            "pICMS": "11.00",
            "vICMS": "55.00"
          }
```

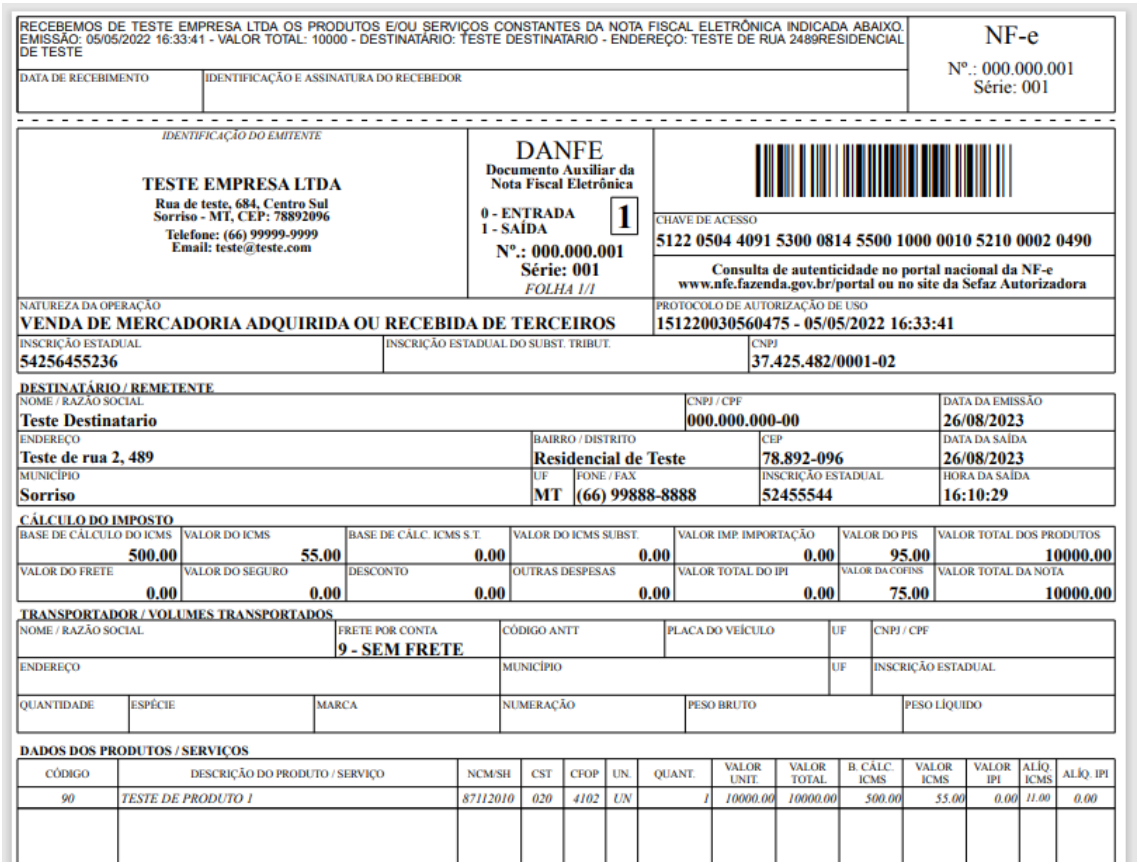
```
        },
        "PIS": {
          "PISAliq": {
            "CST": "01",
            "vBC": "10000",
            "pPIS": "0.95",
            "vPIS": "95.00"
          }
        },
        "COFINS": {
          "COFINSAliq":{
            "CST": "01",
            "vBC": "10000",
            "pPIS": "0.75",
            "vPIS": "75.00"
            }
          }
        }
      }
    ],
    "total": {
      "ICMSTot": {
        "vBC": "500",
        "vICMS": "55.00",
        "vBCST": "0.00",
        "vST": "0.00",
        "vII": "0.00",
        "vPIS": "95.00",
        "vCOFINS": "75.00",
        "vIPI": "0.00",
        "vProd": "10000",
        "vFrete": "0.00",
        "vSeg": "0.00",
        "vDesc": "0.00",
        "vOutro": "0.00",
        "vNF": "10000"
      }
    },
    "transp": {
      "modFrete": "9"
    },
    "pag": {
      "detPag": {
        "indPag": 1,
        "tPag": 15,
        "vPag": 10000
      }
    },
    "infAdic": {
```

```
133         "infCpl": "Teste com as informacoes complementares"
134       }
135     }
136 }
```

Listing 4.2: Exemplo de retorno em JSON da emissão do documento fiscal.

