

Documentación

Biblioteca PHP

Biblioteca PHP

Anonymous · 08/10/2026

Tabla de contenidos

Biblioteca PHP	3
<i>Introducción</i>	4
<i>Licencia</i>	5
<i>Instalación</i>	6
<i>Ejemplo</i>	7
<i>DTE disponibles</i>	9
<i>Casos de uso</i>	10
<i>Funcionalidades</i>	11
<i>Contribuir</i>	16
<i>WSDL certificación</i>	19
<i>Legacy OpenSSL</i>	20
<i>PHPDoc</i>	22

Documentación » Biblioteca PHP

Biblioteca PHP

Biblioteca PHP

Anonymous · 08/10/2026

Biblioteca PHP

Última actualización el 08/10/2026

Documentación » Biblioteca PHP » Introducción

Introducción

Núcleo de LibreDTE en PHP

Anonymous · 08/10/2026

Núcleo de LibreDTE en PHP

 CI passing license AGPL-3.0+ downloads 8.64 k downloads 1.11 k this month

Biblioteca escrita en PHP para la integración con el Servicio de Impuestos Internos (SII) asociada a los Documentos Tributarios Electrónicos (DTE). Se construyó originalmente para ser el núcleo de la [Aplicación Web de LibreDTE Edición Comunidad](#), y hoy es también la base del [Dispatcher](#), los [Servicios Web \(API\)](#), la [Imagen Docker \(API\)](#), la [Imagen Docker \(Consola\)](#) y el [Bridge Python](#).

Revisa [Instalación](#) para partir.

Importante

Usar esta biblioteca la embebe dentro de tu propio proceso: tu software se convierte en un trabajo derivado y queda sujeto a los términos de la [licencia AGPL](#). Si buscas integrarte sin ese efecto, revisa los [Servicios Web \(API\)](#), la [Imagen Docker \(API\)](#) o la [Imagen Docker \(Consola\)](#).

Última actualización el 08/10/2026

Licencia

Términos y condiciones de uso

Anonymous · 08/10/2026

Términos y condiciones de uso

Al utilizar este proyecto, total o parcialmente, automáticamente se acepta cumplir con los [términos y condiciones de uso](#) que rigen a LibreDTE. La [Licencia Pública General Affero de GNU \(AGPL\)](#) solo aplica para quienes respeten los términos y condiciones de uso. No existe una licencia comercial de la biblioteca PHP de LibreDTE, por lo cual no es posible usar el proyecto si no aceptas cumplir dichos términos y condiciones.

La versión resumida de los términos y condiciones de uso de LibreDTE que permiten utilizar el proyecto, son los siguientes:

1. Tienes la libertad de: usar, estudiar, distribuir y cambiar LibreDTE.
2. Si utilizas LibreDTE en tu software, el código fuente de dicho software deberá ser liberado de manera pública bajo licencia AGPL.
3. Si haces cambios a LibreDTE deberás liberar de manera pública el código fuente de dichos cambios bajo licencia AGPL.
4. Debes hacer referencia de manera pública en tu software al proyecto y autor original de LibreDTE, tanto si usas LibreDTE sin modificar o realizando cambios al código.

Importante

Es obligación de quienes quieran usar el proyecto leer y aceptar por completo los [términos y condiciones de uso](#).

Última actualización el 08/10/2026

Documentación » Biblioteca PHP » Instalación

Instalación

Instalación mediante composer

Anonymous · 08/10/2026

Instalación

La forma recomendada de instalar la biblioteca PHP de LibreDTE es mediante composer:

```
composer require libredte/libredte-lib-core
```

Requerimientos

- PHP 8.5.
- Extensiones de PHP:
 - curl.
 - json.
 - mbstring.
 - openssl.
 - soap.
 - xdebug (opcional, solo para *testing*).

