LEDES Software API v1.0 - 2/2/2020

Introduction

The LEDES Software API Subcommittee was established to develop and publish a new software API that will enable direct application-to-application exchange of legal invoices and other legal information. Currently, law firm time and billing applications generate LEDES files which requires manual processing by a billing clerk. This processing generally involves a manual login and uploading of LEDES files to corporate legal department e-billing systems to facilitate the exchange of invoices, accruals and other legal data as well as logging in the corporate legal department e-billing systems to find out the status of a submitted invoice.

The goal of this project is to define an industry standard LEDES Software API specification that when adopted and implemented by software vendors will allow the law firm time and billing applications to communicate directly to the corporate legal department matter management and e-billing applications. This in turn will allow the billing clerks at law firms to simply work within their time and billing applications to view the status of an invoice and to submit invoices, accruals and other legal data– all from one application, requiring no manual processing.

NOTE: Throughout this document, the law firm or other entity who is sending the invoice will be referred to as the "vendor" and the vendor's time & billing system will be referred to as the "sending system". The corporation the law firm or other entity is sending the invoices to will be referred to as the "client" and the client's e-billing system will be receiving the invoice will be referred to as the "receiving system".

Members

Co-Chairs

Name	**Email**
Nicholas Puschak	nick.puschak@mitratech.com
Dan Bodnar	daniel.bodnar@thomsonreuters.com

Involved Members

Name	Email	Company
Jessica Stahler	jstahler@kingspry.com	King Spry Herman Freund & Faul, LLC
Gwen Hess	ghess@grsm.com	Gordon Rees Scully Mansukhani
Sajeel Malani	sajeel.malani@ucop.edu	University of California
Greg Nilsen	greg.nilsen@thomsonreuters.com	Thomson Reuters
Marie Burgess	marie.burgess@aderant.com	Aderant
Kathryn Snyder	KSnyder@zelle.com	Zelle, LLP
Tony Fadulu	tony.fadulu@wolterskluwer.com	Wolters Kluwer
Idris Khajanchi	idris.khajanchi@brooklyn-solutions.com	Brooklyn Solutions
Walter Anderson	walter.anderson@citi.com	Citigroup
Heather M. Magers Keefe	hmagers@foleymansfield.com	Foley Mansfield
David Vasquez	david.vasquez@mitratech.com	Mitratech
Gabriela Isturiz	gabriela@bellefield.com	Bellefield, LLC
Sham Birbahadur	sham.birbahadur@aderant.com	Aderant
Jane Bennitt	jbennitt@globallegalebilling.com	President, LEDES Oversight Committee
Kim Felice	kim.felice@mitratech.com	Mitratech
Jim Hannigan	jhannigan@fenwick.com	Fenwick & West, LLP
K Wang	kwang@anaqua.com	Anaqua
Sherry Askin	sherryaskin@omnisw.com	Omni Software Systems
Joanne Irwin	joanne.irwin@lexisnexis.com	Lexis Nexis
David Nelson	david.nelson@pathfindereconsulting.com	Pathfinder eConsulting / Slaughter & May

Peter Watt	peter.watt@hsf.com	Herbert Smith Freehills
Nadia Strobbia	nadia.strobbia@thomsonreuters.com	Thomson Reuters

Charter

OVERVIEW: The LEDES Software API Subcommittee was established to develop and publish a new software API that will enable direct application-to-application exchange of legal invoices and other legal information. Currently, law firm time and billing applications generate LEDES files which requires manual processing by a billing clerk. This processing generally involves a manual login and uploading of LEDES files to corporate legal department e-billing systems to facilitate the exchange of invoices, accruals and other legal data.
The goal of this project is to define an industry standard LEDES Software API specification that will allow the law firm time and billing applications to communicate directly to the corporate legal department matter management and e-billing applications. This in turn will allow the billing clerks at law firms to simply work within their time and billing applications to view the status of an invoice and to submit invoices, accruals and other legal data– all from one application, requiring no manual processing.
SUBCOMMITTEE MEMBERSHIP: Co-Chairs Nicholas Puschak and Daniel Bodnar have been identified to lead the subcommittee and will work closely with Jane Bennitt from the Board. The subcommittee will interview the LOC e-billing vendor members for input. The subcommittee will propose a new LEDES Software API and post this for public comment.
PURPOSE: The purpose of the sub-committee is to create a new LEDES Software API for the direct Software application-to-application exchange of legal invoices and other legal information.
HIGH LEVEL STAGES:
Stage 1: A subcommittee will be formed and will be comprised of LEDES general membership interested in the project.
Stage 2: The leadership of the subcommittee will schedule technical meetings with the LOC e-billing vendor members to define the scope of this project and to draft a proposed LEDES Software API.
Stage 3: The leadership of the subcommittee will schedule meetings with the general subcommittee to review the draft LEDES Software API and start accepting comments and recommendations from the general membership and the legal community at large.
Stage 4: Incorporation/review of comments and recommendations received regarding the LEDES Software API, finalization of LEDES Software API and submission for ratification by the LEDES Board of Directors.
TIMELINE: Submit proposed LEDES Software API for review/comment by general membership by the summer of 2018. It is the committee’s desire to have the LEDES Software API v1 ratified by the end of 2018.

Committee Co- Chairs:
Nicholas Puschak, Mitratech, nick.puschak@mitratech.com (610-729-1118)
Daniel Bodnar, Thomson Reuters, daniel.bodnar@tr.com, (425-247-7551)

Existing Invoicing APIs

It was decided to review other invoice API in the general industry and in the legal e-billing industry. We looked at the following APIs.

PayPal - https://developer.paypal.com/docs/api/invoicing/
Envoice -
https://documenter.getpostman.com/view/3559821/envoice-api/RVfqnE3a#d0de6d76-fb5f-be11-7b41-3a09544b0544

Requirements

The following are a list of requirements and recommendations for the LEDES API Standard. By using this API, the vendor agrees to implement the following requirements and recommendations in their application.

Web Service

The solution should be a web service. The two leading candidates are REST and SOAP web service technologies. REST (REpresentational State Transfer) is the dominant web service technology being used in the industry today.

Web Service: REST

Security

Security is paramount in the solution since legal invoices can contain highly private information. For example, the invoice line item descriptions and matter names can contain highly sensitive information. A comprehensive security structure is highly recommended, however, the LEDES API standard requires common security measures. Transmission encryption and session authentication are required, with recommended levels. Content encryption is also highly recommended to address additional vulnerabilities in the management of sensitive data. These security recommendations in general will need to be re-evaluated on a regular basis.

Transmission Encryption Security

All data should be encrypted in transit including authentication credentials (passwords), API keys, or JSON Web Tokens. This is needed to allow users to authenticate with the service and protects the integrity of the transmitted data.
This first level of security can be achieved with HTTPS (TLS) and the use of mutually authenticated client-side certificates to provide additional protection for the LEDES API standard. If implemented, client certificates will need to be managed between the law firm (sending system) and e-billing vendor (receiving system) by having certificates built from special 'roots'.

Minimum Requirement: HTTPS (TLS 1.1+)
Recommendation: Client Certificates
Implementation: Vendor application (all vendors)
User: no action required

Session Authentication Security

Once transmission is secure, interaction with the user must be addressed. This can be addressed by employing a secure authentication and session management mechanism.   User authorization for resources and data must be orchestrated through multiple vulnerability points, such as access permissions, session duration, and global logout.

The widely adopted protocols for authentication and session management today are OAuth 2.0 and OpenID Connect. OAuth 2.0 introduces an authorization layer and separates the web client from the sensitive system resources. The user is issued unique credentials, which protects the credentials used to manage these resources. OpenID Connect works with OAuth 2.0 to add an identity layer on top of the authentication layer, so more sophisticated permissions can be orchestrated in the application.

Requirements and recommendations are listed as “or equivalent” in recognition of multiple Identity Providers a vendor may subscribe to in order to offer appropriate authentication security. Popular Identity Providers include Amazon, Google, and Microsoft, however no provider is endorsed or recommended here.

Minimum Requirement: OAuth 2.0 (or equivalent)

Recommendation: Open ID (or equivalent)

Implementation: Vendor applications and system (or service) integration

User: one-time authorization set up

Content Encryption Security

All invoice and related data should be encrypted while in use and at rest. This is the final measure recommended to provide adequate security through the full life cycle of the data and application.