```
{
  "cStat": 100,
  "protocol": 121210000119301,
  "key": 51130132957668000115550010000000051647210112,
  "xMotivo": "Autorizado o uso da NF-e",
  "danfe": "306dce99a96e5da16efb.pdf"
}
```

No Figura 4.1, demonstramos um DANFE que é o documento auxiliar a NF-e, que é gerado após a emissão da NF-e. Já na Figura 4.2 temos um exemplo de DACTE, que é o documento auxiliar a CT-e.



RECEBEMOS DE TESTE EMPRESA LTDA OS PRODUTOS E/OU SERVIÇOS CONSTANTES DA NOTA FISCAL ELETRÔNICA INDICADA ABAIXO. EMISSÃO: 05/05/2022 16:33:41 - VALOR TOTAL: 10000 - DESTINATÁRIO: TESTE DESTINATARIO - ENDEREÇO: TESTE DE RUA 2489RESIDENCIAL DE TESTE

DATA DE RECEBIMENTO | IDENTIFICAÇÃO E ASSINATURA DO RECEBEDOR

NF-e
Nº.: 000.000.001
Série: 001

IDENTIFICAÇÃO DO EMITENTE

TESTE EMPRESA LTDA
Rua de teste, 684, Centro Sul
Sorriso - MT, CEP: 78892096
Telefone: (66) 99999-9999
Email: teste@teste.com

DANFE
Documento Auxiliar da Nota Fiscal Eletrônica
0 - ENTRADA
1 - SAÍDA
1
Nº.: 000.000.001
Série: 001
FOLHA 1/1

CHAVE DE ACESSO
5122 0504 4091 5300 0814 5500 1000 0010 5210 0002 0490
Consulta de autenticidade no portal nacional da NF-e www.nfe.fazenda.gov.br/portal ou no site da Sefaz Autorizadora

NATUREZA DA OPERAÇÃO	PROTOCOLO DE AUTORIZAÇÃO DE USO
VENDA DE MERCADORIA ADQUIRIDA OU RECEBIDA DE TERCEIROS	151220030560475 - 05/05/2022 16:33:41

INSCRIÇÃO ESTADUAL	INSCRIÇÃO ESTADUAL DO SUBST. TRIBUT.	CNPJ