Última actualización el 08/10/2026

Documentación » Biblioteca PHP » Ejemplo

Ejemplo

Emite tu primer DTE con LibreDTE

Anonymous · 08/10/2026

Ejemplo

El siguiente es un ejemplo básico de cómo generar el XML de un DTE usando la biblioteca PHP de LibreDTE.

```
use Derafu\Certificate\Service\CertificateLoader;
use libredte\lib\Core\Application;

// Obtener la instancia de la aplicación de LibreDTE.
$app = Application::getInstance();

// Preparar datos del DTE.
$data = [ /* aquí los datos del DTE */ ];

// Cargar contenido del archivo CAF.
$caf = $app
    ->getPackageRegistry()
    ->getBillingPackage()
    ->getIdentifierComponent()
    ->getCafLoaderWorker()
    ->load(file_get_contents($cafFile))
    ->getCaf()
;

// Cargar el certificado digital.
$certificateLoader = new CertificateLoader();
$certificate = $certificateLoader->loadFromFile($certificateFile, $certificatePass);

// Crear DTE.
$document = $app
    ->getPackageRegistry()
    ->getBillingPackage()
    ->getDocumentComponent()
    ->bill($data, $caf, $certificate)
    ->getDocument()
;

// Mostrar el XML del DTE generado.
echo $document->saveXml() , "\n";
```

Nota

Revisa los [casos de uso](#) para ejemplos detallados de cómo construir los datos del DTE según diferentes situaciones.

Última actualización el 08/10/2026

Documentación » Biblioteca PHP » DTE disponibles

DTE disponibles

Documentos tributarios disponibles

Anonymous · 08/10/2026

Documentos tributarios electrónicos disponibles

Listado de documentos tributarios electrónicos (DTE) disponibles en LibreDTE.

Código	Documento	Clase / <i>builder</i>	Soporte oficial
33	Factura Afecta	FacturaAfecta	Sí
34	Factura Exenta	FacturaExenta	Sí
39	Boleta Afecta	BoletaAfecta	Sí
41	Boleta Exenta	BoletaExenta	Sí
43	Liquidación de Factura	LiquidacionFactura	No (próximamente)
46	Factura de Compra	FacturaCompra	Sí
52	Guía de Despacho	GuiaDespacho	Sí
56	Nota de Débito	NotaDebito	Sí
61	Nota de Crédito	NotaCredito	Sí
110	Factura de Exportación	FacturaExportacion	Sí
111	Nota de Débito de Exportación	NotaDebitoExportacion	Sí
112	Nota de Crédito de Exportación	NotaCreditoExportacion	Sí

Advertencia

El hecho de que un documento tributario electrónico (DTE) cuente con soporte oficial en LibreDTE significa que existe una implementación que permite generar y validar ese tipo de documento, cumpliendo con los requisitos mínimos para pasar las pruebas de la biblioteca en el entorno de certificación del SII. Sin embargo, esto no implica que estén cubiertos todos los escenarios o casos de uso posibles definidos por la normativa tributaria vigente.

Última actualización el 08/10/2026

Documentación » Biblioteca PHP » Casos de uso

Casos de uso

Ejemplos de DTE en YAML

Anonymous · 08/10/2026

Casos de uso

En el directorio `tests/fixtures/yaml/documentos_ok` se encuentran los casos de uso más comunes que se han visto en 10 años de historia de LibreDTE. Se recomienda mucho que el programador revise dichos ejemplos antes de intentar una integración.

Los casos de uso representan una situación específica que se puede dar al crear un DTE.

El listado de casos de uso por tipo de documento es:

1. [Factura Afecta \(DTE 33\)](#)
2. [Factura Exenta \(DTE 34\)](#)
3. [Boleta Afecta \(DTE 39\)](#)
4. [Boleta Exenta \(DTE 41\)](#)
5. [Liquidación de Factura \(DTE 43\)](#)
6. [Factura de Compra \(DTE 46\)](#)
7. [Guía de Despacho \(DTE 52\)](#)
8. [Nota de Débito \(DTE 56\)](#)
9. [Nota de Crédito \(DTE 61\)](#)
10. [Factura de Exportación \(DTE 110\)](#)
11. [Nota de Débito de Exportación \(DTE 111\)](#)
12. [Nota de Crédito de Exportación \(DTE 112\)](#)

Nota

El caso de uso [Liquidación de Factura \(DTE 43\)](#) no está implementado en la biblioteca PHP de LibreDTE.

Última actualización el 08/10/2026

Funcionalidades

Funcionalidades y características

Anonymous · 08/10/2026

Funcionalidades y características

Estructura general

El proyecto se organiza en **paquetes** (Package). Cada paquete agrupa **componentes** (Component) relacionados con un dominio. Cada componente expone su funcionalidad a través de **workers** (Worker), que son los servicios principales. Los workers pueden tener estrategias intercambiables para distintas variantes de una misma operación.