There are multiple methods to provide content encryption and it is highly recommended that any new implementation decisions are part of a comprehensive analysis of current application and system integration vulnerabilities between vendors and with their end users. The current specification for encryption established by the U.S. National Institutes of Standards and Technology (NIST) and adopted globally is Advanced Encryption Standard (AES). This is a symmetric key approach, with accepted key lengths of 128, 192, and 256 bits. Recent successful attacks on 128 bit keys suggest it is appropriate to consider the use of longer keys.

The API allows for the content to be optionally encrypted before it is transmitted, however it is not required. It is highly recommended that eBilling vendor support content encryption, however not all time and billing systems may support it. Below are the requirements for when the content is encrypted.

Minimum Requirement: AES-256 (or equivalent) if content encryption is used.

Implementation: Vendor applications and system integration

User: one-time authorization set up

Send an Invoice

The solution should have the ability to send a LEDES invoice with one or more attachments. Initially all existing LEDES invoice file formats should be supported.

NOTE: LEDES has not established a JSON Invoice standard so it was decided to only support sending the existing LEDES invoice files as a file within this API call. It is anticipated that a future requirement for this API may include development of a LEDES JSON invoice structure with a corresponding Send Invoice JSON call.

Send an Accrual Invoice

The solution should have the ability to send an accrual invoice with one or more attachments.

NOTE: Since the LEDES organization has not developed a specific format for sending accrual information, the requirement will be to support the use of the existing LEDES invoice file formats and simply designate the invoice as an accrual invoice. It is anticipated that a future requirement for this API may include development of a LEDES JSON accrual structure with a corresponding Send Accrual JSON call.

Send a Shadow Invoice

The solution should have the ability to send a shadow invoice with one or more attachments. A shadow invoice is a normal LEDES invoice file but the invoice should not be paid because of a flat fee or other alternative fee arrangement.

Resubmit an Invoice

The solution should have the ability to resubmit an invoice that was previously completely rejected. The resubmitted invoice, like the original should allow for one or more attachments. When an invoice is resubmitted the original invoice must be identified and a comment should be sent to explain why the invoice is being resubmitted. It is recommended that the invoice number of the resubmitted invoice be similar to the original invoice, but the invoice number must still be unique. The receiving application can put a limit on how many times a sending system can resubmit an invoice.

Appeal an Invoice

The solution should have the ability to appeal an invoice that was previously submitted and partially paid. The appeal invoice, like the original should allow for one or more attachments. Only the line items that were rejected and weren’t paid should be submitted in the appeal invoice. For line items that were adjusted and partially paid the appeal invoice should only include the additional amount being appealed. It is recommended that the invoice number of the appeal invoice be similar to the original invoice, but the invoice number must still be unique. When appealing an invoice, the original invoice must be identified and a comment should be sent to explain why the line items of the original invoice are being appealed. The receiving application can put a limit on how many times a sending system can appeal an invoice.

Replace an Invoice

The solution should have the ability to replace an invoice that was previously submitted. A replacement invoice is meant to allow a sending system to replace an invoice if the original invoice was submitted in error, typically immediately after submitting the bad invoice. The receiving application could require that the original invoice be deleted prior to being able to submit a replacement invoice. The original invoice must be identified and a comment should be sent to explain why the invoice is being replaced. The receiving application should immediately void and reject the original invoice and any attachments. The receiving application can refuse the replacement invoice if it is unable to replace the original invoice, for example, if the invoice has already been approved or paid. If the receiving application can completely void the original invoice than the replacement invoice can use the same invoice number, however if the receiving application can not completely void the original invoice then it is recommended that the invoice number of the replacement invoice be similar to the original invoice, but the invoice number must still be unique.

Acknowledgement Information

When an invoice is successfully submitted, the sending application should receive back an invoice ID and the date and time the invoice was received. The Invoice ID should be the receiving system's identifier for the invoice. For an unsuccessful invoice submission the sending application should receive back a set of errors and the date and time of the submission.

Delete an Invoice

The solution should have the ability to tell the receiving application to delete or reject an invoice that was previously submitted. The receiving application can reject this request, for example, if the invoice has already been approved or paid.

Get the Status of an Invoice

The solution should have the ability to get the status of a single invoice sent by a sending system. Status information should include:

- The current state of the invoice and the date and time of that state.
- A list of errors within the Invoice. Each error should include the error code, description and the location within the invoice of the error. The new initiative to establish an industry standard set of LEDES Invoice error codes should be used.
- A list of any adjustments applied to the invoice. Each adjustment should include the adjust type, date and time of the adjustment, the amount and currency of the adjustment, the reason for the adjustment, and the line items being adjusted.
- Payment information including the payment type, date and time of the payment, payment amount and currency, payment reference number, payee, and the account to which the payment was made.

Get the Status of a Set of Invoices

The solution should have the ability to get the status of all invoices which have been sent by a sending system or all invoices whose status has changed since the last request. The status information for each invoice should be the same as above.

LEDES API Endpoints

The following is a list of endpoints for the LEDES API v1.

NOTE: The Google JSON Style Guide is used for conventions.
https://google.github.io/styleguide/jsoncstyleguide.xml
This builds upon the JSON specification found at JSON.org.

This set of calls are required by the API.

API Call Name	API Call Description
Send Invoice LEDES File POST `/v1/invoices/ledesfile`	Send an invoice using one of the existing LEDES file formats (ie. LEDES XML) as a file. This is the main method to send an existing LEDES invoice file. The LEDES file can optionally be encrypted. Returns an *invoiceID, receivedDateTime,* and *errors.* The *invoiceID* that is used for subsequent API calls to send an attachment for an invoice or to get the status of an invoice.
Send Invoice Attachment POST `/v1/invoices/{invoiceID}/attachment`	Send an attachment for a previously submitted Invoice by *invoiceID*. Multiple calls can be made for the same *invoiceID* to send multiple attachments for an Invoice. The attachment file can be optionally encrypted. Returns an *attachmentID*, *receivedDateTime,* and *errors*.
Get Invoice Status GET `/v1/invoices/{invoiceID}`	Get the status of an invoice. Requires passing a valid *invoiceID*. The response is an InvoiceStatus object, which is a fairly complex object supporting a variety of information about the invoice.

This API calls is optional for advanced functionality.

Get Invoice Status Changes GET `/v1/invoices/statusChanges?invoiceStatusMarker={invoiceStatusMarker}`	Get the status of a set of invoices. If no *invoiceStatusMarker* is passed in, then this call returns the status of all submitted invoices plus a *invoiceStatusMarker* for subsequent calls. When passing in a *invoiceStatusMarker* this call will return the status of all invoices that have status updates since that marker was issued. The calling system must save this marker for subsequent calls. This call is optional and is not a requirement for the API.

Possible future API calls.

Send Invoice LEDES JSON POST `/v1/invoices/ledesjson`	Send an invoice using a future LEDES JSON format. Returns an invoiceID.

Delete Invoice DELETE `/v1/invoices/{invoiceID}`	Delete the previously submitted invoice. This call can only be performed on an invoice prior to getting to the Approved or Rejected status.

Send Invoice LEDES file

POST `/v1/invoices/ledesfile`

Send an invoice using one of the existing LEDES file formats (ie. LEDES XML) as a file. This is the main method to send an existing LEDES invoice file. The LEDES file can be optionally encrypted. NOTE: It is already required that all data be encrypted in transit, so this would be additional encryption of the LEDES file, which would then require a mechanism for the receiving system to be able to unencrypt the LEDES file. If encrypted, the encryption method is part of the system setup. Returns an *invoiceID*, *receivedDateTime,* and *errors* (see response below). The *invoiceID* is used for subsequent API calls to send an attachment for the invoice or to get the status of the invoice. The vendor for the invoice should be identified by the authentication and checked against the vendor identified within the LEDES files.

NOTE: The modern E-Invoice APIs support sending invoices and invoice data directly within a REST call using a JSON invoice structure. LEDES has not established a JSON Invoice standard so it was decided to only support sending the existing LEDES invoice files as a file within this call. It is anticipated that a future requirement for this API may include development of a LEDES JSON invoice structure with a corresponding Send Invoice JSON call.

NOTE: This API assumes each LEDES file will only contain a single Invoice.

Request Body

ledesFormat enum required

The LEDES invoice file format. This is repeated in the first line of the invoiceFile.

