**Especificação de APIs e Integrações**

**1. Arquitetura Geral das APIs**

**1.1 Padrões e Convenções**

* **Protocolo**: REST com HTTP/2
* **Formato**: JSON (application/json)
* **Autenticação**: JWT + OAuth 2.0
* **Versionamento**: URL path (/api/v1/)
* **Rate Limiting**: Por usuário e endpoint
* **Documentação**: OpenAPI 3.0 (Swagger)

**1.2 Estrutura Base de Resposta**

interface APIResponse<T> {

success: boolean;

data?: T;

error?: {

code: string;

message: string;

details?: any;

};

pagination?: {

page: number;

per\_page: number;

total: number;

total\_pages: number;

};

meta?: {

timestamp: string;

request\_id: string;

version: string;

};

}

**2. API Core - Autenticação e Usuários**

**2.1 Autenticação**

# POST /api/v1/auth/login

summary: "Autenticação de usuário"

requestBody:

required: true

content:

application/json:

schema:

type: object

properties:

email:

type: string

format: email

password:

type: string

minLength: 8

mfa\_code:

type: string

description: "Código 2FA quando obrigatório"

required: [email, password]

responses:

200:

description: "Login realizado com sucesso"

content:

application/json:

schema:

type: object

properties:

access\_token:

type: string

refresh\_token:

type: string

expires\_in:

type: integer

user:

$ref: '#/components/schemas/User'

401:

description: "Credenciais inválidas"

423:

description: "Conta bloqueada por múltiplas tentativas"

**2.2 Gestão de Usuários**

# GET /api/v1/users/profile

summary: "Perfil do usuário autenticado"

security:

- bearerAuth: []

responses:

200:

content:

application/json:

schema:

$ref: '#/components/schemas/User'

# PUT /api/v1/users/profile

summary: "Atualizar perfil"

security:

- bearerAuth: []

requestBody:

content:

application/json:

schema:

type: object

properties:

name:

type: string

phone:

type: string

preferences:

type: object

properties:

notification\_channels:

type: array

items:

type: string

enum: [email, sms, push, whatsapp]

timezone:

type: string

language:

type: string

default: "pt-BR"

**3. API de Gestão de Casos**

**3.1 CRUD de Casos**

// POST /api/v1/cases

interface CreateCaseRequest {

title: string;

description: string;

client\_id: string;

area\_of\_law: string[];

court?: string;

process\_number?: string;

priority: 'low' | 'medium' | 'high' | 'urgent';

assigned\_lawyers: string[];

tags?: string[];

}

interface CaseResponse {

id: string;

title: string;

description: string;

client: ClientSummary;

area\_of\_law: string[];

status: 'active' | 'pending' | 'closed' | 'archived';

court?: string;

process\_number?: string;

priority: 'low' | 'medium' | 'high' | 'urgent';

assigned\_lawyers: LawyerSummary[];

created\_at: string;

updated\_at: string;

deadlines\_count: number;

documents\_count: number;

next\_deadline?: DeadlineSummary;

}

// GET /api/v1/cases?page=1&per\_page=20&status=active&area\_of\_law=civil

interface CasesListResponse {

cases: CaseResponse[];

pagination: PaginationInfo;

filters: {

available\_areas: string[];

available\_courts: string[];

available\_statuses: string[];

};

}

// GET /api/v1/cases/{case\_id}

interface CaseDetailResponse extends CaseResponse {

timeline: TimelineEvent[];

folders: CaseFolder[];

deadlines: Deadline[];

financial\_summary: FinancialSummary;

ai\_insights: AIInsight[];

}

**3.2 Gestão de Documentos do Caso**

# POST /api/v1/cases/{case\_id}/documents

summary: "Upload de documento"

parameters:

- name: case\_id

in: path

required: true

schema:

type: string

format: uuid

requestBody:

content:

multipart/form-data:

schema:

type: object

properties:

file:

type: string

format: binary

folder:

type: string

enum: [initial\_documents, evidence, court\_decisions, correspondence, internal\_memos]

document\_type:

type: string

enum: [petition, contract, email, decision, report, other]

description:

type: string

is\_confidential:

type: boolean

default: false

required: [file]

responses:

201:

description: "Documento enviado com sucesso"

content:

application/json:

schema:

type: object

properties:

id:

type: string

filename:

type: string

size:

type: integer

document\_type:

type: string

folder:

type: string

processing\_status:

type: string

enum: [pending, processing, completed, failed]

ocr\_extracted\_text:

type: string

nullable: true

ai\_classification:

type: object

properties:

confidence:

type: number

suggested\_type:

type: string

entities\_found:

type: array

items:

type: object

upload\_url:

type: string

description: "URL para download temporário"

