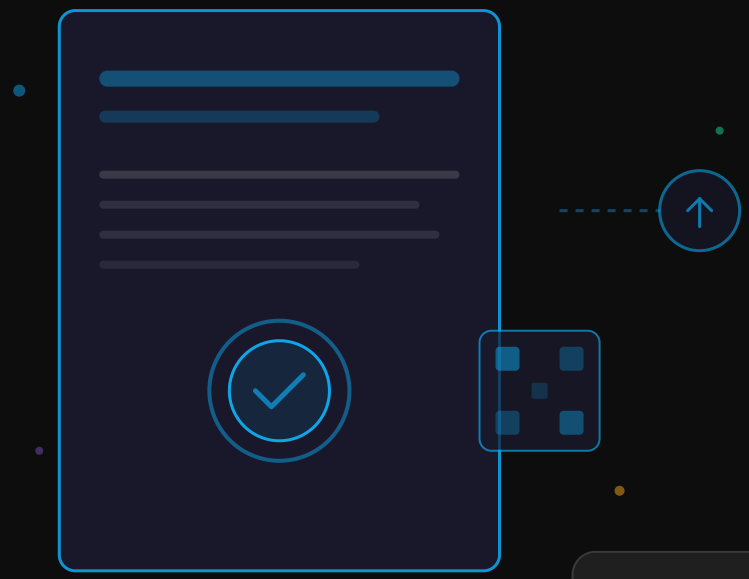
Certificate Vault

Integration Guide

A technical reference for institutions to upload pre-approved certificates directly into VEC.digital for automated verification, tracking, and recipient assignment.



Version 1.0 — June 2026

TABLE OF CONTENTS

INTRODUCTION

VEC.digital supports multiple certificate workflows. This guide covers the **Certificate Vault API** — designed for institutions that already have certificates in PDF or image format and want to upload them directly into VEC.digital for verification, QR code generation, and recipient assignment. For the complete interactive API reference, visit docs.vec.digital/api/operations/tags/certificate-vault.

Vault certificates are **certificate applications** with **source: vault**. Unlike customer-submitted applications (which go through an approval process), vault certificates are **pre-approved by the institution** and go straight to processing — no manual approval needed.

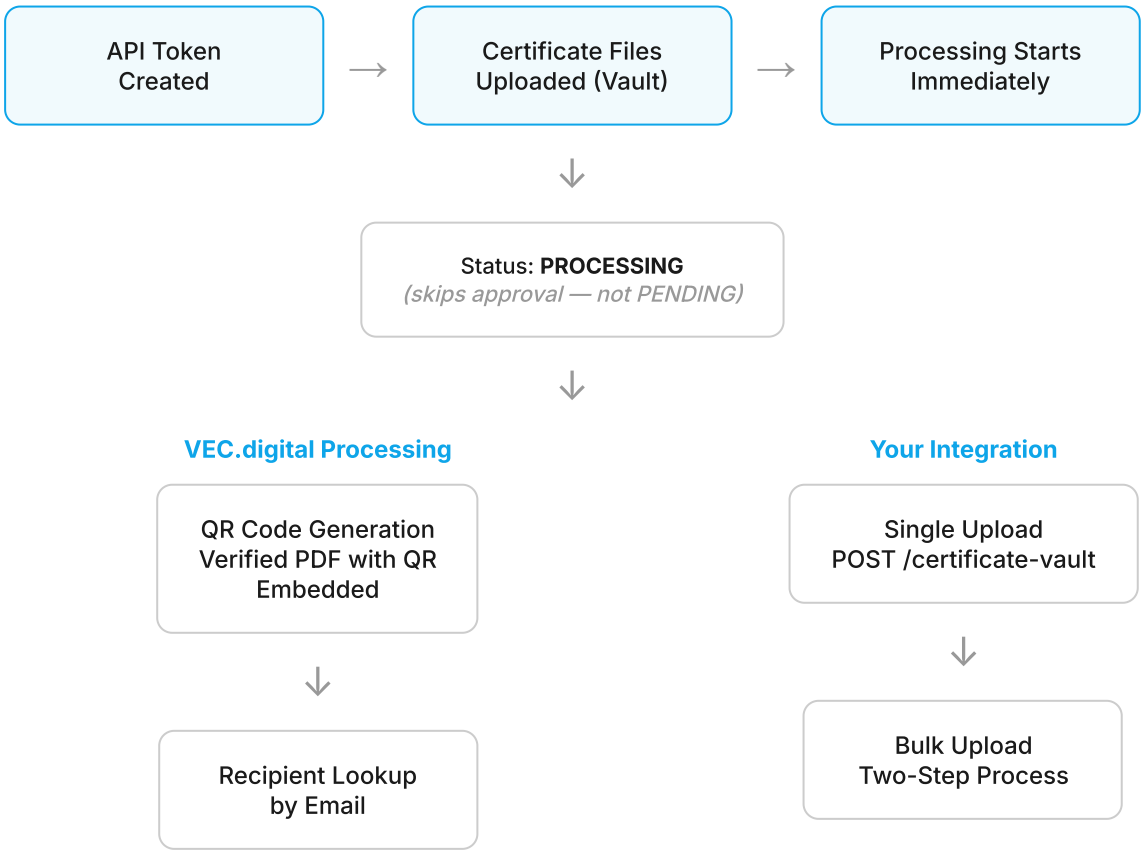
There are two primary integration methods available:

- **API Integration** — Direct HTTP requests to the VEC.digital Business API for single and bulk certificate uploads, downloads, retries, and management
- **Webhook Notifications** — Real-time HTTP callbacks to your server when certificate events occur (approved, failed, retried, etc.)

How the Vault Process Works

Your software communicates with the VEC.digital API using **bearer token authentication**. You upload certificate files (PDF, JPG, PNG) along with recipient metadata. Since the institution is the source, the certificate enters **PROCESSING** status immediately — skipping the approval queue entirely. The platform generates QR codes and verified PDFs, then handles recipient assignment: if the recipient already has a VEC.digital account, the certificate is **automatically assigned**; if not, they receive an **email to view and share their certificate** — no account required. The email also invites them to sign up; once registered with the same email, the certificate is automatically assigned to them.

How the Integration Works



Existing Customer → **Auto-Assigned**
No Account → **View & Share Email** (+ signup invite)

Webhook Events
to Your Server

TERMINOLOGIES

The following terms are used throughout this guide. Familiarize yourself with them before proceeding.

API TOKEN

A unique bearer token generated in your VEC.digital institution dashboard. It authenticates all API requests and carries specific permissions (e.g., **certificate-vault:create**, **certificate-vault:download**). Include it in the **Authorization** header of every request.

CERTIFICATE VAULT

A certificate application with **source: vault**, uploaded directly by an institution via the Business API. Vault certificates skip the customer approval process entirely — they enter **PROCESSING** status immediately upon upload. Institution details are auto-filled, and the recipient is looked up by email: existing customers get the certificate assigned automatically; new recipients receive an email to view and share their certificate without an account — they can also sign up, and once registered with the same email, the certificate is automatically assigned to them.

BEARER AUTHENTICATION

The authentication method used by the VEC.digital Business API. Every API request must include the header **Authorization: Bearer YOUR_API_TOKEN**. This token is tied to your institution account and carries specific permissions. Requests without a valid token receive a **401 Unauthorized** response.

WEBHOOK

An HTTP callback URL that you configure in the VEC.digital dashboard. When a subscribed event occurs (e.g., certificate approved), VEC.digital sends a **POST request** with a JSON payload to your endpoint. Each delivery is signed with **HMAC-SHA256** for verification.

BULK UPLOAD

A two-step process for uploading multiple certificates at once. First, upload individual files and receive a **file_id** for each. Then, submit a bulk request with all entries referencing

their respective file IDs. Processing happens asynchronously in the background.

PREREQUISITES

Before integrating with the VEC.digital Certificate Vault API, ensure the following requirements are met.

API Token & Permissions

Your API token must be generated from an **active institution account** on VEC.digital. Navigate to **Institution Panel** → **API Tokens** → **Create New Token** and assign the following permissions:

> **Note:** *The API token is shown only once after creation. Store it securely. You cannot retrieve it later — you would need to generate a new token.*

PERMISSION	DESCRIPTION
`certificate-vault:list`	List all vault certificates with filtering and pagination
`certificate-vault:view`	View detailed information for a single vault certificate
`certificate-vault:create`	Upload single or bulk vault certificates
`certificate-vault:download`	Download verified PDF files for processed certificates
`certificate-vault:update`	Retry failed vault certificate processing

Webhook Configuration

Webhook endpoints are configured through the **VEC.digital Institution Panel** (not via API). Navigate to **Institution Panel** → **Webhooks** → **Endpoints** → **Create Webhook**. Your institution user account needs the **Manage Webhooks** permission for dashboard access — this is separate from the API token permissions above.