Possible values: `LEDES98B, LEDES98BI, LEDES98BIV2, LEDES2000, LEDESXML20, LEDESXML21`

---

encrypted enum required

Whether the invoiceFile is encrypted or not.

Possible values: `Y, N`

---

ledesFilename string required

The name of the LEDES file. This does not need to be unique or be the invoice number, which is a common filename for LEDES file. It can simply be the filename of the LEDES file on the sending system side for reference purposes. This should be the filename without mime type (ie. INV-2019-123). This should not include any network or folder path information. Receiving systems can set a limit for the length of filenames with the appropriate error message described below.

---

fileMIMEType string required

The MIME type of the file.

Example value: text/plain

NOTE: Here is a fairly good list of mime types:
https://www.iana.org/assignments/media-types/media-types.xhtml
https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types/Complete_list_of_MIME_types

---

invoiceType enum required

The type of Invoice being submitted.
Possible values: invoice, accrual, shadow, resubmit, appeal, replacement.

invoiceType	**Description**
invoice	The submission of a standard invoice to be paid.
accrual	The submission is a pre-bill or accrual invoice and should not be paid. NOTE: Since the LEDES organization has not developed a specific format for sending accrual information or an accrual invoice the existing LEDES invoice file formats will be used and simply designate the invoice as an accrual invoice.

shadow	The submission is a shadow invoice that should not be paid. Shadow invoices are usually submitted for flat fee or other alternative fee arrangements.
resubmit	The submission is an invoice that was previously rejected. The original invoice must be identified with the relatedInvoiceID element. The comment element in this call should be used to explain why the invoice is being resubmitted. It is recommended that the invoice number of the resubmitted invoice be similar to the original invoice, but the invoice number must still be unique. The receiving application can put a limit on how many times a sending system can resubmit an invoice.
appeal	The submission is an invoice that was previously submitted and partially paid. Only the line items that were rejected and weren't paid should be submitted in the appeal invoice. For line items that were adjusted and partially paid the appeal invoice should only include the additional amount being appealed. The original invoice must be identified with the relatedInvoiceID element. The comment element should be used to explain why the line items of the original invoice are being appealed. It is recommended that the invoice number of the appeal invoice be similar to the original invoice, but the invoice number must still be unique.
replacement	The submission is an invoice that was previously submitted and you would like this new invoice to replace the previous one. The original invoice must be identified with the relatedInvoiceID element. The comment element should be used to explain why the invoice is being replaced. The application can refuse the replacement invoice if it is unable to completely replace the original invoice, for example, if its has already be approved or paid. This invoice type is meant to allow the sending system to submit an invoice if the original one was in error, typically immediately after submitting the bad invoice. The receiving application could require that the Delete Invoice **`(/v1/invoices/{`*`invoiceID`*`})`** method be called prior to being able to submit a replacement invoice. If the receiving application can completely void the original invoice, then the replacement invoice can use the same invoice number. However, if the receiving application can not completely void the original invoice, then it is recommended that the invoice number of the replacement invoice be similar to the original invoice, but the invoice number must still be unique.

---

relatedInvoiceID string optional

The Invoice ID of a related invoice that the submitted invoice is associated. For example, if an invoice is rejected and then a new invoice is submitted with adjustments or as an appeal, this would be the invoice ID of the originally submitted invoice. This is required for invoiceTypes of resubmit, appeal and replacement.

---

comment string optional

This is optional, but it’s recommended to be used for resubmit, appeal, and replacement submissions to explain why they are being submitted.

---

ledesFile string required

This is the actual LEDES file. The actual LEDES file is not passed in the API call, but instead the relative path to the file is passed as a REST Multipart call. See the CURL sample request below for how this parameter is passed. The file should be in the format according to the LEDESFormat specified above. The LEDES file may be encrypted according to the encrypted parameter. If encrypted, the encryption method is part of the system setup. The receiving system may set a limit to the size of the file.

Response Body

A successful request returns the HTTP 201 Created status code and a JSON response body that returns the invoice ID and date/time it was officially received. The errors property will be NULL for successful responses and will contain an array of error messages when a HTTP 400 Bad Request is returned.

---

invoiceID string required (if successful)

The receiving system’s ID of the invoice. This is used for subsequent API calls to send an attachment for an invoice or to get the status of an invoice. It must be a unique reference to the invoice within the receiving system. Each receiving system can decide what type of ID to return for an invoice. It may be recommended to use a GUID (globally unique identifier) for the invoiceID. It is **not** recommended to use the vendor’s Invoice Number for this ID for security reasons since this may not be encrypted. invoiceID is Read only.

---

receivedDateTime datetime (ISO8601) required

The audit date time for when the receiving system officially received the invoice.

errors Array [string] optional

If there are any errors, this element should be used to describe the errors.

Unsuccessful Responses

An unsuccessful request returns the HTTP 400 Error status code. Also see the section on standard REST Error codes at the end of this document.

NOTE: Common e-billing invoice errors detected within the LEDES file data itself are not included in the unsuccessful responses listed below for the Send Invoice LEDES File call. This is because the receiving system will need to first receive the LEDES file, save it, parse it and process it with various LEDES file checks and audit rules. These common e-billing errors are made available via the Invoice Status API calls.

These errors should be returned if any of the required request body fields are missing.

- `"error": "ledesFormat required field missing."`
- `"error": "encrypted required field missing."`
- `"error": "ledesFilename required field missing."`
- `"error": "fileMIMEType required field missing."`
- `"error": "invoiceType required field missing."`
- `"error": "relatedInvoiceID required for invoiceTypes of resubmit, appeal and replacement."`
- `"error": "ledesFile required field missing."`

These errors should be returned if any of the enum fields have an invalid format. Each implementation can decide which items to support and should list the supported values in the error message.

- `"error": "Invalid ledesFormat value. Supported formats include ..."`

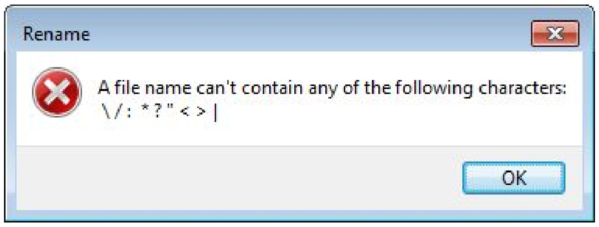
  `WHERE … would be a list of LEDES formats that the receiving system supports. For example: "error": "invalid ledesFormat value. Supported formats include LEDES98B, LEDES98BI, LEDES98BIV2, LEDES2000, LEDESXML20, LEDESXML21"`
- `"error": "Invalid encrypted value. Supported values are Y and N."`
- `"error": "ledesFilename length too long. Filename is limited to X characters."`

  `WHERE X would be the number of characters allowed in the filename that the receiving system supports.`

- "error": "ledesFilename is an invalid file name. A file name can’t contain any of the following characters: ..."

  WHERE ... would be the set of characters not allowed in the file names that the receiving system does not support.

For example, here is the error message from Windows.


Rename

A file name can't contain any of the following characters:
\ / : * ? " < > |

- "error": "Invalid fileMIMEType value. Supported formats include ..."

  WHERE … would be a list of fileMIMEType’s that the receiving system supports.
- "error": "Invalid invoiceType value. Supported formats include ..."

  WHERE … would be a list of invoice types the receiving system supports. For example: "error": "invalid invoiceType value. Supported formats include invoice, accrual, shadow, resubmit, appeal, replacement."
- "error": "Replacement invoice is not allowed ..."

  WHERE … would explain why the replacement invoice is not allowed.