54256455236		37.425.482/0001-02

DESTINATÁRIO / REMETENTE

NOME / RAZÃO SOCIAL			CNPJ / CPF	DATA DA EMISSÃO
Teste Destinatario			000.000.000-00	26/08/2023
ENDEREÇO	BAIRRO / DISTRITO		CEP	DATA DA SAÍDA
Teste de rua 2, 489	Residencial de Teste		78.892-096	26/08/2023
MUNICÍPIO	UF	FONE / FAX	INSCRIÇÃO ESTADUAL	HORA DA SAÍDA
Sorriso	MT	(66) 99888-8888	52455544	16:10:29

CÁLCULO DO IMPOSTO

BASE DE CÁLCULO DO ICMS	VALOR DO ICMS	BASE DE CÁLC. ICMS S.T.	VALOR DO ICMS SUBST.	VALOR IMP. IMPORTAÇÃO	VALOR DO PIS	VALOR TOTAL DOS PRODUTOS
500.00	55.00	0.00	0.00	0.00	95.00	10000.00
VALOR DO FRETE	**VALOR DO SEGURO**	**DESCONTO**	**OUTRAS DESPESAS**	**VALOR TOTAL DO IPI**	**VALOR DA COFINS**	**VALOR TOTAL DA NOTA**
0.00	0.00	0.00	0.00	0.00	75.00	10000.00

TRANSPORTADOR / VOLUMES TRANSPORTADOS

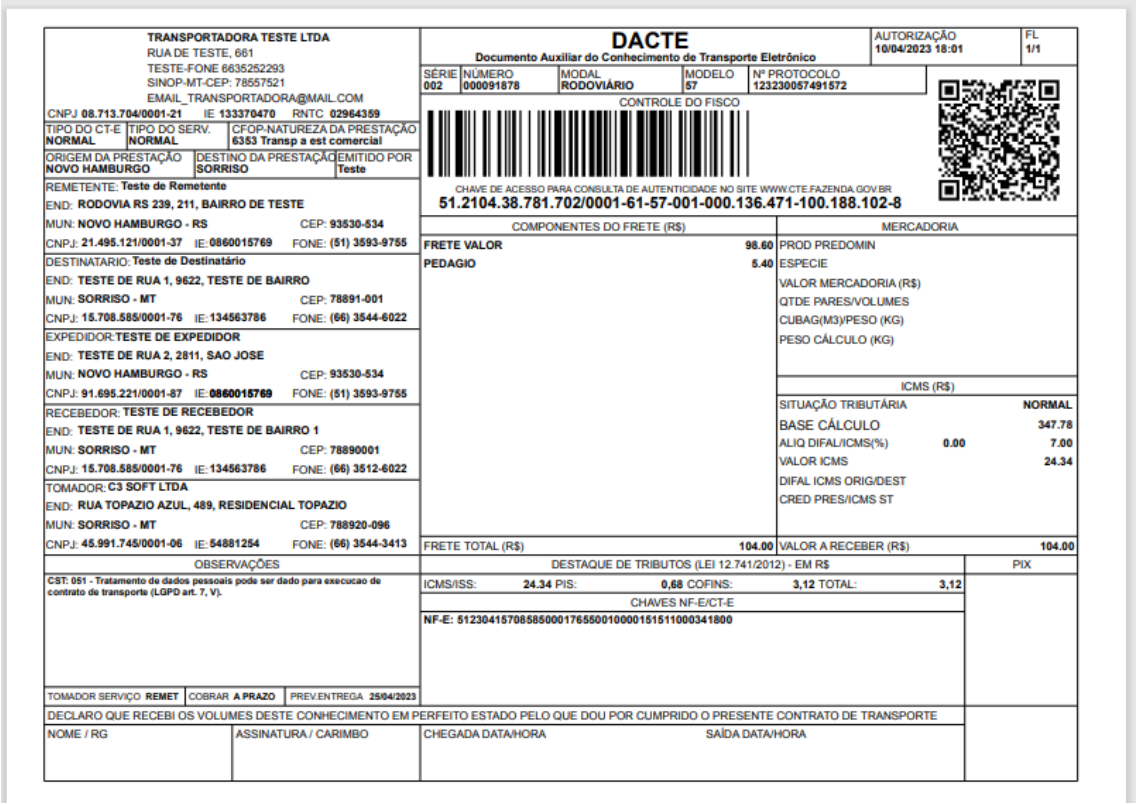
NOME / RAZÃO SOCIAL	FRETE POR CONTA	CÓDIGO ANTT	PLACA DO VEÍCULO	UF	CNPJ / CPF
	9 - SEM FRETE				
ENDEREÇO		MUNICÍPIO		UF	INSCRIÇÃO ESTADUAL

QUANTIDADE	ESPÉCIE	MARCA	NUMERAÇÃO	PESO BRUTO	PESO LÍQUIDO

DADOS DOS PRODUTOS / SERVIÇOS

CÓDIGO	DESCRIÇÃO DO PRODUTO / SERVIÇO	NCM/SH	CST	CFOP	UN.	QUANT.	VALOR UNIT.	VALOR TOTAL	B. CÁLC. ICMS	VALOR ICMS	VALOR IPI	ALÍQ. ICMS	ALÍQ. IPI
90	TESTE DE PRODUTO 1	87112010	020	4102	UN	1	10000.00	10000.00	500.00	55.00	0.00	11.00	0.00

Figura 4.1: Exemplo de DANFE gerado pela API.