Supported File Formats

FORMAT	EXTENSIONS	MAX SIZE
PDF	`.pdf`	20 MB
Images	`.jpg` `.jpeg` `.png`	20 MB

Software Requirements

Your integration environment needs one of the following:

HTTP Client Library

Any HTTP client capable of making **multipart/form-data** requests with bearer token headers. Examples: cURL, Guzzle (PHP), Axios (JS), Requests (Python), RestTemplate (Java).

HMAC-SHA256 Library

Required for webhook signature verification. Available in all major languages: **hash_hmac()** in PHP, **crypto.createHmac()** in Node.js, **hmac** module in Python.

PROCEDURE

Follow these steps in order to complete the integration. The process covers API token generation, single and bulk certificate uploads, webhook configuration, and signature verification.

Step 1: Obtain an API Token

1. Sign in to your institution account at **vec.digital/login**
2. Navigate to **Institution Panel** → **API Tokens**
3. Click **"Create New Token"**, provide a descriptive name, and select the required permissions listed in the Prerequisites section.
4. **Copy and securely store the token** — it will only be displayed once.

Step 2: Test Your Connection

Verify your token works by making a test request to list certificate templates:

```
curl -X GET \
  'https://api.vec.digital/business/certificate-templates' \
  -H 'Accept: application/json' \
  -H 'Authorization: Bearer YOUR_API_TOKEN'
```

A successful response returns **200 OK** with a JSON array of your certificate templates. If you receive a **401 Unauthorized** error, verify your token is correct.

> **Note:** *All API requests must be made over* **HTTPS**. *The base URL for production is* **api.vec.digital**. *Requests over HTTP will be rejected.*

Step 3: Authentication Headers

Every API request must include these two headers:

Required HTTP Headers

Accept	`application/json`
Authorization	`Bearer YOUR_API_TOKEN`

PROCEDURE

Step 4: Single Certificate Upload

Upload a single vault certificate with its file (PDF, JPG, JPEG, or PNG). The certificate enters **processing** status immediately and is handled asynchronously.

Request — POST /business/certificate-vault

```
curl -X POST \
  'https://api.vec.digital/business/certificate-vault' \
  -H 'Accept: application/json' \
  -H 'Authorization: Bearer YOUR_API_TOKEN' \
  -F 'certificate_title=Certificate of Achievement' \
  -F 'full_name=John Smith' \
  -F 'applicant_email=john@example.com' \
  -F 'issue_date=2026-06-01' \
  -F 'details=Awarded for outstanding performance' \
  -F 'certificate_file=@/path/to/certificate.pdf'
```

Request Fields

FIELD	REQUIRED	DESCRIPTION
`certificate_title`	REQUIRED	Title of the certificate (max 255 characters)
`full_name`	REQUIRED	Full name of the recipient (max 255 characters)
`applicant_email`	REQUIRED	Email address of the recipient (max 255 characters)
`issue_date`	OPTIONAL	Issue date in **YYYY-MM-DD** format (defaults to current date)
`details`	OPTIONAL	Additional details about the certificate (max 1000 characters)
`certificate_file`	REQUIRED	PDF, JPG, JPEG, or PNG file (max 20 MB)

Response — 201 Created

```
{
  "message": "Vault certificate uploaded successfully",
  "data": {
    "id": 42,
    "uuid": "9e7860f0-05d1-4fd0-b3e0-e98e19f1cf15",
    "reference_id": "VEC-IC-202567DAE48425BAD",
    "certificate_title": "Certificate of Achievement",
    "full_name": "John Smith",
    "applicant_email": "john@example.com",
    "source": { "value": "vault", "label": "Vault" },
    "status": {
      "value": "processing",
      "label": "Processing",
      "description": "Certificate is being processed"
    }
  }
}
```

Duplicate Prevention: *The system rejects uploads where the same institution, applicant email, certificate title, and issue date already exist. A* **422** *error is returned with the message: "A vault certificate with this email, title, and issue date already exists"*

PROCEDURE

Step 5: Bulk Upload (Two-Step Process)

For uploading multiple certificates at once, use the two-step bulk process. First upload individual files, collect the **file_id** from each response, then submit all entries in a single bulk request.