CURL Sample Request - Successful Invoice Submission

NOTE: The actual LEDES file is sent separately in a multi-part post.

```
curl -v -X POST https://ServerURL/v1/invoices/ledesfile \
-H "Authorization: XYZABC123" \
-F 'details={
   "ledesFormat": "LEDES98B",
   "encrypted": "Y",
   "ledesFilename": "INV2018-3342",
   "fileMIMEType": "text/plain",
   "invoiceType": "invoice"
};type=application/json’ \
-F 'ledesFile=@/path/to/file/INV2018-3342.txt'
```

WHERE

- '/path/to/file/INV2018-3342.txt' is the path on the calling system's server for the LEDES file. Also notice that spaces must be prefaced with a backslash (\).
- `XYZABC123` is the security access token
- `ServerURL` is the receiving servers URL

CURL Sample Response - Successful Invoice Submission

Note that the errors property was not returned since there were no errors.

```
{
  "invoiceID": "INV2-RUVR-ADWQ-H89Y-ABCD",
  "receivedDateTime": "2018-07-24 12:11:52 PDT"
}
```

CURL Sample Request - Unsuccessful Invoice Submission

NOTE: The ledesFormat and encrypted values are invalid.

```
curl -v -X POST https://ServerURL/v1/invoices/ledesfile \
-H "Authorization: XYZABC123" \
-F 'details={
   "ledesFormat": "LEDES1998B",
   "encrypted": "AES",
   "ledesFilename": "INV2018-3352",
   "fileMIMEType": "text/plain",
   "invoiceType": "invoice"
};type=application/json' \
-F 'ledesFile=@/path/to/file/INV2018-3342.txt'
```

CURL Sample Response - Unsuccessful Invoice Submission

Note that the invoiceID was not returned, since there were errors in the call and no invoice was received. The receivedDateTime was returned to acknowledge when the call was responded to. The errors array property was returned with two errors.

```
{
  "receivedDateTime": "2018-07-24 12:11:52 PDT"
  "errors": [ {
      "error": "Invalid ledesFormat value - LEDES1998B. Supported
formats include LEDES98B, LEDES98BI, LEDES98BIV2, LEDES2000,
LEDESXML20, LEDESXML21"
    },{
      "error": "Invalid encrypted value. Supported values are Y and
N."
```

```
    }
  ]
}
```

CURL Sample Request - Accrual Invoice Submission

Notice the invoiceType..

```
curl -v -X POST https://ServerURL/v1/invoices/ledesfile \
-H "Authorization: XYZABC123" \
-F 'details={
   "LEDESFormat": "LEDES98B",
   "encrypted": "Y",
   "ledesFilename": "ACC 2018-3312",
   "fileMIMEType": "text/plain",
   "invoiceType": "accrual"
};type=application/json’ \
-F 'ledesFile=@/path/to/file/INV2018-3342.txt'
```

CURL Sample Request - Shadow Invoice Submission

Notice the invoiceType.

```
curl -v -X POST https://ServerURL/v1/invoices/ledesfile \
-H "Authorization: XYZABC123" \
-F 'details={
   "LEDESFormat": "LEDES98B",
   "encrypted": "Y",
   "ledesFilename": "SHA2018-3322",
   "fileMIMEType": "text/plain",
   "invoiceType": "shadow"
};type=application/json’ \
-F 'ledesFile=@/path/to/file/INV2018-3342.txt'
```

CURL Sample Request - Resubmit Invoice Submission

Notice the invoiceType, relatedInvoiceID, and comment.

```
curl -v -X POST https://ServerURL/v1/invoices/ledesfile \
-H "Authorization: XYZABC123" \
-F 'details={
   "LEDESFormat": "LEDES98B",
   "encrypted": "Y",
   "ledesFilename": "INV2018-3342R1",
   "fileMIMEType": "text/plain",
   "invoiceType": "resubmit",
   "relatedInvoiceID": "INV2-RUVR-ADWQ-H89Y-ABCD",
   "comment": "This invoice is being resubmitted with changes to
Jack Jones hourly rate as we discussed."
};type=application/json' \
-F 'ledesFile=@/path/to/file/INV2018-3342.txt'
```

CURL Sample Request - Appeal Invoice Submission

Notice the invoiceType, relatedInvoiceID, and comment.

```
curl -v -X POST https://ServerURL/v1/invoices/ledesfile \
-H "Authorization: XYZABC123" \
-F 'details={
   "LEDESFormat": "LEDES98B",
   "encrypted": "Y",
   "ledesFilename": "INV2018-3342A1",
   "fileMIMEType": "text/plain",
   "invoiceType": "appeal",
   "relatedInvoiceID": "INV2-RUVR-ADWQ-H89Y-ABCD",
   "comment": "Jane Doe's lines items that were previously rejected
are being appealed as we discussed."
};type=application/json' \
-F 'ledesFile=@/path/to/file/INV2018-3342.txt'
```

CURL Sample Request - Replacement Invoice Submission

Notice the invoiceType, relatedInvoiceID, and comment.

```
curl -v -X POST https://ServerURL/v1/invoices/ledesfile \
-H "Authorization: XYZABC123" \
-F 'details={
   "LEDESFormat": "LEDES98B",
   "encrypted": "Y",
   "ledesFilename": "INV2018-3342",
   "fileMIMEType": "text/plain",
   "invoiceType": "replacement",
   "relatedInvoiceID": "INV2-RUVR-ADWQ-H89Y-ABCD",
   "comment": "Invoice INV2018-3342 is being completely replaced
with this new version due to errors on our side. Please void and
disregard previously submitted invoice."
};type=application/json’ \
-F 'ledesFile=@/path/to/file/INV2018-3342.txt'
```

HTTP Sample Request

```
POST https://ServerURL/v1/invoices/ledesfile
Host: somehost.com
Authorization: <Access-Token>
Accept: application/json
Content-Type: multipart/form-data;
boundary="3320-jfd90-3j2f90j233-f23"
Content-Length: 834

--3320-jfd90-3j2f90j233-f23
Content-Disposition: form-data; name=details
Content-Type: application/json; charset=utf-8

{“LEDESFormat”: “LEDES98B”,
 “encrypted”: “Y”,
 “ledesFilename”: “INV2018-3342”,
 “fileMIMEType”: “text/plain”,
 “invoiceType”: “invoice”}

—3320-jfd90-3j2f90j233-f23
Content-Disposition: form-data; name=ledesFile
Content-Type: text/plain

…file content...
--3320-jfd90-3j2f90j233-f23--

WHERE:
```

- Somehost.com is the ebilling host URL
- 3320-jfd90-3j2f90j233-f23 is the multi-part boundary RFC2388

Successful Sample Response

```
HTTP/1.1 201 OK
Content-Type: application/json
{
"invoiceID": "INV2-RUVR-ADWQ-H89Y-ABCD",
"receivedDateTime": "2018-07-24 12:11:52 PDT"
}
```

Send Invoice Attachment

POST **`/v1/invoices/{invoiceID}/attachment`**

Send an attachment for a previously submitted Invoice by *invoiceID*. Multiple calls can be made for the same *invoiceID* to send multiple attachments for an Invoice. The attachment file can be optionally encrypted. NOTE: It is already required that all data be encrypted in transit, so this would be additional encryption of the attachment, which would then require a mechanism for the receiving system to be able to unencrypt the attachment. If encrypted, the encryption method is part of the system setup. Returns an *attachmentID*, *receivedDateTime,* and *errors* (see response below). The *attachmentID* may be used for subsequent API calls that are developed in the future.

Path Parameters

invoiceID string required

The ID of the invoice for which to associate the attachment. This is returned from the Send Invoice LEDES File call and the Send Invoice LEDES JSON call.

Request Body

attachmentFilename string required

The filename of the attachment. This can be the filename without mime type (ie. Invoice123Receipt1). This does not need to be unique and is probably just the filename of the attachment on the sending system side for reference. This should not include the network or folder names. An error should be issued if the filename is invalid, see error below.

fileMIMEType string required

The MIME type of the file.

NOTE: Here is a fairly good list of mime types:
https://www.iana.org/assignments/media-types/media-types.xhtml
https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types/Complete_list_of_MIME_types

attachmentType enum required

The type of attachment related to the Invoice.

Possible values: invoice_pdf, receipt, status_report, financial_summary, tax_authority _file, other

encrypted enum required

Whether the attachment file is encrypted or not.

Possible values: `Y, N`

file clob required

The invoice attachment file. This may be encrypted according to encrypted parameter. If encrypted, the encryption method is part of the system setup.
The receiving system may set a limit to the size of the file.

Response Body

A successful request returns the HTTP `201 Created` status code and a JSON response body that returns the invoice ID and date/time it was officially received. The errors property will be NULL for successful responses and will contain an array of error messages when a HTTP 400 Bad Request is returned.

attachmentID string required (if successful)

The receiving system's ID of the attachment. Read only.

receivedDateTime datetime (ISO8601) required

The audit date time for when the receiving system officially received the attachment.

---

errors Array [string] optional

If there are any errors, this element should be used to describe the errors.