TRANSPORTADORA TESTE LTDA
RUA DE TESTE, 661
TESTE-FONE 6635252293
SINOP-MT-CEP: 78557521
EMAIL_TRANSPORTADORA@MAIL.COM
CNPJ 08.713.704/0001-21 IE 133370470 RNTC 02964359

TIPO DO CT-E	TIPO DO SERV.	CFOP-NATUREZA DA PRESTAÇÃO
NORMAL	NORMAL	6353 Transp a est comercial

ORIGEM DA PRESTAÇÃO	DESTINO DA PRESTAÇÃO	EMITIDO POR
NOVO HAMBURGO	SORRISO	Teste

REMETENTE: Teste de Remetente
END: RODOVIA RS 239, 211, BAIRRO DE TESTE
MUN: NOVO HAMBURGO - RS CEP: 93530-534
CNPJ: 21.495.121/0001-37 IE: 0860015769 FONE: (51) 3593-9755

DESTINATARIO: Teste de Destinatário
END: TESTE DE RUA 1, 9622, TESTE DE BAIRRO
MUN: SORRISO - MT CEP: 78891-001
CNPJ: 15.708.585/0001-76 IE: 134563786 FONE: (66) 3544-6022

EXPEDIDOR: TESTE DE EXPEDIDOR
END: TESTE DE RUA 2, 2811, SAO JOSE
MUN: NOVO HAMBURGO - RS CEP: 93530-534
CNPJ: 91.695.221/0001-87 IE: 0860015769 FONE: (51) 3593-9755

RECEBEDOR: TESTE DE RECEBEDOR
END: TESTE DE RUA 1, 9622, TESTE DE BAIRRO 1
MUN: SORRISO - MT CEP: 78890001
CNPJ: 15.708.585/0001-76 IE: 134563786 FONE: (66) 3512-6022

TOMADOR: C3 SOFT LTDA
END: RUA TOPAZIO AZUL, 489, RESIDENCIAL TOPAZIO
MUN: SORRISO - MT CEP: 788920-096
CNPJ: 45.991.745/0001-06 IE: 54881254 FONE: (66) 3544-3413

OBSERVAÇÕES
CST: 051 - Tratamento de dados pessoais pode ser dado para execucao de contrato de transporte (LGPD art. 7, V).

TOMADOR SERVIÇO	COBRAR	PREV.ENTREGA
REMET	A PRAZO	25/04/2023

DECLARO QUE RECEBI OS VOLUMES DESTE CONHECIMENTO EM PERFEITO ESTADO PELO QUE DOU POR CUMPRIDO O PRESENTE CONTRATO DE TRANSPORTE

NOME / RG	ASSINATURA / CARIMBO	CHEGADA DATA/HORA	SAÍDA DATA/HORA

DACTE
Documento Auxiliar do Conhecimento de Transporte Eletrônico

AUTORIZAÇÃO 10/04/2023 18:01 | FL 1/1

SÉRIE	NÚMERO	MODAL	MODELO	Nº PROTOCOLO
002	000091878	RODOVIÁRIO	57	123230057491572

CONTROLE DO FISCO

CHAVE DE ACESSO PARA CONSULTA DE AUTENTICIDADE NO SITE WWW.CTE.FAZENDA.GOV.BR
51.2104.38.781.702/0001-61-57-001-000.136.471-100.188.102-8

COMPONENTES DO FRETE (R$)	
FRETE VALOR	98.60
PEDAGIO	5.40

MERCADORIA	
PROD PREDOMIN	
ESPECIE	
VALOR MERCADORIA (R$)	
QTDE PARES/VOLUMES	
CUBAG(M3)/PESO (KG)	
PESO CÁLCULO (KG)	

ICMS (R$)

SITUAÇÃO TRIBUTÁRIA		NORMAL
BASE CÁLCULO		347.78
ALIQ DIFAL/ICMS(%)	0.00	7.00
VALOR ICMS		24.34
DIFAL ICMS ORIG/DEST		
CRED PRES/ICMS ST		

FRETE TOTAL (R$)	104.00
VALOR A RECEBER (R$)	104.00

DESTAQUE DE TRIBUTOS (LEI 12.741/2012) - EM R$
ICMS/ISS: 24.34 PIS: 0,68 COFINS: 3,12 TOTAL: 3,12

CHAVES NF-E/CT-E
NF-E: 51230415708585000176550010000151511000341800

PIX

Figura 4.2: Exemplo de DACTE gerado pela API.