El núcleo de LibreDTE tiene solo el paquete **Billing**, con los siguientes componentes:

Componente	Descripción
billing.trading_parties	Entidades participantes: emisor, receptor, contribuyente, mandatario.
billing.identifier	Autorización de folios (CAF).
billing.document	Ciclo de vida completo de los DTE: parseo, normalización, construcción, validación, renderizado, envío.
billing.integration	Cliente para los servicios web del SII (API SOAP).
billing.exchange	Intercambio de documentos entre partes vía email o SII.
billing.book	Libros y registros tributarios.
billing.ownership_transfer	Cesión electrónica (factoring/AEC).

Servicios externos usados por LibreDTE Core

XML (derafu/xml)

1. ✓ `XmlDocument`: extiende `DomDocument` con funcionalidades adicionales, incluyendo `query()`.
2. ✓ `XmlEncoder` y `XmlDecoder`: conversión entre arreglo PHP y string XML.
3. ✓ `XmlValidator`: validación de esquema XSD.

Certificados digitales (`derafu/certificate`)

1. ✓ `Certificate`: representa un certificado digital con clave pública, privada y metadatos (RUN, nombre, período de validez).
2. ✓ `CertificateFaker`: genera certificados falsos para pruebas.

Firma electrónica (`derafu/signature`)

1. ✓ `SignatureGenerator`, `SignatureValidator`, `XmlSignatureNode`: firma y validación XML DSIG estándar, pero preparada para lo que el SII requiere.

Funcionalidades de LibreDTE Core

Partes comerciales (`billing.trading_parties`)

1. ✓ `Contribuyente`, `Emisor`, `Receptor`, `Mandatario`, `PersonaNatural`: entidades que representan a los participantes de un DTE.
2. ✓ `ContribuyenteFactory`, `EmisorFactory`, `ReceptorFactory`, `MandatarioFactory`: fábricas para construir cada entidad.
3. ✓ `MandatarioManagerWorker`: gestión del ciclo de vida de mandatarios.
4. ✓ `AutorizacionDte`: representación de la autorización de un emisor para emitir DTE.

Autorización de folios — CAF (`billing.identifier`)

1. ✓ `Caf`: representa un archivo CAF con clave pública, privada, tipo de documento, rango de folios y período de validez.
2. ✓ `CafFaker` y `CafFakerWorker`: generación de CAF falsos para pruebas.
3. ✓ `CafLoaderWorker`: carga de archivos CAF desde disco u otras fuentes.
4. ✓ `CafValidatorWorker`: validación de un CAF (vigencia, rango, firma).
5. ✓ `CafProviderWorker`: proveedor de CAF que combina carga y validación.
6. ✓ `CafBag`: colección de CAF para múltiples tipos de documento.

Documentos tributarios — DTE (`billing.document`)

1. ✓ `TipoDocumento` y `TipoDocumentoFactory`: representa un tipo de DTE con sus atributos (código, nombre, categoría, operación, tag XML, etc.).
2. ✓ Entidades de documento (`FacturaAfecta`, `FacturaExenta`, `FacturaCompra`, `FacturaExportacion`, `BoletaAfecta`, `BoletaExenta`, `NotaCredito`, `NotaDebito`, `NotaCreditoExportacion`, `NotaDebitoExportacion`, `GuiaDespacho`, `LiquidacionFactura`): heredan de `AbstractDocument`.
3. ✓ `SobreEnvio`: representa el XML `EnvioDTE` o `EnvioBOLETA` con métodos para firmar y validar.
4. ✓ `ImpuestoAdicionalRetencion`: soporte para impuestos adicionales y retenciones (excepto combustibles).
5. ✓ `ParserWorker`: parseo de datos de entrada en formato JSON, XML, YAML y formulario HTML.