Unsuccessful Responses

An unsuccessful request returns the HTTP 400 Error status code. Also see the section on standard REST Error codes at the end of this document.

In addition, the implementation should respond with these recommended errors.

These errors should be returned if any of the required request body fields are missing.

- `"error": "attachmentFilename required field missing."`
- `"error": "fileMIMEType required field missing."`
- `"error": "attachmentType required field missing."`
- `"error": "encrypted required field missing."`
- `"error": "attachmentFilename required field missing."`

These errors should be returned if any of the enum fields have an invalid format. Each implementation can decide which items to support and should list the supported values in the error message.

- `"error": "Invalid invoiceID."`
- `"error": "Invalid file size. Files size is limited to ..."`
  `WHERE … would be the size limited a file that the receiving system supports.`
- `"error": "invalid encrypted value. Supported values are Y and N."`
- `"error": "Invalid fileMIMEType value. Supported formats include ..."`
  `WHERE … would be a list of fileMIMEType's that the receiving system supports.`
- `"error": "attachmentFilename length too long. File name is limited to X characters."`
  `WHERE X would be the number of characters allowed in the file name that the receiving system supports.`
- `"error": "attachmentFilename is an invalid file name. A file name can't contain any of the following characters: ..."`

```
WHERE ... would be the set of characters not allowed in
the file names that the receiving system does not support.
```

● "error": "invalid attachmentType value. Supported values are invoice_pdf, receipt, status_report, financial_summary, tax_authority _file, other"

CURL Sample Request - Successful Attachment Submission

NOTE: The actual file is sent separately in a multi-part post.

```
curl -v -X POST
https://ServerURL/v1/invoices/INV2-RUVR-ADWQ-H89Y-ABCD/attachment \
-H "Authorization: XYZABC123" \
-F 'details={
     "attachmentFilename": "Receipt for Invoice",
     "fileMIMEType": "pdf",
     "attachmentType": "receipt",
     "Encrypted": "Y"
     };type=application/json’ \
-F 'file=@/path/to/file/Receipt\ for\ Invoice.pdf'
```

WHERE

- WHERE ‘/path/to/file/Receipt for Invoice.pdf’ is the path on the calling system’s server for the LEDES file. Also notice that spaces must be prefaced with a backslash (\).
- `XYZABC123` is the security access token
- `ServerURL` is the receiving servers URL

CURL Sample Response - Successful Attachment Submission

```
{
  "attachmentID": "INVA-RUVR-ADWQ-H89Y-ABCD",
  "receivedDateTime": "2018-07-24 12:11:52 PDT"
}
```

CURL Sample Request - Unsuccessful Invoice Submission

NOTE: The InvoiceID and attachmentType values are invalid.

```
curl -v -X POST
https://ServerURL/v1/invoices/INV2-RUVR-ADWQ-H89Y-XXXX/attachment \
-H "Authorization: XYZABC123" \
-F 'details={
      "attachmentFilename": "Receipt for Invoice",
      "fileMIMEType": "pdf",
      "attachmentType": "rec",
      "Encrypted": "Y"
      };type=application/json' \
-F 'file=@/path/to/file/Receipt\ for\ Invoice.pdf'
```

CURL Sample Response - Unsuccessful Attachment Submission

Note that the attachmentID was not returned, since there were errors in the call and no attachment was received. The receivedDateTime was returned to acknowledge when the call was responded to. The errors array property was returned with two errors.

```
{
  "receivedDateTime": "2018-07-24 12:11:52 PDT"
  "errors": [ {
      "error": "Invalid invoiceID."
    },{
   ●  "error": "invalid attachmentType value. Supported values are
      invoice_pdf, receipt, status_report, financial_summary,
      tax_authority _file, other"
    }
  ]
}
```

Get Invoice Status

GET **/v1/invoices/{*invoiceID*}**

Get the status of an invoice. Requires passing a valid *invoiceID*. The *invoiceID* is returned from the Send Invoice LEDES File call and the Send Invoice LEDES JSON call. The response is an InvoiceStatus object, which is a fairly complex object supporting a variety of information about the invoice.

Path Parameters

invoiceID string required

The ID of the invoice for which to show the status. This is returned from the Send Invoice LEDES File call and the Send Invoice LEDES JSON call.

Response Body

A successful request returns the HTTP `200 OK` status code and a JSON response body that returns the InvoiceStatus object. The errors property will be NULL for successful responses and will contain an array of error messages when a HTTP 400 Bad Request is returned.

invoiceStatus object InvoiceStatus required (if successful)

The InvoiceStatus object for the invoice. Read only.

errors Array [string] optional

If there are any errors when placing the Get Invoice Status call, this element should be used to describe the errors. This is NOT used for any errors in the invoice, which are returned in the invoiceStatus object. The one obvious error here would be that the invoiceID is invalid.

Unsuccessful Responses

An unsuccessful request returns the HTTP 400 Error status code. Also see the section on standard REST Error codes at the end of this document.

In addition, the implementation should respond with these recommended errors.

- `"error": "Invalid invoiceID."`

InvoiceStatus Object

This object describes a particular invoice's status and includes any invoice errors, adjustments, or payment data.

invoiceID string required

The receiving system's ID of the invoice returned by the Send Invoice call. Read only.

vendorInvoiceNumber string required

The original invoice number specified by the sending system. Read only.

status enum required

The invoice status. Each receiving system can decide what values they would like to return from the possible values documented below. Read only.

Status	**Description**
received	Receiving system has simply received the Invoice, however no file error processing or other actions have occurred at this time. The invoice will stay in this state until the file error checking has been completed.
file_error	There are errors in understanding the invoice data. The invoiceErrors array should list the errors. If no file errors are found then the status should be set to one of the following statuses, which will imply that the invoice data was understood. Even though there may be no file errors found, this does not mean there are no errors in the invoice and it still may be rejected due to some automated audit checks or manual checks by a user.
pending_client	The receiving system understands the invoice and it is pending review or future action by the client.
pending_tax_authority	The receiving system understands the invoice and it is pending approval from the tax authority, when that is required.
pending_vendor	The receiving system understands the invoice and the invoice is awaiting a response for some action from the Vendor. For example, if the client needs a receipt attachment to be submitted.
delivered_to_client	The receiving system has delivered the invoice to the client system. This optional status can be used when the invoices is passed to a separate client system.
rejected	The receiving system has rejected the invoice with errors. The invoiceErrors array should list the errors.
approved	The invoice was approved by the receiving system or by users and its awaiting payment.
sent_to_ap	The invoice was sent to an Accounts Payable system.
paid	The invoice was paid. The invoicePayment array should list any payment data. On some systems the receiving system may not receive payment data or even a paid status.

statusDateTime datetime (ISO8601) required

The date and time the Invoice Status was last set within the receiving system. Read only.

rejectionNote string optional

When an invoice is rejected, this can be used to describe the overall reason for the rejection. Note that the invoiceErrors element holds a more detailed description of each individual error and this element should duplicate that information. Read only.

originalTotal number optional

The original invoice total amount from the vendor. This should be a floating point number with two decimal places. This should always be included unless the LEDES files could not be parsed. Read only.

originalCurrency string optional

The currency for the original invoice total amount from the vendor. This should always be included unless the LEDES files could not be parsed. The three-character ISO-4217 currency code. Read only.

approvedTotal number optional

The invoice total amount approved by the client. For jurisdictions where clients can pay less than the originalTotal this is the amount the client plans to pay or has paid. The invoiceAdjustments section below should document the adjustments the client is imposing on the invoice. If the status is Paid, this should also match the invoicePayment data below. This should be a floating point number with two decimal places. Read only.

approvedCurrency string optional

The currency for the invoice total amount approved by the client. The three-character ISO-4217 currency code. Read only.

invoiceErrors Array [InvoiceError object] optional

An array of invoice errors. This element should contain data when the invoiceStatus is file_error or rejected. As various errors are identified in the invoice over time they should be accumulated here with each error recording the date and time it was identified. The InvoiceError object defined below describes the elements of an invoice error. Read only.

adjustments Array [InvoiceAdjustment object] optional

An array of adjustments applied to the invoice. For jurisdictions where adjustments to an invoice are allowed before payment, this is used to tell the vendor what adjustments are being applied to the invoice prior to the invoice being sent to AP for payment. For jurisdictions where adjustments are not allowed (typically where taxes are involved) this can be optionally used to describe the adjustments that need to be made prior to re-submitting the invoice. In this case the current invoice would be rejected and the adjustments would describe the necessary adjustments needed in a new invoice with the approvedTotal describing the total amount the client expecting to receive in the new invoice. When adjustments are recorded here the approvedTotal should equal the originalTotal minus the sum of all the adjustments. As various adjustments are identified in the invoice over time they should be accumulated here with each adjustment recording the date and time it was identified. Read only.

payments Array [InvoicePayment object] optional

The invoice payment information when the status is Paid. Read only.