Evento de cancelamento

No exemplo da Listagem 4.3, ilustramos uma solicitação em formato JSON que o cliente envia à API para cancelar um documento fiscal. Para esse endpoint é necessário o envio do tipo de ambiente da emissão, o modelo, a chave do documento fiscal emitido, o protocolo de emissão e a justificativa do cancelamento. Em seguida na Listagem 4.4 descrevemos a resposta da API, que contém as informações detalhadas do documento fiscal cancelado.

Listing 4.3: Exemplo de requisição em JSON para cancelamento de documento fiscal.

```
{
  "tpAmb": "2",
  "modelo": "55",
  "chNFe": "51230337665483200109551130000230011497104922",
  "nProt": "151210019764405",
  "xJust": "NFE EMITIDA INDEVIDAMENTE."
}
```

Listing 4.4: Exemplo de retorno em JSON do cancelamento de documento fiscal.

```
{
    "cStat": "135,
    "xMotivo": "Evento registrado e vinculado a NF-e"
}
```

Evento de inutilização

No exemplo da Listagem 4.5, ilustramos uma solicitação em formato JSON que o cliente envia à API para inutilizar um número de documento fiscal, esse serviço é utilizado quando por algum motivo algum documento é emitido pulando alguma númeração, diante disso é necessário inutilizar a númeração que foi pulada. Esse endpoint é disponibilizado para NF-e e NFC-e. Já a CT-e em seu último layout não disponibiliza essa opção. Para esse endpoint é necessário o envio do ano, tipo de ambiente, modelo, série, número inicial, número final e a justificativa. Em seguida na Listagem 4.5 descrevemos a resposta da API, que contém as informações sobre a inutilização requisitada.

Listing 4.5: Exemplo de requisição em JSON para inutilizar numeração de documento fiscal.

```
{
  "year": "2023",
  "tpAmb": "1",
  "modelo": "55",
  "serie": "1",
  "numeroInicial": "67",
  "numeroFinal": "67",
  "xJustificativa": "sistema pulou numeracao."
}
```

Listing 4.6: Exemplo de retorno em JSON da inutilização da numeração do documento fiscal.

```
{
  "cStat": "102",
  "xMotivo": "Inutilizacao de numero homologado"
}
```

Evento de carta de correção

No exemplo da Listagem 4.7, ilustramos uma solicitação em formato JSON que o cliente envia à API para a criação de uma carta de correção de um documento fiscal. Esse serviço é utilizado para corrigir alguma informação enviada erroneamente na emissão do documento, não sendo possível alteração de informações sobre tributação, produtos e correções cadastrais que implique mudança do destinatário ou remetente. Esse endpoint é disponibilizado para NF-e e CT-e apenas, a NFC-e não conta com essa opção. Para esse endpoint é necessário o tipo de ambiente, modelo, a chave do documento e a correção dos dados emitidos errados. Em seguida na Listagem 4.8 descrevemos a resposta da API, que contém as informações sobre a carta de correção requisitada.