# GET /api/v1/cases/{case\_id}/documents

summary: "Listar documentos do caso"

parameters:

- name: folder

in: query

schema:

type: string

- name: document\_type

in: query

schema:

type: string

**4. API de IA - Geração de Documentos**

**4.1 Geração de Petições**

// POST /api/v1/ai/generate/petition

interface GeneratePetitionRequest {

case\_id: string;

petition\_type: 'initial' | 'answer' | 'appeal' | 'motion';

template\_id?: string;

facts: {

parties: Party[];

chronology: ChronologyEvent[];

main\_facts: string;

damages?: DamageDescription;

};

legal\_grounds: {

primary\_laws: string[];

supporting\_jurisprudence?: string[];

legal\_theory: string;

};

requests: {

main\_request: string;

alternative\_requests?: string[];

provisional\_measures?: string[];

case\_value?: number;

};

preferences: {

tone: 'formal' | 'assertive' | 'conciliatory';

max\_pages: number;

include\_jurisprudence: boolean;

citation\_style: 'abnt' | 'juridica';

};

}

interface GeneratePetitionResponse {

generation\_id: string;

status: 'queued' | 'processing' | 'completed' | 'failed';

estimated\_time\_seconds: number;

document?: {

content: string;

structure: DocumentStructure;

citations: CitationInfo[];

word\_count: number;

page\_count: number;

confidence\_score: number;

warnings: string[];

};

ai\_metadata: {

model\_used: string;

processing\_time\_ms: number;

sources\_consulted: number;

tokens\_used: number;

};

}

// GET /api/v1/ai/generate/{generation\_id}/status

interface GenerationStatusResponse {

generation\_id: string;

status: 'queued' | 'processing' | 'completed' | 'failed';

progress\_percentage: number;

current\_step: string;

estimated\_remaining\_seconds: number;

result?: GeneratePetitionResponse['document'];

error\_message?: string;

}

**4.2 Validação de Citações**

# POST /api/v1/ai/validate/citations

summary: "Validar citações em texto jurídico"

requestBody:

content:

application/json:

schema:

type: object

properties:

text:

type: string

description: "Texto contendo citações a serem validadas"

validation\_level:

type: string

enum: [basic, complete, deep]

default: complete

description: |

- basic: Verifica se citação existe

- complete: Verifica existência + vigência

- deep: Análise completa + precedentes relacionados

required: [text]

responses:

200:

content:

application/json:

schema:

type: object

properties:

validation\_id:

type: string

overall\_accuracy:

type: number

description: "Percentual de citações válidas"

citations\_found:

type: integer

citations\_validated:

type: integer

results:

type: array

items:

type: object

properties:

citation\_text:

type: string

type:

type: string

enum: [legislation, jurisprudence, doctrine]

status:

type: string

enum: [valid, invalid, outdated, revoked, modified]

confidence:

type: number

official\_source:

type: string

format: uri

current\_text:

type: string

description: "Texto atual se houve modificação"

suggestions:

type: array

items:

type: string

related\_citations:

type: array

items:

type: string

**5. API de Prazos e Calendário**

**5.1 Cálculo de Prazos**

// POST /api/v1/deadlines/calculate

interface CalculateDeadlineRequest {

trigger\_event: {

type: 'publication' | 'service' | 'hearing' | 'decision' | 'custom';

date: string; // ISO 8601

description: string;

};

deadline\_type: {

category: 'response' | 'appeal' | 'compliance' | 'filing';

specific\_type: string; // ex: "contestacao", "apelacao\_civel"

legal\_basis?: string; // ex: "CPC art. 335"

};

process\_info: {

court: string;

procedure\_type: 'common' | 'summary' | 'special' | 'labor' | 'criminal';

area\_of\_law: string;

parties\_count?: number;

};

considerations?: {

include\_warnings: boolean;

check\_holidays: boolean;

consider\_court\_vacation: boolean;

};

}

interface CalculateDeadlineResponse {

calculation\_id: string;

deadline\_date: string; // ISO 8601

business\_days: number;

calendar\_days: number;

legal\_basis: {

primary\_rule: string; // ex: "CPC art. 335"

additional\_rules: string[];

court\_specific\_rules: string[];

};

calculation\_details: {

start\_date: string;

end\_date: string;

excluded\_days: {

date: string;

reason: string;

type: 'holiday' | 'weekend' | 'court\_vacation';

}[];

working\_days\_counted: number;