InvoiceError Object

This object describes a particular error on an Invoice. An array of InvoiceError objects are referenced in the InvoiceStatus Object which is returned from the Get Invoice Status call if the invoice has an invoiceStatus of file_error or rejected.

errorType enum required

The type of error. Read only.
Possible values: file_structure, missing_field, bad_file_data, invoice_level_error, line_item_error, audit_error, afa_error

datetime datetime (ISO8601) required

The date and time the error was identified. Read only.

errorCode enum required

The error code defined in the LEDES Error Codes documentation. Read only.
Possible values: (See LEDES Error Codes)

errorName string required

A short name of the error from the LEDES Error Codes documentation. Read only.

errorDescription string required

A description of the error. The LEDES Error Codes documentation has recommended descriptions and some errors may have additional information specific to the error. This can also be used to hold a user's manually entered reason for an error in the invoice. Read only.

---

lineItem LineItem object required for LineItemError

If the errorType is LineItemError this is required and it identifies the line item from the original LEDES file that contains the error. It should contain the original values of the line item to allow the time and billing system to identify what line item contains the error. Read only.

InvoiceAdjustment Object

This object describes a particular adjustment on an Invoice. An array of InvoiceAdjustment objects are referenced in the Invoice Status Object which is returned from the Get Invoice Status call if the invoice has any adjustments applied.

---

adjustmentType enum required

The type of adjustment. Read only.
Possible values: InvoiceLevelAdjustment, LineItemAdjustment

---

datetime datetime (ISO8601) required

The date and time the adjustment was identified. Read only.

---

adjustmentAmount money required

The adjustment amount. A positive value represents a reduction in the invoice. Read only.

---

adjustmentCurrency enum required

The currency of the adjustment. Read only.

---

adjustmentReason string required

A description of why the adjustment was applied. Read only.

---

originalLineItem LineItem object required for LineItemAdjustment

If the adjustmentType is LineItemAdjustment this is required and this identifies the line item from the original LEDES file that is being adjusted. It should contain the original values of the line item to allow the time and billing system to know what line item is being adjusted. Read only.

---

adjustedlLineItem LineItem object required for LineItemAdjustment

If the adjustmentType is LineItemAdjustment this is required and describes the changes to the line item that is being adjusted. This allows the time and billing system to know what exactly is being adjusted. Read only.

LineItem Object

This object describes an invoice line item and is used within the InvoiceError object when there is a line item error or within the InvoiceAdjustment object when there is a line item adjustment. The properties for this object are very conditional depending on their use. When the LineItem object is being used to simply refer to a specific line item, for example InvoiceError.lineItem or InvoiceAdjustment.originalLineItem, then if the receiving system has maintained the LEDES line item number from the original invoice it can send back just the lineItemRef property to uniquely identify the line item. If however, the receiving system does not maintain the LEDES line item number from the original invoice it must send back the original values of the rest of the LineItem object properties for the time and billing system to identify the line item. When the LineItem object is being used to specify a line item adjustment, for example InvoiceAdjustment.adjustedLineItem, then the receiving system should send back just the properties of the LineItem object that have been changed in the adjustment. Read only.

---

lineItemRef number conditional

This is the line item number from the original LEDES file and should be set if possible. This is the file_item_nbr field in LEDES XML or the LINE_ITEM_NUMBER field in LEDES98B/BI. If the receiving system hasn't maintained the original LEDES line number reference, then a value of -9999 should be returned and the rest of the line item fields should be set for the time & billing application to understand which line item is being referenced. This property is not used when specifying a line item adjustment in the case of InvoiceAdjustment.adjustedLineItem since the reference to the line item is specified with InvoiceAdjustment.originalLineItem. Read only.

---

lineItemType enum conditional

The type of line item. Read only.
Possible values: fee, expense

---

chargeDate date (ISO8601) conditional

The date of the line item. The date the service was performed. Read only.

---

timekeeperID number conditional

The timekeeper ID for the line item. This is required for a Fee and optional for an Expense. Read only.

---

chargeDescription string conditional

A narrative description of the service or fee. Read only.

---

taskCode string conditional

Task code for this line item. For example, “L110”. Read only.

---

activityCode string conditional

Activity code for this line item. For example, “A101”. Read only.

---

expenseCode string conditional

Expense code for Expense line items only. Read only.

---

units number conditional

The units for the line item. For a Fee this is the number of hours billed on the line item. For an Expense this is the number of units. Read only.

---

rate money conditional

The rate for the line item. For a Fee this is the timekeeper’s hourly rate. For an Expense this is the unit amount. Read only.

---

baseAmount money conditional

The base amount (baseAmount = units * rate) for the line item prior to any adjustments or discounts. Read only.

---

itemDiscountCreditAmount money conditional

The items discount or credit amount. Read only.

---

totalAmount money conditional

The final billed value (totalAmount = baseAmount + itemDiscountCreditAmount) for the line item including any adjustments or discounts. Read only.

InvoicePayment Object

This object describes payment information for an Invoice. An array of InvoicePayment objects are referenced in the Invoice Status Object which is returned from the Get Invoice Status call if the invoice has any adjustments applied.

---

paymentType enum required

The type of payment. Read only.
Possible values: Check, Wire, ACH, CreditCard

---

datetime datetime (ISO8601) required

The date and time of the payment. Read only.

---

paymentAmount money required

The payment amount. Read only.

---

paymentCurrency enum required

The currency of the payment. Read only.

---

paymentRef string required

The reference information for the payment, for example, the check number. For Wire and ACH this should be the confirmation number. Read only.

---

payee string required

The entity to whom the payment was made. Read only.

---

paidToAccount string required

The account to which the payment was made for paymentType's of Wire, ACH, and Credit. Read only.

CURL Sample Request - Successful Status Request

```
curl -v -X GET https://ServerURL/v1/invoices/INV2-RUVR-ADWQ-H89Y-ABCD
\
-H "Content-Type: application/json" \
-H "Authorization: XYZABC123"
```

WHERE
- `XYZABC123` is the security access token
- `ServerURL` is the receiving servers URL

CURL Sample Response - Successful Received Status

In this example, the LEDES files was simply received by the receiving system and not further processing has occurred yet.

```
{
   "invoiceID": "INV2-RUVR-ADWQ-H89Y-ABCD",
   "vendorInvoiceNumber": "96542",
   "status": "received",
   "statusDateTime": "2018-07-24 12:11:52 PDT",
   "originalTotal": "21450.00",
   "originalCurrency": "USD",
   "approvedAmount": "",
   "approvedCurrency": "USD",
   "invoiceErrors": [
   ],
   "adjustments": [
   ],
   "payments": [
   ]
}
```

CURL Sample Response - Successful File Error Status

In this example two errors, an invoice level error and a line item error, are returned and the receiving system has maintained the LEDES Line Number Reference so it uses just that to refer to the line item.

```
{
   "invoiceID": "INV2-RUVR-ADWQ-H89Y-ABCD",
   "vendorInvoiceNumber": "96542",
   "status": "file_error",
   "statusDateTime": "2018-07-24 12:11:52 PDT",
   "originalTotal": "21450.00",
   "originalCurrency": "USD",
   "approvedAmount": "",
   "approvedCurrency": "USD",
   "invoiceErrors": [
      {
         "errorType": "missing_field",
         "datetime": "2018-07-24 12:12:12 PDT",
         "errorCode": "MF104",
         "errorName": "Invoice Date Missing",
         "errorDescription": "The invoice date is missing and is a
required field."
      },
         "errorType": "bad_file_data",
         "datetime": "2018-07-24 12:13:22 PDT",
         "errorCode": "BD214",
         "errorName": "Invalid Timekeeper",
         "errorDescription": "Line item #6 has an invalid timekeeper
ID: 12345",
         "lineItem": {
            "lineItemRef": "6"
         }
      }
   ],
   "invoiceAdjustments": [
   ],
   "invoicePayments": [
   ]
}
```