Listing 4.7: Exemplo de requisição em JSON para carta de correção do documento fiscal.

```
{
  "tpAmb": "1",
  "modelo": "55",
  "chNFe": "51430437665232000109550010000000051895541246",
  "xCorrecao": "Alteracao de conteudo lancado errado"
}
```

Listing 4.8: Exemplo de retorno em JSON da carta de correção do documento fiscal.

```
{
  "cStat": 135,
  "xMotivo": "Evento registrado e vinculado a NF-e"
}
```

Cálculo de impostos

No exemplo da Listagem 4.9, ilustramos uma solicitação em formato JSON que o cliente envia à API para realizar o cálculo de alguns impostos, esse serviço é um auxílio da API, nele não existe comunicação com a sefaz nem tem valor fiscal. A API consegue realizar os cálculos de ICMS, PIS, COFINS e IPI. Em seguida na Listagem 4.10 descrevemos a resposta da API, que contém as informações sobre o imposto calculado pela API.

Listing 4.9: Exemplo de requisição em JSON para cálculo de imposto.

```
{
  "taxApplies": true,
  "productValue": "1520",
  "totalValue": "1630",
  "haveFreight": false,
  "freightValue": "",
  "haveSafe": false,
  "safeValue": "",
  "haveFCP": false,
  "pICMS": "2",
  "pFCP": ""
}
```

Listing 4.10: Exemplo de retorno em JSON para cálculo de imposto.

```
{
  "vBC": 1520,
  "pICMS": "2",
  "vICMS": 30.400000000000002,
  "pFCP": ""
}
```

Capítulo 5

Conclusão

Diante do cenário atual a API se mostra uma boa opção para empresas que desejam trabalhar com a emissão de NF-e, NFC-e e CT-e sem a necessidade do desenvolvimento e também da manutenção do software, uma vez que é recorrente a alteração no layout e das regras de validação por parte da SEFAZ na emissão destes documentos. Contudo, o estudo e desenvolvimento da aplicação não se encerraram, tendo como proposta implementar a MDF-e, que tem como objetivo, consolidar as informaçoes da NF-e e CT-e, que permite o rastreamento da circulação física da carga, identifica o responsável pelo transporte em cada trecho do transporte e regitra o momento de início e fim do transporte. Outro objetivo é implementar também futuramente a NFS-e, documento cuja responsabilidade até então era dos municípios, mas agora conta com um padrão nacional, e cada vez mais municípios tem aderido a esse novo padrão, com isso a API passaria também a realizar a emissão das notas de serviço prestados.

A API foi validada e está em funcionamento, servindo ao seu objetivo de cálculo de impostos e emissão de documentos em um sistema desenvolvido na linguagem C# para gestão empresarial. Mostrou-se efetiva em seu papel e reduziu o tempo de desenvolvimento e manutenção do software de gestão, uma vez que com a API foi possível abstrair suas funcionalidades da aplicação.

Referências Bibliográficas

Jorge, Lucas Campos (2023), 'Projeto e arquitetura de api rest para sistema de monitoramento de redes Ópticas', Disponível em: https://bdm.unb.br/bitstream/10483/27278/1/2020_LucasCamposJorge_tcc.pdf. Acesso em: 23 de setembro 2023.

Martin, Robert C. (2019), *Arquitetura limpa: o guia do artesão para estrutura e design de software*, Alta Books.

Node.js (2023), 'Documentação', Disponível em: https://redmonk.com/sogrady/2022/03/28/language-rankings-1-22/. Acesso em: 02 de outubro 2023.

Platform, WS (2023), 'Microsserviços x monolítica qual arquitetura e melhor para o meu negocio', Disponível em: https://cws.digital/insights/microsservicos-x-monolitica-qual-arquitetura-e-melhor-para-o-meu-negocio.pdf. Acesso em: 23 de setembro 2023.