6. ✓ **NormalizerWorker**: normalización de datos por tipo de documento, con estrategias específicas para cada uno y traits de normalización (detalle, descuentos/recargos, IVA, exportación, transporte, boletas, impuestos adicionales).
7. ✓ **BuilderWorker**: construcción del XML del DTE (timbre, firma, validación de esquema), con estrategia por tipo de documento.
8. ✓ **SanitizerWorker**: sanitización de los datos del DTE por tipo de documento.
9. ✓ **ValidatorWorker**: validación de datos, esquema XSD y firma electrónica del DTE por tipo de documento.
10. ✓ **RendererWorker** con **EstandarRendererStrategy**: generación de PDF en formato estándar (hoja carta).
11. ✓ **LoaderWorker**: carga de un DTE desde XML.
12. ✓ **DispatcherWorker**: despacho de documentos al siguiente paso del proceso.
13. ✓ **BatchProcessorWorker**: procesamiento en lote de múltiples documentos.
14. ✓ **DocumentBagManagerWorker**: gestión del **DocumentBag** (contenedor de datos de un DTE en proceso).

Intercambio de documentos (**billing.exchange**)

1. ✓ **SenderWorker**: envío de documentos con los siguientes handlers y estrategias:
 - ✓ **EmailSenderHandler** + **SmtplibSenderStrategy**: envío por correo electrónico vía SMTP.
 - ✓ **SiiSenderHandler** + **DteSenderStrategy**: envío de DTE al SII.
 - ✓ **SiiSenderHandler** + **BoletaSenderStrategy**: envío de boletas al SII.
2. ✓ **ReceiverWorker**: recepción de documentos con:
 - ✓ **EmailReceiverHandler** + **ImapReceiverStrategy**: recepción de documentos desde correo electrónico vía IMAP.
3. ✓ **DocumentResponseWorker**: generación y validación de XML de respuesta al intercambio de DTE:
 - ✓ **buildEnvioRecibos()**: genera el XML **EnvioRecibos** con recibos de mercaderías (acción ERM). Cada Recibo se firma individualmente con ID **LibreDTE_T{tipo}F{folio}** y el **SetRecibos** se firma con ID **LibreDTE_SetDteRecibidos**.
 - ✓ **buildRespuestaEnvio()**: genera el XML **RespuestaDTE** con **RecepcionEnvio** (acuse del sobre) o **ResultadoDTE** (resultado comercial por DTE). Se firma con ID **LibreDTE_ResultadoEnvio**.
 - ✓ **validateSchema()**: valida el esquema XSD del documento, retorna **XmlDocumentInterface**.
 - ✓ **validateSignature()**: valida todas las firmas electrónicas, retorna **array<SignatureValidationResultInterface>**.

Libros tributarios (**billing.book**)

1. ✓ **TipoLibro**, **TipoOperacion**: enumeraciones de tipos de libro y operación.
2. ✓ **LibroComprasVentas**, **LibroGuias**, **LibroBoletas**, **ResumenVentasDiarias**: entidades que heredan de **AbstractBook**.
3. ✓ **LoaderWorker** con estrategias **ArrayLoaderStrategy** por tipo de libro.

4. ✓ `BuilderWorker` con estrategias `BuilderStrategy` por tipo de libro.
5. ✓ `ValidatorWorker`: validación de esquema XSD (`validateSchema()`) y firma electrónica (`validateSignature()`).
6. □ `RendererWorker`: generación del PDF de los libros.
7. ✓ `BookBag`: contenedor de datos del libro en proceso.

Cesión electrónica / Factoring (`billing.ownership_transfer`)

1. ✓ `AbstractOwnershipTransferDocument`: clase base para las entidades del AEC, con `getXml()`, `getId()`, `getSchema()` y `getSignatureNamespace()` (retorna `null` para usar namespace `xmldsig` estándar).
2. ✓ Entidades `DteCedido`, `Cesion`, `Aec`: heredan de `AbstractOwnershipTransferDocument`. Cada una corresponde a una capa de firma del AEC.
3. ✓ `AecBag`: contenedor de datos que acepta `DocumentInterface` (primera cesión) o `Aec` existente (re-cesión). El `seq` es auto-calculado si se omite.
4. ✓ `AecWorker`: generación y validación del Archivo Electrónico de Cesión (AEC):
 - ✓ `build()`: construye el AEC completo con tres capas de firma. Soporta primera cesión (desde un `DocumentInterface`) y re-cesión (desde un `Aec` existente, añadiendo una nueva `Cesion` al mismo `DTECedido`). El SII permite hasta 40 cesiones sobre el mismo DTE.
 - ✓ `validateSchema()`: valida el esquema XSD del AEC, retorna `XmlDocumentInterface`.
 - ✓ `validateSignature()`: valida todas las firmas del AEC (`DTECedido`, `Cesion(es)` y `AEC`), retorna `array<SignatureValidationResultInterface>`.

