Overview

The purpose of this documentation is to provide information about how EDGAR filers may use Application Programming Interfaces ("APIs") to check EDGAR system status, manage users, submit EDGAR filings, and check the status of submitted filings. (For information on how developers may use APIs to access EDGAR submissions by company and extracted XBRL data, see our resources regarding data.sec.gov.)

These APIs are based on REST semantics with resource names that are based on the verbs or actions that are being performed on discrete resources within the system. In general these APIs use JSON for request and response communication and binary or XML requests for filing submissions.

These APIs make use of standard HTTP semantics including various HTTP methods defined by the HTTP RFC as well as response codes as maintained by the Internet Assigned Numbers Authority (IANA).

Tokens

The EDGAR APIs make use of HTTP Bearer authentication with tokens provided by Filer Management. There are two types of tokens: the Filer API Token and the User API Token (collectively: API Tokens).

API Tokens provide access to the EDGAR API and should be stored and handled securely. The SEC will never ask for the contents of a Token.

These API Tokens are issued by the Filer Management application. The Filer API Token is generated by a Technical Admin and should be provided to applications that require it for communication with EDGAR. The User API Token is generated by individuals that will be creating or submitting filings to EDGAR. The User API Token should be provided to applications that will make use of the token when communicating with EDGAR on behalf of the individual.

The API Tokens are JWT tokens in JOSE format with an encrypted payload that was signed before it was encrypted. For more information on the JWT format see the documentation at JWT.io.

A JWT token is presented as a series of base64 stanzas separated by a "." character. The first stanza, the header, is not encrypted and contains information that may be useful to application or integration teams. The example JWT header, given below, is the first stanza from a sample Filer API Token.

```
ewogICJjaWsiOiAiMDAwMDAwMDAwMCIsCiAgImtpZCI6ICI3NWM4ZDM3OS0yZDgyLTQxNW
UtYTNjNS03NTAxNjk5ZTMxMDEiLAogICJhbGciOiAiRUNESC1FUyIsCiAgImV4cGlyZXNB
dCI6ICIyMDI1LTA3LTI1VDAzOjAwOjAwWiIKfQ==
```

When decoded this header contains some useful information for both the EDGAR API and applications integrating with the EDGAR API.

```
{
  "cik": "0000000000",
  "kid": "75c8d379-2d82-415e-a3c5-7501699e3101",
  "alg": "ECDH-ES",
  "expiresAt": "2025-07-25T03:00:00Z"
}
```

In the case of the Filer API Token you can see that it contains the key ID ("kid") and the algorithm ("alg") used to encrypt the payload. This is required by the server to decode the JWT provided to the EDGAR API.

In this header the CIK ("cik") and expires at ("expiresAt") fields are present in addition to the key ID and algorithm. The CIK is the CIK that the Filer API Token is assigned to and can be used by applications to ensure that the Filer API Token matches the organization that is communicating with the EDGAR API. Optionally, filers could also specify a login CIK if there is a delegation from the EDGAR account corresponding to the login CIK to the filer API token EDGAR account/CIK. Delegations can be established on the Filer Management dashboard under “Manage Delegations and User Groups.” For more information on establishing delegations, see How Do I “Understand and Manage Delegation.”

The expires at ("expiresAt") field allows applications to check the expiration status of a token without making a call to the EDGAR API. This does not check if the token has been manually revoked but allows applications to check the status of tokens locally.

The User API Token contains similar information in the same format. There is useful information contained in the header stanza that can be used by applications to determine information about the token.

```
{
  "kid": "75c8d379-2d82-415e-a3c5-7501699e3101",
  "alg": "ECDH-ES",
  "userId": "0ebc7681-2b96-42a8-8e76-efd8c23b1b13",
  "expiresAt": "2024-08-24T03:00:00Z"
}
```

The User ID is also contained within the header. This gives applications the capability to uniquely identify the token. The expires at ("expiresAt") field also gives applications the ability to determine if the token should still be valid without calling the EDGAR API. This only checks for expiration and the token could still be revoked within the EDGAR system.

Token Expiration

To prevent expiring tokens from interrupting the filing process the Filer API Tokens and the User API Tokens are set to expire on weekdays and outside of business hours. This means that each token will be issued for the time explained in the table below and then it will be checked to ensure that it does not expire on a day that EDGAR is not operating.