PULUCENO, THIAGO VIEIRA (2023), 'Estudo de caso sobre uma api rest utilizando a abordagem de programaÇÃo orientada e eventos com a plataforma node.js', Disponível em: https://repositorio.ufsc.br/bitstream/handle/123456789/184658/DOCUMENTO%20TCC%20THIAGO%20VIEIRA%20PULUCENO%20-%2008138048.pdf. Acesso em: 30 de setembro 2023.

Rodrigues, Bruno (2023), 'Um protótipo para emissão e gerenciamento de notas fiscais eletrônicas', Disponível em: https://www.cin.ufpe.br/~tg/2008-1/br.pdf. Acesso em: 30 de setembro 2023.

SEFAZ (2023), ‘Documentação’, Disponível em: https://www.nfe.fazenda.gov.br/portal/listaConteudo.aspx?tipoConteudo=ndIjl+iEFdE=. Acesso em: 25 de setembro 2023.

Typescript (2023), ‘Documentação’, Disponível em: https://www.typescriptlang.org/docs/. Acesso em: 20 de setembro 2023.

(Jorge 2023) (Rodrigues 2023) (Martin 2019) (PULUCENO 2023) (Node.js 2023) (SEFAZ 2023) (Typescript 2023) (Platform 2023)

Anexo A

Termo de Autorização do Autor

UNIVERSIDADE FEDERAL DE MATO GROSSO
BIBLIOTECA CENTRAL – BC/UFMT
BIBLIOTECA DIGITAL DE MONOGRAFIAS

BDM - UFMT

Termo de Autorização do Autor

Na qualidade de titular dos direitos de autor da publicação, autorizo a UFMT, por meio da Biblioteca Central, a disponibilizar, a partir desta data, na Biblioteca Digital de Trabalhos de Curso e Monografias de Especialização (ou em qualquer outro sistema informatizado/on-line de gestão de acervos, utilizado pela Instituição) o texto integral da obra abaixo citada, para fins de consulta, leitura, impressão e/ou download, de acordo com a **Lei nº 9.610/98**, a título de divulgação da produção científica brasileira, sem ressarcimento dos direitos autorais.

Declaro ainda estar ciente de que a mídia contendo o documento digital poderá ser descartada pela Biblioteca Central da UFMT após a inclusão do trabalho na Biblioteca Digital de Trabalhos de Curso e Monografias de Especialização da UFMT ou em outro sistema da Instituição.

1. Identificação do(a) Autor(a):

Nome:	
RG:	CPF:
E-mail:	
Telefone:	Tel. Celular:

* Em caso de trabalhos com autoria conjunta, como por exemplo, Trabalhos de Curso e Monografias de Especialização elaboradas por mais de um aluno, cada autor deverá preencher um termo, assinalando no campo específico que se trata de autoria conjunta.

2. Identificação da Monografia:

Categoria: ( ) Monografia de Graduação ( ) Monografia de Especialização
Autoria conjunta: ( ) Sim ( ) Não
Em caso de trabalho com autoria conjunta, listar os nomes dos demais autores:
Título:
Palavras-chave:
Departamento:
Curso:
Data de Apresentação/Defesa:
Orientador(a):

3. Tipo de Acesso ao Documento: ( ) Total ( ) Parcial*

Em caso de publicação parcial, especifique os capítulos a serem retidos: _____.

*A restrição poderá ser mantida por até 01 (um) ano a partir da data de autorização da publicação, desde que devidamente justificada. A extensão deste prazo requer justificativa junto à Biblioteca Central da UFMT. O resumo e os metadados ficarão sempre disponibilizados.

OBS.: Havendo concordância com a publicação eletrônica, mesmo com restrições temporárias de acesso, torna-se imprescindível o envio do **Trabalho** em formato digital (PDF) à Biblioteca Central da UFMT, lembrando que esta Unidade não efetuará quaisquer alterações no conteúdo dos arquivos recebidos.

Assinatura: _____.

Cuiabá, _____ de _____ de _____.