};

urgency\_level: 'low' | 'medium' | 'high' | 'critical';

warnings: string[];

reminders: {

advance\_days: number[];

channels: ('email' | 'sms' | 'push' | 'whatsapp')[];

};

}

// POST /api/v1/deadlines

interface CreateDeadlineRequest {

case\_id: string;

title: string;

description: string;

due\_date: string;

deadline\_type: string;

priority: 'low' | 'medium' | 'high' | 'urgent';

assigned\_to: string[];

legal\_basis?: string;

calculation\_id?: string; // Se veio de cálculo automático

reminders: {

days\_before: number[];

channels: string[];

};

tags?: string[];

}

**5.2 Calendário e Alertas**

# GET /api/v1/calendar/deadlines

summary: "Consultar prazos no calendário"

parameters:

- name: start\_date

in: query

required: true

schema:

type: string

format: date

- name: end\_date

in: query

required: true

schema:

type: string

format: date

- name: case\_ids

in: query

schema:

type: array

items:

type: string

- name: priority

in: query

schema:

type: array

items:

type: string

- name: assigned\_to

in: query

schema:

type: string

responses:

200:

content:

application/json:

schema:

type: object

properties:

events:

type: array

items:

type: object

properties:

id:

type: string

title:

type: string

description:

type: string

start\_date:

type: string

format: date-time

end\_date:

type: string

format: date-time

case\_info:

type: object

properties:

case\_id:

type: string

case\_title:

type: string

client\_name:

type: string

priority:

type: string

status:

type: string

enum: [pending, in\_progress, completed, overdue]

assigned\_lawyers:

type: array

items:

type: object

days\_until\_due:

type: integer

urgency\_color:

type: string

description: "Cor para exibição no calendário"

**6. API de Pesquisa Jurídica**

**6.1 Busca Avançada**

// POST /api/v1/legal-search/advanced

interface AdvancedSearchRequest {

query: {

text: string;

exact\_phrase?: string;

exclude\_words?: string;

synonyms\_enabled: boolean;

};

filters: {

sources: ('legislation' | 'jurisprudence' | 'doctrine')[];

date\_range?: {

start: string;

end: string;

};

courts?: string[];

areas\_of\_law?: string[];

document\_types?: string[];

rapporteur?: string;

importance\_level?: ('low' | 'medium' | 'high')[];

};

sort: {

field: 'relevance' | 'date' | 'court\_hierarchy' | 'citation\_count';

order: 'asc' | 'desc';

};

pagination: {

page: number;

per\_page: number; // max 50

};

options: {

include\_snippets: boolean;

highlight\_terms: boolean;

group\_similar: boolean;

include\_ai\_summary: boolean;

};

}

interface SearchResult {

id: string;

title: string;

summary: string;

source\_type: 'legislation' | 'jurisprudence' | 'doctrine';

court?: string;

date: string;

relevance\_score: number;

importance\_level: 'low' | 'medium' | 'high';

snippet: string;

highlighted\_terms: string[];

legal\_references: string[];

cited\_count: number;

official\_url: string;

document\_type: string;

metadata: {

rapporteur?: string;

case\_number?: string;

parties?: string[];

legal\_area: string[];

keywords: string[];

};

}

interface AdvancedSearchResponse {

results: SearchResult[];

pagination: PaginationInfo;

search\_metadata: {

query\_processed: string;

total\_results: number;

search\_time\_ms: number;

sources\_searched: string[];

filters\_applied: any;

};

ai\_insights?: {

summary: string;

key\_trends: string[];

related\_topics: string[];

suggested\_refinements: string[];

};

}

**6.2 Análise Jurisprudencial**

# POST /api/v1/legal-search/jurisprudence-analysis

summary: "Análise de tendências jurisprudenciais"

requestBody:

content:

application/json:

schema:

type: object

properties:

topic:

type: string

description: "Tema para análise"

courts:

type: array

items:

type: string

description: "Tribunais a serem analisados"

time\_period:

type: object

properties:

start\_year:

type: integer

end\_year:

type: integer

analysis\_depth:

type: string

enum: [overview, detailed, comprehensive]

default: detailed

required: [topic]

responses:

200:

content:

application/json:

schema:

type: object

properties:

analysis\_id:

type: string

topic:

type: string

period\_analyzed:

type: object

total\_decisions:

type: integer

trend\_analysis:

type: object

properties:

general\_trend:

type: string

enum: [stable, increasing, decreasing, fluctuating]

trend\_confidence:

type: number

key\_changes:

type: array

items:

type: object

properties:

year:

type: integer

change\_description:

type: string

significance:

type: string

enum: [low, medium, high]

court\_divergences:

type: array

items:

type: object

properties:

courts:

type: array

items:

type: string

divergence\_topic:

type: string

positions:

type: array

items:

type: object

key\_precedents:

type: array

items:

$ref: '#/components/schemas/SearchResult'

statistics:

type: object

properties:

favorable\_decisions:

type: integer

unfavorable\_decisions:

type: integer

neutral\_decisions:

type: integer

success\_rate:

type: number

**7. APIs de Integração Externa**

**7.1 Integração com Tribunais**

// POST /api/v1/integrations/courts/pje/search

interface PJeSearchRequest {

credentials: {

cpf: string;

password: string;

certificate?: string; // Base64 encoded certificate

};

search\_params: {

process\_number?: string;

party\_name?: string;

lawyer\_oab?: string;

court\_code: string;

date\_range?: {

start: string;

end: string;

};

};

options: {

include\_documents: boolean;

include\_movements: boolean;

max\_results: number;

};

}

interface CourtIntegrationResponse {

integration\_id: string;

status: 'pending' | 'processing' | 'completed' | 'error';

court\_system: 'pje' | 'esaj' | 'projudi' | 'eproc';

results?: {

processes\_found: number;

processes: CourtProcess[];

last\_sync: string;

warnings: string[];

};

error\_details?: {

error\_code: string;

error\_message: string;

retry\_possible: boolean;

};

}

interface CourtProcess {

number: string;

court: string;

class: string;

subject: string;

parties: ProcessParty[];

current\_status: string;

last\_movement: {

date: string;

description: string;

};

documents: ProcessDocument[];

movements: ProcessMovement[];

case\_value?: number;

distribution\_date: string;

judge: string;

}

// POST /api/v1/integrations/courts/automated-filing

interface AutomatedFilingRequest {

court\_system: 'pje' | 'esaj' | 'projudi';

credentials: CourtCredentials;

petition: {

document\_id: string;

petition\_type: string;

process\_number?: string; // Para petições intermediárias

attachments: string[]; // Document IDs

};

filing\_data: {

urgency: boolean;

confidential: boolean;

digital\_signature\_required: boolean;

court\_fees\_payment?: {

method: string;

reference: string;

};

};

confirmation\_required: boolean; // Se true, pausa para confirmação humana

}

**7.2 Integração com ERPs Jurídicos**

# POST /api/v1/integrations/erp/sync

summary: "Sincronização com ERP jurídico"

requestBody:

content:

application/json:

schema:

type: object

properties:

erp\_type:

type: string

enum: [astrea, themis, saj\_adv, advbox, custom]

connection\_config:

type: object

properties:

api\_url:

type: string

api\_key:

type: string

username:

type: string

additional\_params:

type: object

sync\_options:

type: object

properties:

entities:

type: array

items:

type: string

enum: [clients, cases, billing, calendar, documents]

direction:

type: string

enum: [import, export, bidirectional]

incremental:

type: boolean

description: "Se true, sincroniza apenas mudanças desde último sync"

conflict\_resolution:

type: string

enum: [overwrite\_local, overwrite\_remote, manual\_review]

required: [erp\_type, connection\_config]

responses:

202:

description: "Sincronização iniciada"

content:

application/json:

schema:

type: object

properties:

sync\_id:

type: string

status:

type: string

enum: [queued, running, completed, failed]

estimated\_duration:

type: integer

description: "Duração estimada em segundos"

# GET /api/v1/integrations/erp/sync/{sync\_id}

summary: "Status da sincronização"

responses:

200:

content:

application/json:

schema:

type: object

properties:

sync\_id:

type: string

status:

type: string

progress:

type: object

properties:

percentage:

type: integer

current\_step:

type: string

entities\_processed:

type: integer

entities\_total:

type: integer

results:

type: object

properties:

imported\_count:

type: integer

exported\_count:

type: integer

conflicts\_found:

type: integer

errors:

type: array

items:

type: object

conflicts:

type: array

items:

type: object

properties:

entity\_type:

type: string

entity\_id:

type: string

conflict\_type:

type: string

local\_version:

type: object

remote\_version:

type: object

resolution\_options:

type: array

items:

type: string

**8. Webhooks e Notificações**

**8.1 Sistema de Webhooks**

# POST /api/v1/webhooks

summary: "Configurar webhook"

requestBody:

content:

application/json:

schema:

type: object

properties:

url:

type: string

format: uri

events:

type: array

items:

type: string

enum: [

'case.created', 'case.updated', 'case.closed',

'deadline.created', 'deadline.approaching', 'deadline.overdue',

'document.uploaded', 'document.generated', 'document.approved',

'ai.generation.completed', 'ai.validation.failed',

'integration.sync.completed', 'integration.error'