Token Type	Minimum Validity
Filer API Token	1 year
User API Token	30 days

Note: the system cannot account for holidays that have not been determined in the schedule at the time the token is issued.

Authentication

Authorize 🔓

In order to authenticate to the EDGAR APIs the token or tokens that will be used must be added to the "Authorization" header with the "bearer " prefix. For example: given a token like 'abc123' the contents of the "Authorization" header would be "bearer abc123". In the event that two tokens are being used they should be separated by a "," or a " ". For example: given the Filer API Token "filer123" and the User API Token "user345" the contents of the "Authorization" header would be "bearer filer123,user345".

The EDGAR system will calculate authorization to use API resources based on the tokens provided. The CIK represented by the Filer API Token and the individual represented by the User API token are used to determine if the request can be completed by the system.

In the OpenAPI Specification that the EDGAR APIs use each of the two possibilities for presenting tokens is covered by a Security Scheme. Each API method uses either the "Filer API Token" security scheme or the "Filer API Token and User API Token" security scheme. Each of these schemes expect a token (or tokens) issued by the Filer Management application and presented in the way described above.

To test APIs with Authorization you can click the **Authorize** button above or click the "lock" icon next to any API resource. This will allow you to set the "bearer" authentication for any requests that use the same security scheme.

Authentication Errors

The EDGAR API methods that require Token authentication will return status messages when the token(s) sent with the request are not valid for use with the EDGAR APIs. The intent of these messages is to narrow down the range of problems to assist in removing invalid tokens or updating the contents of tokens that are not valid.

Each message will include the index of the token implicated in the issue as part of the message. The index starts at "1". If two tokens are sent then "1" would represent what is identified as the first token sent and "2" would be the second token.

If the request is prevented from authenticating (and receiving a 401 or 403 status code) messages will generally be of the level "ERROR". If the request authenticates these messages will be returned as "NOTICE" level messages with the response.

Sample Error	Recommend Action
token {index} is not in the format expected by this application, check the contents of the token	This indicates that the token provided is not in the JWE (JSON Web Encryption) format required of JWT tokens used with this application. This could be due to a partial token being provided to the application or other issues transferring or copying the token.
token {index} is missing a required header field ({field name})	The JWE header has been modified in a way that makes it incompatible with the EDGAR APIs. Token modification makes the token invalid.
token {index} does not have required header fields for verification	The JWE header has been modified in a way that makes it incompatible with the EDGAR APIs. Token modification makes the token invalid.
token {index} is not valid for use with this application	The token does not match the expectations of a token created for use with this system. This could mean that the token is being presented to the wrong environment (ex: a token created for a test environment is being used against the production system) or the token is from a different application altogether.
token {index} of type {type} is either expired or has been revoked, check Filer Management for more details	The token that came with this request is no longer valid within the system. This usually indicates that the token has expired or has been revoked. Checking Filer Management is the only way to determine the status of the individual token. The 'type' value will either be 'Filer API Token' or 'User API Token'.
only one token of a given type can be used at a time, token {index} is of the same type ({type}) as another token	A token that was presented was the same type as a token parsed earlier. The EDGAR APIs can only accept one token of a given type (Filer API Token, User API Token) with each request. The first valid token presented with a given request and of a unique type will be the token used to process the request. This message is intended to provide guidance if multiple tokens of the same type are sent. The 'type' value will either be 'Filer API Token' or 'User API Token'.

If, for example, the request contained a valid Filer API Token and an invalid User API Token in the header a message like the following could be returned:

```
{
  "tracking": "38ab654e20162523396d447d22ae8887",
  "locator": "4n05cc",
  "messages": [
    {
      "type": "ERROR",
      "content": "Forbidden"
    },
    {
      "type": "ERROR",
      "content": "token 2 is not in the format expected by this applic
ation, check the contents of the token"
    }
  ]
}
```

This is a 403 error because, while some valid credentials were submitted with the request, those credentials alone did not meet the requirements to access the resource that was requested. The system cannot accept the second token (token index 2) either because the token is in an invalid format or was not created for use with the application it was sent to. A message is returned at the error level for both the http "Forbidden" response and for the token parsing failure.