Integración con SII (`billing.integration`)

1. ✓ `SiiDteWorker`: cliente para los servicios web del SII con las siguientes operaciones:
 - ✓ `authenticate()`: obtención del token de sesión.
 - ✓ `sendXmlDocument()`: envío de un XML al SII, retorna Track ID.
 - ✓ `checkXmlDocumentSentStatus()`: consulta del estado del envío sin detalles de errores.
 - ✓ `requestXmlDocumentSentStatusByEmail()`: solicita al SII el correo con estado del envío y detalles de errores.
 - ✓ `validateDocument()`: verifica que un DTE es legalmente válido (folio, total, etc.).
 - ✓ `validateDocumentSignature()`: verificación avanzada de la firma del DTE en el SII.
 - ✓ `consumeWebservice()`: consumo genérico de un servicio SOAP del SII.
2. ✓ `SiiRcvWorker`: cliente para el Registro de Compra y Venta (RCV) del SII con las siguientes operaciones:
 - ✓ `submitDocumentAcceptance()`: ingresa aceptación o reclamo de un DTE (acciones: ERM, ACD, RCD, RFP, RFT).
 - ✓ `listDocumentEvents()`: lista el historial de eventos de un DTE.
 - ✓ `checkDocumentAssignability()`: consulta si un DTE puede ser cedido (factoring/AEC).
 - ✓ `getDocumentSiiReceptionDate()`: consulta la fecha en que el SII recibió un DTE.
3. ✓ `SiiRtcWorker`: cliente para el Registro de Transferencias de Créditos (RTC) del SII:
 - ✓ `sendAec()`: envía un AEC al RTC. Recibe `SiiRequestInterface`, `XmlDocumentInterface` (el AEC), RUT del cedente y email de contacto. Retorna Track ID. Usa endpoint

`cgi_rtc/RTC/RTCAnotEnvio.cgi` (distinto al de DTE).

Última actualización el 08/10/2026

Documentación » Biblioteca PHP » Contribuir

Contribuir

Cómo contribuir a LibreDTE

Anonymous · 08/10/2026

Contribuir a LibreDTE



Si deseas contribuir a LibreDTE, te lo agradezco. Después de más de 10 años aportando al software libre en Chile, a veces se hace difícil tener todo el tiempo que quisiera para realizar mejoras a LibreDTE. Por lo que cualquier corrección de error, mejora o nueva funcionalidad que tengas en mente me comprometo a revisarla.

[Esteban Delaf](#), Autor de LibreDTE.

Trabaja con una copia del repositorio

Si deseas contribuir con el proyecto, especialmente resolviendo alguna de las [incidencias abiertas](#), debes:

1. Hacer una copia mediante una bifurcación del [repositorio en GitHub](#) y luego clonar dicho repositorio.

```
git clone https://github.com/TU_USUARIO/libredte-lib-core
cd libredte-lib-core
```

2. Crear una rama para los cambios.

```
git checkout -b nombre-branch-que-blabla
```

3. Programar tu código siguiendo los lineamientos descritos más abajo.

```
declare(strict_types=1);

// Seguir el estándar PSR12 o, idealmente, PER Coding Style 3.0.
```

```
// Seguir los principios KISS, SRP, DRY, YAGNI y TDD.  
// Ejecutar las validaciones y pruebas.
```

4. Guardar los cambios en tu repositorio local.

```
git commit -m 'Se agrega...'
```

5. Subir los cambios a tu repositorio en Github.

```
git push origin nombre-branch-que-blabla
```

6. Crear una [solicitud en el repositorio de LibreDTE](#) para unir tu rama con la oficial de LibreDTE.

Lineamientos de código

El código debe seguir los siguientes lineamientos:

1. Respetar al menos el estándar [PSR12](#). Idealmente seguir [PER Coding Style 3.0](#).
2. Seguir los patrones y principios:
 - **KISS**: Mantén el código simple y directo. Evita complejidades innecesarias.
 - **SRP**: Cada clase y método debe tener una única responsabilidad.
 - **DRY**: Evita la duplicación de código. Si encuentras el mismo código en varios lugares, refactorízalo para que se utilice de manera centralizada.
 - **YAGNI**: No implementes funcionalidades que no sean necesarias en el presente. El código debe ser lo más minimalista posible, construyendo solo lo que realmente se necesita.
 - **TDD**: Escribe las pruebas antes del código. Esto ayuda a garantizar que el código haga lo que se supone que debe hacer.
3. Tener el 100% de cobertura en las pruebas.
4. Escribir la documentación correspondiente en el **código** utilizando PHPDoc, y que será validada por *phpstan* con al menos el nivel 5.