Step 5a: Upload Individual Files

Upload each certificate file and save the returned `file_id`:

Request — POST /business/certificate-vault/upload-file

```
curl -X POST \
  'https://api.vec.digital/business/certificate-vault/upload-file' \
  -H 'Accept: application/json' \
  -H 'Authorization: Bearer YOUR_API_TOKEN' \
  -F 'file=@/path/to/certificate1.pdf'
```

Response — 201 Created

```
{
  "message": "File uploaded successfully",
  "data": {
    "file_id": "9e3b101f-c264-4f97-945f-20dfa7163768",
    "file_name": "certificate1.pdf",
    "mime_type": "application/pdf",
    "size": 245760
  }
}
```

> **Note:** *Save each `file_id` — you will need them all in Step 5b. Each file_id is a UUID that references the temporarily stored file.*

Step 5b: Submit Bulk Entries

Submit all entries referencing the collected file IDs. The bulk upload is processed **asynchronously**.

Request — POST /business/certificate-vault/bulk

```
curl -X POST \
  'https://api.vec.digital/business/certificate-vault/bulk' \
  -H 'Accept: application/json' \
  -H 'Authorization: Bearer YOUR_API_TOKEN' \
  -F 'entries[0][certificate_title]=Certificate of Achievement' \
  -F 'entries[0][full_name]=John Smith' \
  -F 'entries[0][applicant_email]=john@example.com' \
  -F 'entries[0][file_id]=9e3b101f-c264-4f97-945f-20dfa7163768' \
  -F 'entries[0][issue_date]=2026-06-01' \
  -F 'entries[1][certificate_title]=Certificate of Excellence' \
  -F 'entries[1][full_name]=Jane Doe' \
  -F 'entries[1][applicant_email]=jane@example.com' \
  -F 'entries[1][file_id]=8d2a0f9e-b1c3-4d86-a570-f12345678901' \
  -F 'entries[1][issue_date]=2026-06-01'
```

Response — 202 Accepted


```
{
  "message": "Bulk vault upload queued for processing",
  "data": {
    "batch_id": "8f7d6e5c-4b3a-2910-z987-y654x321w000",
    "total": 2,
    "status": "processing"
  }
}
```


Use the `batch_id` to track processing progress through the **list endpoint** with status filtering.

PROCEDURE

Step 6: List, View & Download

Listing Vault Certificates

Retrieve all vault certificates for your institution with optional filtering:

List all certificates

```
GET /business/certificate-vault
```

Filter by status

```
GET /business/certificate-vault?status=failed
```

Filter by email

```
GET /business/certificate-vault?applicant_email=john@example.com
```

Search by name, email, title, or reference ID

```
GET /business/certificate-vault?search=John
```

View a Single Vault Certificate

```
GET /business/certificate-vault/{uuid}
```

Replace `{uuid}` with the certificate UUID returned during upload.

Download Verified PDF

```
curl -X GET \
  'https://api.vec.digital/business/certificate-vault/9e7860f0-05d1-4fd0-b3e0-e98e19f1cf15/downlo
ad' \
  -H 'Authorization: Bearer YOUR_API_TOKEN' \
  --output certificate.pdf
```

> **Note:** *The verified PDF is only available after processing completes. If the certificate is still processing, you will receive a* **404** *response: "Verified PDF not found. The certificate may still be processing."*

Step 7: Retry Failed Certificates

If a certificate fails processing, retry it using:

```
POST /business/certificate-vault/{uuid}/retry
```

Only certificates with **failed** status can be retried. Attempting to retry a non-failed certificate returns a **400** error.

PROCEDURE

Step 8: Configure Webhooks

Webhooks allow your application to receive **real-time HTTP notifications** when certificate events occur, eliminating the need to poll the API. Configure webhook endpoints through the **Institution Panel** at **Institution Panel** → **Webhooks** → **Endpoints**. For the full webhook documentation, see docs.vec.digital/guides/webhooks.

Webhook Setup Requirements

- Your endpoint must be publicly accessible over **HTTPS**
- Each institution can create up to **5 webhook endpoints**
- Each webhook must subscribe to at least **1 event**
- Webhook URLs must be valid HTTPS URLs (max 2048 characters)
- Configured through the **Institution Panel** — not via API

Available Webhook Events

EVENT	TRIGGER
`certificate_application.created`	A new certificate record is created (single upload, bulk upload, or customer submission)
`certificate_application.processing`	An institution approves a customer-submitted application and processing begins
`certificate_application.approved`	A certificate has been successfully processed and approved
`certificate_application.rejected`	An institution administrator rejects a customer-submitted application
`certificate_application.failed`	Background processing encountered a critical error (PDF generation, QR code, etc.)
`certificate_application.retried`	An administrator re-initiates processing for a previously failed application

For detailed payload schemas for each event, see docs.vec.digital/guides/webhook-events.