Communication

A typical request to an EDGAR API would be made over HTTP, supplying an API Token, and would contain either JSON or XML contents. The following simple GET request is to get the status of the EDGAR system.

```
GET /status HTTP/1.1
Authorization: bearer eyJjaWsiOiIwMDAxMzE2MDAzIiwia2lkIjoiOThiOTg0ZTk1
NTk5Y2Y3YWNlMDRjYzk1N2U2OWE1MDIiLCJhbGciOiJFUzUxMiIsImV4cGlyZXNBdCI6Ij
IwMjUtMDctMjVUMDM6MDA6MDBaIiwidHlwIjoiSldUIn0.eyJKV1QiOiJFTkNSWVBURUQg
VkFMVUUifQ.AeF5hGsaJnQQEZUk-OBfVUzf-nuu5zJSn9MSK5Y7eSlj0RiJ_7jKKUA4J11
I61tfe5TOBxrnhwqMiWuzdl3gtJnyAIqBeF-G5_A93EUbI3ttW_pUXQupKsdmW1lDnHyS3
ciGbCfkeXz28VWSufMLORqzq1wi-EzgpbT9cFrZR7YGMOj9
Accept: application/json
```

A typical EDGAR API response will include an HTTP status code as well as a response body. The HTTP status code is the primary indicator of the status of a response. In general: a response in the 200-299 range is a "success" response, a response in the 400-499 range represents that something is wrong with the request, and anything in the 500-599 range means that something went wrong while processing the request.

The response body that returns from the API will include a tracking ID ("tracking") that can be used when communicating with the Help Desk to identify the particular request in question. The response will also include a property called "locator" that can also assist the Help Desk in finding the record in the EDGAR system.

```
200 OK
Content-Type: application/json

{
  "tracking": "5fdac24eb9160787519516fde4499652",
  "locator": "1fecde",
  "message": "EDGAR is operating normally. All submissions will receive today’s filing date.",
  "condition": "ACCEPTING"
}
```

In addition to the tracking ID and locator there are other elements that can be found among responses. The **messages** can contain information that can indicate further information about a response. These are intended to be supplemental to any HTTP code that is returned.

Request Headers

During communication with EDGAR API resources the only explicitly required HTTP request header is the "Authorization" header used to communicate token information. The EDGAR APIs are entirely stateless and do not require any other request headers or cookies to be sent with the request.

When using Submission API resources (such as Single Test Filing or Single Live Filing) the client can also provide the "Expect" header with the content "100-continue." This will allow the EDGAR API resource to respond more efficiently to the request and evaluate the request before the client sends the full body of the request. More information on the "Expect" header and the functionality of "100-continue" can be found in the HTTP RFC.

Rate Limiting

The EDGAR API resources may rate-limit specific resources based on the needs of the EDGAR system. In general the Developer Resources provide for certain rate limits and behaviors of clients accessing EDGAR systems. The specific rate limits for the EDGAR API resources are subject to change.

In the event that a rate limit is exceeded the 429 (Too Many Requests) status code will be returned. More information on this code can be found in the status code table.

Some resources will return response headers indicating a caching status through the use of the "ETag" header, "Last-Modified" header, or the "Cache-Control" header with "max-age" directive. Clients should be aware of these headers and use them to gauge how long they should wait between requests for a specific resource.

Response Codes

The APIs use common HTTP status response codes as registered with the IANA. The following table shows the common codes and when/how they are generally used in the EDGAR APIs

Code	Status	Description
200	**OK**	The API resource was accessed successfully. Messages may be attached to the response for additional context.
202	**Accepted**	The API resource was accessed successfully. This status code is used by the submission API to indicate that the submission has been transmitted to EDGAR and that it has an Accession Number. In the HTTP standard this response is used to indicate that there is further processing that will be done on the resource after the response.
204	**No Content**	The API resource that was requested is empty. This can be used when a list of items is requested but no items were found to

Code	Status	Description
		populate the list. A 204 response is empty and will include no JSON and cannot be parsed.
400	**Bad Request**	The API resource was not able to process the request because of an error with the request. The messages returned with this request will provide more insight into the nature of the error.
401	**Unauthorized**	The request made to the API resource was not able to be authorized. This usually means that invalid token material was provided with the "Authorization" header. This could mean a partial or corrupted token or a token that has expired.
403	**Forbidden**	The request made to the API has been authorized, meaning that the tokens presented contain enough information to identify the Filer and the User (if required), but that identifying information does not have the rights or permission to access the resource.