Validación del código

Una vez tengas tu código debes ejecutar los siguientes comandos para validarlo:

```
# Revisión del estilo de código según estándar definido para la biblioteca.  
php-cs-fixer fix -v --dry-run --diff --config=php-cs-fixer.php .
```

```
# Buscar errores potenciales en tu código sin ejecutar tests.  
phpstan analyse --configuration=phpstan.neon --memory-limit=1G  
  
# Pruebas unitarias, funcionales y de integración de toda la biblioteca.  
XDEBUG_MODE=coverage vendor/bin/phpunit --configuration=phpunit.xml
```

Importante

Si tu código no pasa las validaciones anteriores no será aceptado para ser unido al repositorio oficial.

Organización de pruebas

Las pruebas se organizan en 3 tipos:

1. **Unitarias:** deben probar solo una clase. Estas pruebas no deben realizar conexiones al SII.
2. **Funcionales:** prueban una funcionalidad con múltiples clases. Estas pruebas no deben realizar conexiones al SII.
3. **Integración:** prueban una funcionalidad que requiere realizar conexión al SII.

Última actualización el 08/10/2026

Documentación » Biblioteca PHP » WSDL certificación

WSDL certificación

Nota sobre los WSDL de certificación

Anonymous · 08/10/2026

Archivos WSDL del ambiente de certificación

Algunos WSDL del ambiente de certificación no funcionan tal cual los provee SII. Lo anterior ya que apuntan a un servidor llamado `nogal.sii.cl` el cual no es accesible desde Internet. Quizás es un servidor local del SII para desarrollo.

Por lo que se han modificado los WSDL para el ambiente de certificación del SII y usar el servidor `maullin.sii.cl` que es el correcto.

Estos WSDL se usarán automáticamente al solicitar el WSDL del ambiente de certificación a través de LibreDTE. Los WSDL para el ambiente de producción son directamente los proporcionados por SII.

Referencias

- <http://stackoverflow.com/a/28464354/3333009>

Última actualización el 08/10/2026

Documentación » Biblioteca PHP » Legacy OpenSSL

Legacy OpenSSL

Nota sobre el uso de OpenSSL 3

Anonymous · 08/10/2026

Legacy OpenSSL

A partir de OpenSSL 3, ciertos algoritmos que anteriormente eran estándar han sido considerados inseguros o “no compatibles”, entre ellos la encriptación de certificados PKCS#12 con 40BitRC2 que es utilizada por los certificados digitales que se utilizan con el SII.

Para que se puedan utilizar los certificados digitales, se debe editar la configuración de OpenSSL en `/etc/ssl/openssl.cnf` para activar los proveedores *legacy*.

```
[default_conf]
openssl_conf = openssl_init

[openssl_init]
providers = provider_sect

[provider_sect]
default = default_sect
legacy = legacy_sect

[default_sect]
activate = 1

[legacy_sect]
activate = 1
```

Revisar compatibilidad con OpenSSL 3

Ejecutar:

```
openssl pkcs12 -in certificadoDigital.pfx -info -nokeys -nocerts
```

Nota: la extensión puede ser `.pfx` o `.p12`.

Si en la respuesta está presente `PKCS7 Encrypted data: pbeWithSHA1And40BitRC2-CBC` indica que se está utilizando 40BitRC2 en el certificado digital para encriptarlo, lo cual es obsoleto e inseguro.

Referencias

- https://wiki.openssl.org/index.php/OpenSSL_3.0#Providers
- <https://stackoverflow.com/a/74416513>
- <https://stackoverflow.com/a/72600724>

Última actualización el 08/10/2026

Documentación » Biblioteca PHP » PHPDoc

PHPDoc

Documentación del código fuente

Anonymous · 08/10/2026

PHPDoc

Última actualización el 08/10/2026