CURL Sample Response - Successful File Error Status

In this example two errors, an invoice level error and a line item error, are returned and the receiving system hasn't maintained the LEDES Line Number Reference so it needs to send back the full LineItem Object to refer to the line item. It returns -9999 as the `lineItemRef`.

```
{
   "invoiceID": "INV2-RUVR-ADWQ-H89Y-ABCD",
   "vendorInvoiceNumber": "96542",
   "status": "file_error",
   "statusDateTime": "2018-07-24 12:11:52 PDT",
   "originalTotal": "21450.00",
   "originalCurrency": "USD",
   "approvedAmount": "",
   "approvedCurrency": "USD",
   "invoiceErrors": [
      {
         "errorType": "missing_field",
         "datetime": "2018-07-24 12:12:12 PDT",
         "errorCode": "MF104",
         "errorName": "Invoice Date Missing",
         "errorDescription": "The invoice date is missing and is a
required field."
      },
         "errorType": "bad_file_data",
         "datetime": "2018-07-24 12:13:22 PDT",
         "errorCode": "BD214",
         "errorName": "Invalid Timekeeper",
         "errorDescription": "Line item #6 has an invalid timekeeper
ID: 12345",
         "lineItem": {
            "lineItemRef": "-9999"
            "lineItemType": "fee"
            "chargeDate": "2018-07-24",
            "timekeeperID": "12345",
            "chargeDescription": "Review of Deposition",
            "taskCode": "L110",
            "activityCode": "A101",
            "expenseCode": "",
            "units": "3",
            "rate": "300.00",
            "baseAmount": "900.00",
            "itemDiscountCreditAmount": "0.00",
            "totalAmount": "900.00"
         }
      }
   ],
   "invoiceAdjustments": [
   ],
   "invoicePayments": [
   ]
}
```

CURL Sample Response - Successful Pending Client Status

In this example, the LEDES files was accepted and it is pending review or future action by the client.

```
{
   "invoiceID": "INV2-RUVR-ADWQ-H89Y-ABCD",
   "vendorInvoiceNumber": "96542",
   "status": "pending_client",
   "statusDateTime": "2018-07-24 12:11:52 PDT",
   "originalTotal": "21450.00",
   "originalCurrency": "USD",
   "approvedAmount": "",
   "approvedCurrency": "USD",
   "invoiceErrors": [
   ],
   "adjustments": [
   ],
   "payments": [
   ]
}
```

CURL Sample Response - Successful Pending Vendor Status

In this example, the LEDES files is awaiting a response for some action from the Vendor. For example, if the client needs a receipt attachment to be submitted.

```
{
   "invoiceID": "INV2-RUVR-ADWQ-H89Y-ABCD",
   "vendorInvoiceNumber": "96542",
   "status": "pending_vendor",
   "statusDateTime": "2018-07-24 12:11:52 PDT",
   "originalTotal": "21450.00",
   "originalCurrency": "USD",
   "approvedAmount": "",
   "approvedCurrency": "USD",
   "invoiceErrors": [
   ],
   "adjustments": [
   ],
   "payments": [
   ]
}
```

CURL Sample Response - Successful Delivered to Client Status

In this example, the LEDES files was delivered to the client system. This optional status can be used when the invoices is passed to a separate client system.

```
{
   "invoiceID": "INV2-RUVR-ADWQ-H89Y-ABCD",
   "vendorInvoiceNumber": "96542",
   "status": "delivered_to_client",
   "statusDateTime": "2018-07-24 12:11:52 PDT",
   "originalTotal": "21450.00",
   "originalCurrency": "USD",
   "approvedAmount": "",
   "approvedCurrency": "USD",
   "invoiceErrors": [
   ],
   "adjustments": [
   ],
   "payments": [
   ]
}
```

CURL Sample Response - Successful Rejected Status

In this example two errors, an invoice level error and a line item error, are returned and the receiving system hasn't maintained the LEDES Line Number Reference so it needs to send back the full LineItem Object to refer to the line item. It returns -9999 as the `lineItemRef`.

```
{
   "invoiceID": "INV2-RUVR-ADWQ-H89Y-ABCD",
   "vendorInvoiceNumber": "96542",
   "status": "rejected",
   "statusDateTime": "2018-07-24 12:11:52 PDT",
   "originalTotal": "21450.00",
   "originalCurrency": "USD",
   "approvedAmount": "",
   "approvedCurrency": "USD",
   "invoiceErrors": [
      {
         "errorType": "InvoiceLevelError",
         "datetime": "2018-07-24 12:12:12 PDT",
         "errorCode": "IE209",
         "errorName": "Invoice Exceeds Agreed upon Fixed Amount",
         "errorDescription": "The Invoice Total is incorrect. Received
Invoice Total=21450; Fixed Fee Amount=21400"
      },
      {
         "errorType": "LineItemError",
         "datetime": "2018-07-24 12:13:22 PDT",
         "errorCode": "LE127",
         "errorName": "Incorrect Line Item Total",
         "errorDescription": "Line Item #5 has an incorrect hourly
rate. The rate for this timekeeper should be $600",
         "lineItem": {
            "lineItemRef": "5"
            "lineItemType": "Fee"
            "chargeDate": "2018-07-24",
            "timekeeperID": "695",
            "chargeDescription": "Review of Deposition",
            "taskCode": "L110",
            "activityCode": "A101",
            "expenseCode": "",
            "units": "4",
            "rate": "350.00",
            "baseAmount": "3475.00",
            "itemDiscountCreditAmount": "0.00",
            "totalAmount": "3475.00"
         }
      }
   ],
   "adjustments": [
   ],
   "payments": [
   ]
}
```

CURL Sample Response - Approved Status

This example shows a response for an invoice that has been approved for payment with no errors or adjustments. The invoice has not yet been sent to AP for payment.

```
{
   "invoiceID": "INV2-RUVR-ADWQ-H89Y-ABCD",
   "vendorInvoiceNumber": "96542",
   "status": "Approved",
   "statusDateTime": "2018-07-24 12:11:52 PDT",
   "originalTotal": "21450.00",
   "originalCurrency": "USD",
   "approvedAmount": "21450.00",
   "approvedCurrency": "USD",
   "invoiceErrors": [
   ],
   "adjustments": [
   ],
   "payments": [
   ]
}
```

Sample Response - Approved (with Adjustments)

This example shows a response for an invoice that has been approved for payment with no errors, however there is a line item adjustment applied. The invoice has not yet been sent to AP for payment.

```
{
   "invoiceID": "INV2-RUVR-ADWQ-H89Y-ABCD",
   "vendorInvoiceNumber": "96542",
   "status": "approved",
   "statusDateTime": "2018-07-24 12:11:52 PDT",
   "originalTotal": "21450.00",
   "originalCurrency": "USD",
   "approvedAmount": "20377.50",
   "approvedCurrency": "USD",
   "invoiceErrors": [
   ],
   "adjustments": [
      {
         "adjustmentType": "LineItemAdjustment",
         "datetime": "2018-07-24 12:12:12 PDT",
         "adjustmentAmount": "200.00",
         "adjustmentAmountCurrency": "USD",
         "adjustmentReason": "Services should have been performed by
an Associate instead of a Partner."
         “originalLineItem” : {
            "lineItemRef": "5"
         }
         “adjustedLineItem” : {
            "lineItemRef": "5"
            "lineItemType": "Fee"
            "chargeDate": "2018-07-24",
            "timekeeperID": "395",
            "chargeDescription": "Review of Deposition",
            "taskCode": "L110",
            "activityCode": "A320",
            "units": "4",
            "rate": "300.00",
            "baseAmount": "1200.00",
            "itemDiscountCreditAmount": "0.00",
            "totalAmount": "1200.00"
        }
      }
   ],
   "payments": [
   ]
}
```

CURL Sample Response - Sent to AP Status

This example shows a response for an invoice that has been approved for payment with no errors or adjustments. The invoice has been sent to AP for payment, but has not yet been paid.

```
{
   "invoiceID": "INV2-RUVR-ADWQ-H89Y-ABCD",
   "vendorInvoiceNumber": "96542",
   "status": "sent_to_ap",
   "statusDateTime": "2018-07-24 12:11:52 PDT",
   "originalTotal": "21450",
   "originalCurrency": "USD",
   "approvedAmount": "21450",
   "approvedCurrency": "USD",
   "invoiceErrors": [
   ],
   "adjustments": [
   ],
   "payments": [
   ]
}
```