Code	Status	Description
		There are several reasons this could happen. The most common is when making a submission and the Filer API Token and User API Token combination does not have delegated permissions for the CIK or Login CIK combination. A user that is locked in Filer Management will also cause a "Forbidden" response. Another scenario that produces "Forbidden" is requesting the status of a submission with a Filer API Token that does not match the CIK or Login CIK of the submission and does not have delegation to either.
405	**Method Not Allowed**	Typically seen when using the wrong method to send data to an API resource. Each resource responds to either a single specific method or a set of methods. Using the wrong method, like PUT instead of POST, will cause this response.

Code	Status	Description
406	**Not Acceptable**	Each API resource expects the client to accept a response in "application/json" format through the use of the "Accept" header. Any other value in the "Accept" header will cause the API resource to respond with the "Not Acceptable" response.
411	**Length Required**	When accessing the Submission API resources a "Content-Length" header is required if not using the "Transfer-Encoding" header with the last value "chunked". If not provided the API may respond with this status code.
413	**Content Too Large**	The EDGAR API will reject submissions that are larger than 850M. This value is independent of maximum filing sizes in EDGAR and is separately configurable. A submission successfully transmitted through an EDGAR API resource can still be suspended by EDGAR due to being too large.

Code	Status	Description
415	**Unsupported Media Type**	Each API resource expects to provide a response in a supported media type. In general the preferred media type for requests and responses is "application/json" however some methods do support "application/xml" in both request and response. Notably the submission resources expect the request to be either "application/xml" or "application/octet-stream". The Filer Management resources are "application/json" only. This media type is specified by the client using the "Content-type" header.
429	**Too Many Requests**	In the event that an API resource is being requested too often a status code of 429 may be returned. This is a signal that either the service has become overloaded and cannot handle more requests or that the client is making too many requests in a given timeframe and is being

Code	Status	Description
		asked to make requests less frequently. In the Developer Resources page there is more information on Fair Access and how various resources should be accessed.
500	**Internal Server Error**	When the EDGAR API resource that was requested encounters an error processing the request a status code of 500 will be returned. The help desk can be contacted in cases with persistent errors of this nature.
503	**Service Unavailable**	This method is returned by EDGAR API resources when they are unavailable such as when EDGAR is out of operating hours.

These status codes are provided in reference to communication between the requestor and the API and are indicative of the communication between the requestor and the EDGAR API resources. It is possible for systems between the requestor and the EDGAR API to produce and return their own status codes.

The EDGAR Operational Status API will return a 200 in most cases, regardless of the system status. This is to indicate that the API resource has received, understood, and processed the response. If a 400 or 500 is encountered while accessing this resource then the requesting party would determine that there has been an error. If a 200 response is received with a condition of `NO_COMMUNICATION` then the requesting party knows that, while EDGAR is not able to respond, the API was able to provide a notification of that condition.

Error Responses

The API resources will return an error response when the HTTP status code is in the range of 400-599. An error response from the API methods will consist of an HTTP code, a tracking and locator value, and any relevant messages. The JSON body of an error response would look like this:

```
{
  "tracking": "3a053e7351f176e74fe84585184443a0",
  "locator": "95bb3a",
  "messages": [
    {"type": "ERROR", "content": "Forbidden"}
  ]
}
```

An error response will include at least one but possible more messages. In many cases the message will match the standard HTTP status code reason but in some cases they may provide more context to the error.

User Agent

It is expected that every request to an EDGAR API resource includes a "User-Agent" header that identifies the vendor and version of the software being used to contact EDGAR API resources. The content of this header will make it easier for integrators and SEC EDGAR staff to work together to identify issues accessing EDGAR resources.

Failure to provide this header may result in "NOTICE" level messages from the API or may result in a request returning a response code of 400 for failing to provide a valid user agent.

Server Endpoints

The EDGAR API specification defines the specific server endpoints for communicating with different services. The servers are specified independently to enable flexibility in the future and should be considered as separate resources by implementors.

Server Name	Base URL
EDGAR Operational Status API	https://api.edgarfiling.sec.gov

Server Name	Base URL
EDGAR Submission API	https://api.edgarfiling.sec.gov
EDGAR Filer Management API	https://api.edgarfiling.sec.gov

Each API resource path is relative to the server specified in the endpoint specification for that resource. The full path to any resource is the path of the server and the path of the method combined. For example, to create the path to the Status resource of the EDGAR Status API you would need the base url for the EDGAR Status API and the path to the status resource.

```
https://api.edgarfiling.sec.gov/status
```

This URL would be appropriate for a "get" request to determine the status of the EDGAR system.