Webhook Payload

Every webhook delivery contains a standardized JSON payload:

```
{
  "event": "certificate_application.approved",
  "webhook_id": "a1b2c3d4-e5f6-7890-abcd-ef1234567890",
  "timestamp": "2026-06-04T10:30:00Z",
  "application_id": 42,
  "uuid": "9e7860f0-05d1-4fd0-b3e0-e98e19f1cf15",
  "certificate_title": "Certificate of Achievement",
  "applicant_email": "john@example.com",
  "source": "vault",
```

```
  "trigger_context": "bulk_upload"
}
```

PROCEDURE

Step 9: Verify Webhook Signatures

Every webhook delivery includes an **HMAC-SHA256 signature** in the `X-Webhook-Signature` header. Always verify this signature before processing the payload to ensure the request originated from VEC.digital.

Delivery Headers

HEADER	DESCRIPTION
`X-Webhook-Signature`	HMAC-SHA256 signature of the raw request body using your signing secret
`X-Webhook-Event`	The event name (e.g., `certificate_application.created`)
`X-Webhook-ID`	The webhook endpoint UUID
`X-Webhook-Timestamp`	ISO 8601 timestamp of when the delivery was sent
`Content-Type`	`application/json`

Verification Logic

```
expected_signature = HMAC-SHA256(raw_request_body, your_signing_secret)
valid = timing_safe_compare(X-Webhook-Signature header, expected_signature)
```

PHP Example

```php
function verifySignature(string $rawBody, string $signatureHeader, string $secret): bool {
    $expected = hash_hmac('sha256', $rawBody, $secret);
    return hash_equals($expected, $signatureHeader);
}

// In your handler:
$rawBody    = file_get_contents('php://input');
$signature  = $_SERVER['HTTP_X_WEBHOOK_SIGNATURE'];
$event      = $_SERVER['HTTP_X_WEBHOOK_EVENT'];

if (!verifySignature($rawBody, $signature, $webhookSecret)) {
    http_response_code(401);
    echo 'Invalid signature';
    exit;
}

$payload = json_decode($rawBody, true);
// Process the event...
```

Node.js Example

```javascript
const crypto = require('crypto');

function verifySignature(rawBody, signatureHeader, secret) {
  const expected = crypto
    .createHmac('sha256', secret)
```

```
    .update(rawBody)
    .digest('hex');

  return crypto.timingSafeEqual(
    Buffer.from(signatureHeader),
    Buffer.from(expected)
  );
}
```

> **Important:** *Always use a* **timing-safe comparison** *function to prevent timing attacks. Never use regular string equality (== or ===).*

PROCEDURE

Step 10: Delivery Behavior & Best Practices

Delivery & Retry Behavior

- Webhooks are delivered **asynchronously** — your endpoint should respond quickly with a **2xx** status code
- Failed deliveries (non-2xx responses or network errors) are retried up to **3 times** with **30-second** backoff intervals
- The `last_delivery_status` in the dashboard shows **success**, **failed**, or **none**
- The `consecutive_failures` counter tracks repeated failures and resets to 0 on success
- A disabled webhook (`is_active: false`) will not receive deliveries

Best Practices

- **Always verify signatures** before processing webhook payloads — never trust unverified requests
- **Respond quickly** — return a 2xx status immediately, then process the event asynchronously in your application
- **Handle duplicates** — webhooks may occasionally be delivered more than once; make your handler **idempotent**
- **Use HTTPS** — your endpoint must be publicly accessible over HTTPS with a valid SSL certificate
- **Log deliveries** — keep a record of received payloads and their signatures for debugging
- **Monitor consecutive failures** — check the webhook dashboard periodically for endpoints with high failure counts

Certificate Status Lifecycle

Vault certificates follow a specific lifecycle. Unlike customer-submitted applications (which start at `pending`), vault certificates enter `processing` immediately:

STATUS	VAULT BEHAVIOR
processing	Vault certificates start here immediately after upload — no approval step. QR code generation, verified PDF creation, and recipient notification are in progress.
approved	Processing completed successfully. Verified PDF with embedded QR code is ready for download. Recipient has been notified.
failed	Processing encountered an error (e.g., PDF generation failure). Can be retried via the **retry endpoint**.
revoked	Certificate has been revoked by the institution.