]

filters:

type: object

properties:

case\_ids:

type: array

items:

type: string

user\_ids:

type: array

items:

type: string

priority\_levels:

type: array

items:

type: string

secret:

type: string

description: "Secret para validação HMAC"

active:

type: boolean

default: true

required: [url, events]

# Estrutura do payload do webhook

WebhookPayload:

type: object

properties:

id:

type: string

description: "ID único do evento"

event:

type: string

description: "Tipo do evento"

timestamp:

type: string

format: date-time

data:

type: object

description: "Dados específicos do evento"

user\_id:

type: string

description: "ID do usuário que triggou o evento"

signature:

type: string

description: "Assinatura HMAC-SHA256 para validação"

**8.2 API de Notificações**

// POST /api/v1/notifications/send

interface SendNotificationRequest {

recipients: {

user\_ids?: string[];

roles?: string[];

case\_members?: string; // case\_id

};

message: {

title: string;

body: string;

action\_url?: string;

action\_text?: string;

};

channels: ('push' | 'email' | 'sms' | 'whatsapp')[];

priority: 'low' | 'medium' | 'high' | 'urgent';

schedule?: {

send\_at: string; // ISO 8601

timezone: string;

};

metadata?: {

case\_id?: string;

deadline\_id?: string;

document\_id?: string;

category: string;

};

}

// GET /api/v1/notifications

interface NotificationListResponse {

notifications: {

id: string;

title: string;

body: string;

category: string;

priority: string;

read: boolean;

created\_at: string;

action\_url?: string;

metadata?: any;

}[];

pagination: PaginationInfo;

unread\_count: number;

}

**9. Rate Limiting e Quotas**

**9.1 Configuração de Rate Limits**

interface RateLimitConfig {

endpoint\_limits: {

'/api/v1/ai/generate/\*': {

requests\_per\_minute: 10;

requests\_per\_hour: 100;

burst\_limit: 20;

};

'/api/v1/legal-search/\*': {

requests\_per\_minute: 60;

requests\_per\_hour: 1000;

burst\_limit: 100;

};

'/api/v1/cases/\*': {

requests\_per\_minute: 100;

requests\_per\_hour: 5000;

burst\_limit: 200;

};

};

user\_tier\_multipliers: {

'solo': 1.0;

'small\_office': 2.0;

'medium\_office': 5.0;

'enterprise': 10.0;

};

quota\_limits: {

'ai\_documents\_per\_month': {

'solo': 50;

'small\_office': 200;

'medium\_office': 500;

'enterprise': -1; // unlimited

};

'storage\_gb': {

'solo': 10;

'small\_office': 50;

'medium\_office': 200;

'enterprise': 1000;

};

};

}

// Headers retornados em todas as respostas

interface RateLimitHeaders {

'X-RateLimit-Limit': string; // Limite por janela de tempo

'X-RateLimit-Remaining': string; // Requests restantes

'X-RateLimit-Reset': string; // Timestamp do reset

'X-RateLimit-Retry-After': string; // Segundos para retry (quando limitado)

'X-Quota-Limit': string; // Limite mensal

'X-Quota-Used': string; // Quota usada no mês

'X-Quota-Remaining': string; // Quota restante

}

**10. Monitoramento e Health Checks**

**10.1 Health Check Endpoints**

# GET /health

summary: "Health check básico"

responses:

200:

content:

application/json:

schema:

type: object

properties:

status:

type: string

enum: [healthy, degraded, unhealthy]

timestamp:

type: string

format: date-time

version:

type: string

uptime\_seconds:

type: integer

# GET /health/detailed

summary: "Health check detalhado (requer autenticação admin)"

responses:

200:

content:

application/json:

schema:

type: object

properties:

overall\_status:

type: string

services:

type: object

properties:

database:

type: object

properties:

status:

type: string

response\_time\_ms:

type: number

connection\_pool:

type: object

redis:

type: object

properties:

status:

type: string

response\_time\_ms:

type: number

memory\_usage:

type: string

ai\_services:

type: object

properties:

openai\_api:

type: object

local\_models:

type: object

external\_apis:

type: object

properties:

courts\_integration:

type: object

legal\_databases:

type: object

performance\_metrics:

type: object

properties:

avg\_response\_time\_ms:

type: number

requests\_per\_minute:

type: number

error\_rate\_percentage:

type: number

active\_users:

type: integer

Esta especificação de APIs fornece a base completa para desenvolvimento e integração do sistema jurídico com IA, garantindo escalabilidade, confiabilidade e facilidade de uso para todas as funcionalidades principais.