EDGAR Operational Status API

This API reports the status of the EDGAR system. This resource requires presentation of a Filer API Token only.

GET `/status` EDGAR System Status

Filer Management API

A set of API endpoints that allows you to enroll in EDGAR Next, verify permissions, get filer information, manage CCC, manage individuals, and manage delegations.

POST `/fm/enrollment` Enrollment

GET `/fm/{cik}` View filer account information

POST `/fm/{cik}/ccc` Generate CCC

PUT `/fm/{cik}/ccc` Create custom CCC

POST `/fm/{cik}/delegationRequests` Request delegation invitations

POST `/fm/{cik}/delegations` Send delegation invitations

GET `/fm/{cik}/delegations` View delegations

POST `/fm/{cik}/individuals` Add individuals

GET `/fm/{cik}/individuals` View individuals

PUT `/fm/{cik}/individuals` Change roles

DELETE `/fm/{cik}/individuals` Remove individuals

GET `/fm/{cik}/verify` Filing credentials verification

Submission API

Submission of filings to EDGAR by API can be made through the Submission API. The Submission API requires the presentation of a Filer API Token in combination with a User API Token. Optionally, filers could also specify a login CIK if there is a delegation from the EDGAR account corresponding to the login CIK to the filer API token EDGAR account/CIK. Delegations can be established on the Filer Management dashboard under “Manage Delegations and User Groups.” For more information on establishing delegations, see How Do I “Understand and Manage Delegation.”

Filings transmitted through the Submission API are expected to conform to the filing standard(s) provided by the SEC. Information is provided on the required format and contents of filings by the Technical Specifications. Information and guidance on filing can be found in the EDGAR Filing Manual. Information on specific forms can be found in the SEC Forms Index.

The Submission API will return accession numbers for the submitted filings. A 202 HTTP status code indicates only that processing will begin in EDGAR and an accession number or accession numbers have been generated. Filings with errors will be processed and suspended by EDGAR. The results of processing (status and errors) are not provided by this endpoint and are instead available to clients using the API through the Submission Status API.

When using Submission API endpoints, the client can optionally specify the "Expect: 100-continue" HTTP request header to allow a response from these API resources to be returned before uploading the entire filing in the event that the EDGAR API determines that the request will not be allowed. This header should only be used for requests that have a body more than 1MiB in size.

Filings may be transmitted via either SINGLE or BULK transmission. A SINGLE transmission provides a single filling and will result in a response containing an accession number if the filing is transmitted successfully or an error if there is an issue while transmitting the filing. Even if a filing is successfully transmitted through the submission API, the filing may still be suspended by EDGAR if the contents of the filing do not satisfy EDGAR filing requirements. The accession number may be provided to the Submission Status API in order to check whether the filing was accepted by EDGAR.

A BULK filing, in contrast, is a single document that contains multiple filings to be submitted. If one or more filings are successfully transmitted then one or more accession numbers will be created and returned in the same order filings are included in the BULK filing document. However, if a filing is not successfully transmitted then an accession number may not be created, which means that the filer may have difficulty determining which accession number corresponds to which filing in the BULK filing document. Even if a filing is successfully transmitted through the Submission API, the filing may still be suspended by EDGAR if the contents of the filing do not satisfy EDGAR filing requirements. The relevant accession number may be provided to the Submission Status API in order to check whether the filing was accepted by EDGAR.

POST `/submission/bulk/live` Bulk Live Submission

POST `/submission/bulk/test` Bulk Test Submission

POST `/submission/single/live` Single Live Submission

POST `/submission/single/test` Single Test Submission

Submission Status API

The Submission Status API provides information from the EDGAR system. It requires a Filer API Token and accession number(s). To obtain information about a submission through the Submission Status API, you must use a Filer API Token that belongs to the CIK that submitted the filing or the CIK that was the filer for the submission. The Submission Status API will only return the detailed information about the filling (like error codes or the filer notification text) once the filing has reached the end of processing signified by the field 'final' being set to true.

The statuses of filings are retained for limited amounts of time. The statuses of Accepted TEST filings are retained for two (2) days, Accepted LIVE filings are retained for sixty (60) days, and

Suspended filings, regardless of type, are retained for six (6) days.

POST `/submission/status` Check Multiple Submission Statuses

GET `/submission/{accessionNumber}/status`
Check Single Submission Status