Sample Response - Paid

This example shows a response for an invoice that has been approved for payment with no errors or adjustments. The invoice has been sent to AP for payment and has been paid with the payment information detailed.

```
{
   "invoiceID": "INV2-RUVR-ADWQ-H89Y-ABCD",
   "vendorInvoiceNumber": "96542",
   "status": "Approved",
   "statusDateTime": "2018-07-24 12:11:52 PDT",
   "originalTotal": "21450.00",
   "originalCurrency": "USD",
   "approvedAmount": "21450.00",
   "approvedCurrency": "USD",
   "invoiceErrors": [
   ],
   "adjustments": [
   ],
   "payments": [
     {
       "paymentType": "Check",
       "datetime": "2018-08-21 10:12:12 PDT",
       "paymentAmount": "21450.00",
       "paymentCurrency": "USD",
       "paymentRef": "3384455",
       "payee": "Moose and Squirrel",
       "paidToAccount": ""
     }
   ]
}
```

Get Invoice Status Changes

`GET`
`/v1/invoices/statusChanges?invoiceStatusMarker={invoiceStatusMarker}`

Get the status of a set of invoices. If no *invoiceStatusMarker* is passed in then this call returns the status of all submitted invoices plus a *invoiceStatusMarker* for subsequent calls. When passing in a *invoiceStatusMarker* this call will return the status to all invoices that have status updates since that marker was issued. The calling system must save this marker for subsequent calls. This call is optional and is not a requirement for the API.

Path parameters

invoiceStatusMarker string optional

The marker returned from a previous call to invoices/statusChanges.

Successful Response Body

A successful request returns the HTTP `201 Created` status code and a JSON response body that returns an array of invoicestatus object listing all the invoices and their statuses and the invoiceStatusMarker..

invoiceStatusList	Array [object InvoiceStatus]	required

An array of invoicestatus object for the invoice who's status have changed. Read only.

invoiceStatusMarker	string	required

The invoiceStatusMarker is used in the next call to Get Invoice Status Changes. This is an opaque marker to identify to the system the last set of statuses received by the consuming system. Read only.

Unsuccessful Response

An unsuccessful request returns the HTTP 400 Error status code. The implementation should respond with this error if the invoiceStatusMarker is invalid.

Invalid invoiceStatusMarker Error

```
"error" : "Invalid invoiceStatusMarker."
```

CURL Sample Request - All Invoice Statuses

This first example does not pass in a invoiceStatusToken so the response would be all invoice statuses.

```
curl -v -X POST /v1/invoices/statusChanges \
-H "Authorization: <Access-Token>"
```

CURL Sample Request - Changed Invoice Statuses

This second example passes in a invoiceStatusToken so the response would be all invoice statuses for just the invoices with a status change from the time the invoiceStatusToken was given.

```
curl -v -X POST /v1/invoices/statusChanges?token=AC39F29 \
-H "Authorization: <Access-Token>"
```

CURL Sample Response

```
{ "invoiceStatusList":
[{
   "invoiceID": "INV2-RUVR-ADWQ-H89Y-ABCD",
   "vendorInvoiceNumber": "96541",
   "invoiceStatus": "PendingClient",
   "invoiceStatusDateTime": "2018-07-24 12:11:52 PDT",
   "invoiceOriginalTotal": "21450.00",
   "invoiceOriginalCurrency": "USD",
   "invoiceApprovedAmount": "",
   "invoiceApprovedCurrency": "USD",
   "invoiceErrors": [
   ],
   "invoiceAdjustments": [
   ],
   "invoicePayments": [
   ]
}, {
   "invoiceID": "INV2-RUVR-ADWQ-H89Y-1234",
   "vendorInvoiceNumber": "96542",
   "invoiceStatus": "Approved",
   "invoiceStatusDateTime": "2018-07-24 12:11:52 PDT",
   "invoiceOriginalTotal": "21450.00",
   "invoiceOriginalCurrency": "USD",
   "invoiceApprovedAmount": "21450.00",
   "invoiceApprovedCurrency": "USD",
   "invoiceErrors": [
   ],
   "invoiceAdjustments": [
   ],
   "invoicePayments": [
      {
         "paymentType": "Check",
         "datetime": "2018-08-21 10:12:12 PDT",
         "paymentAmount": "21450.00",
         "paymentCurrency": "USD",
         "paymentRef": "3384455",
         "payee": "Moose and Squirrel",
         "paidToAccount": ""
      }
   ]
}
],
 "invoiceStatusMarker" : "AC39F29"
}
```

Common Objects

[Internet date and time format]

HTTP Status Codes & Errors

Every REST API request should return a success or error HTTP status code. The HTTP status code standard categorized errors into 3 types. The 2xx status codes represent that a request was processed successfully, a 4xx return status code indicates an error was encountered while processing a request. A 5xx HTTP status codes indicates infrastructure errors that prevented the request from being processed.

Useful references:

- https://restfulapi.net/http-status-codes/
- https://developer.paypal.com/docs/api/overview/#error-status-codes

400 Response Errors Explained

An unsuccessful request returns the HTTP 400 Error status code.

The implementation should respond with missing required fields or invalid field value types.

With the client's credentials:

- A 401 error response indicates the request has not been applied because it lacks valid authentication credentials for the target resource.
- A 403 error response indicates the server understood the request but refuses to authorize it.

Subcommittee Meetings & Recordings

2/14/18 - **Play recording**

3/14/18 - **Play recording** (1 hr 13 min 17 sec)

4/11/18 - **Play recording** (1 hr 9 min 15 sec)

5/18 - meeting was not recorded

6/13/18 - **Play recording** (1 hr 30 min 19 sec)

7/12/18 - **Play recording** (3 hr 4 min 48 sec)

8/8/18 - **Play recording** (1 hr 14 min 49 sec)

9/12/18 - **Play recording** (1 hr 14 min 25 sec)

10/10/18 - **Play recording** (1 hr 5 min 2 sec)

Attendees: Nick Puschak; David Vasquez; Jane Bennitt; Gwen Hess; Sajeel Malani; Sherry Askin; Heather M. Magers Keefe; Nagendra; Tony Fadulu; Erik Martin
Topics: Added a CURL example for calling the Send LEDES Invoice call with the Multi-part post. Reviewed some Error Codes and Descriptions and how they would map into the InvoiceError object. Added a new field "Error Name" to the InvoiceError object.

11/14/18 - Play recording (1 hr 1 min 58 sec)

Attendees: Nick Puschak; David Vasquez; Jane Bennitt; Kris; Sajeel Malani; Sherry Askin
Topics: Reviewed Get Invoice Status Change call and decided to remove the Get Initial Status Token call.

12/12/2018 - **Play recording** (1 hr 22 min 42 sec)

Attendees: Nick Puschak; David Vasquez; Sajeel Malani; Sherry Askin; Heather Magers; Gwen Hess

Topics:Discussed security, HTTP Request/Response examples, REST errors.

1/9/2019 - **Play recording** (1 hr 13 min 36 sec)

Attendees: Nick Puschak; Sherry Askin; Gwen Hess

Topics:Discussed Sherry's rework of security sections R2. Added new invoiceType, relatedInvoiceID, and comment elements to Send Invoice LEDES file.

2/13/2019 - **Play recording** (51 min 25 sec)

Attendees: Nick Puschak; Sherry Askin; Gwen Hess. Jane Bennitt

Topics:Security. We first thought to make it required to have all LEDES files and attachment files encrypted, but decided to keep it optional for now and leave the encrypted parameter in the SendLEDESFile and SendAttachment calls.

3/20/2019 - **Play recording** (1 hr 22 min 19 sec)

Attendees: Nick Puschak; Sherry Askin; Gwen Hess. Jane Bennitt, David Vasquez, Sajeel Malani

Topics:Reviewed changes that I made after a review of the API with a technical resource knowledgeable with REST API. This included change all enumerations to lowercase and underscores; removing repetitive use of "invoice" in the Invoice Status object properties; added an error property to the returned object to pass back errors; removed /status from the Invoice Status API call. Sherry Askin reviewed all the changes to the security section.