> **Note:** *Vault certificates never enter the `pending` or `in_review` statuses because they are institution-sourced and pre-approved. The `pending` and `in_review` statuses are only used for customer-submitted applications that require institutional approval.*

Notification Behavior

When a vault certificate is processed:

- **Recipient has a VEC.digital account:** The certificate is automatically assigned to their account and they receive an assignment notification email
- **Recipient does not have an account:** They receive an email to view and share their certificate — no account required. The email also invites them to create a free VEC.digital account; once registered with the same email, the certificate is automatically assigned to them

EXCEPTIONS

This section covers common errors you may encounter during integration and their resolution steps.

401 Unauthorized — *"Unauthorized - Invalid API token"*

This error occurs when the API token is missing, expired, or invalid. Verify the following:

- The **Authorization** header is present and formatted as `Bearer YOUR_API_TOKEN`
- The token was copied correctly — no leading/trailing whitespace or line breaks
- The token has not been revoked in the VEC.digital dashboard
- If unsure, generate a new API token and replace the old one

403 Forbidden — *"Forbidden - Insufficient permissions"*

The token is valid but lacks the required permission for this endpoint.

- Check the `required_permission` field in the error response
- Navigate to **Institution Panel** → **API Tokens** and edit the token's permissions
- Ensure the permission matching the endpoint is enabled (e.g., `certificate-vault:create`)

< OR >

422 Unprocessable Entity — *"A vault certificate with this email, title, and issue date already exists"*

Duplicate prevention has blocked this upload. The system enforces uniqueness on the combination of institution, applicant email, certificate title, and issue date.

- Check if the certificate was already uploaded — use the **list endpoint** to search by email or title
- If the existing certificate is in **failed** status, use the **retry endpoint** instead of re-uploading
- If the issue date differs, include it in the request to create a distinct record

404 Not Found — *"Verified PDF not found. The certificate may still be processing."*

The download was requested before processing completed.

- Check the certificate status via the **view endpoint** — wait until status is `approved`
- Subscribe to the `certificate_application.approved` webhook to get notified when the PDF is ready
- If the status is `failed`, retry the certificate first

Webhook Delivery Failed — *"Connection Refused" / "Timeout"*

The webhook delivery could not reach your endpoint server.

- Verify your endpoint URL is correct and publicly accessible over **HTTPS**

- Ensure your server responds within **10 seconds** — process events asynchronously
- Check your firewall and server logs for incoming requests from VEC.digital IPs
- Use the **"Test Webhook"** button in the dashboard to validate your endpoint
- Review the **Deliveries** page for detailed response codes and error messages

QUICK REFERENCE

API Endpoints Summary

METHOD	ENDPOINT	DESCRIPTION
`POST`	`/business/certificate-vault`	Upload single vault certificate
`POST`	`/business/certificate-vault/upload-file`	Upload file for bulk processing
`POST`	`/business/certificate-vault/bulk`	Submit bulk entries
`GET`	`/business/certificate-vault`	List vault certificates
`GET`	`/business/certificate-vault/{uuid}`	View a vault certificate
`GET`	`/business/certificate-vault/{uuid}/download`	Download verified PDF
`POST`	`/business/certificate-vault/{uuid}/retry`	Retry a failed certificate
`GET`	`/business/certificate-templates`	List certificate templates
`GET`	`/business/batches`	List certificate batches
`GET`	`/business/certificates`	List all certificates

HTTP Status Codes

CODE	MEANING
200	Successful read or retry operation
201	Resource created successfully (single upload, file upload)
202	Bulk upload accepted and queued for processing
400	Bad request — invalid operation (e.g., retry non-failed certificate)
401	Authentication failed — invalid or missing API token
403	Permission denied — token lacks required permission
404	Resource not found or PDF not yet available
422	Validation error — duplicate, missing fields, or invalid file_id
500	Server error — contact support if persistent

Additional Resources

RESOURCE	URL
Certificate Vault Guide	docs.vec.digital/guides/certificate-vault
Vault API Reference	docs.vec.digital/api/operations/tags/certificate-vault

RESOURCE	URL
Webhooks Guide	docs.vec.digital/guides/webhooks
Webhook Events Reference	docs.vec.digital/guides/webhook-events

Support

For integration support, contact the VEC.digital team at **support@vec.digital** or visit vec.digital/support. Include your institution name, API token name (not the token itself), and the relevant request/response details when reporting issues.