Cal.diy booking attendees and artifacts

Use these capabilities to manage the people attached to a booking or to read its generated calendar links, provider references, recording and transcript metadata, and conferencing sessions. This page owns the remaining 10 operations in the

SECTION	Connectors
TYPE	Connector
PROTOCOL	AIP 1.0
SOURCE	docs/connectors/cal-diy/capabilities/booking-attendees-and-artifacts.md

All operations use provider API version `2024-08-13`. Every input and result in this group is restricted, contains PII, and requires redaction.

Choose an operation

Attendees and guests

Capability	Provider request	Risk	Approval
`cap:cal_diy:booking.attendee.list`	`GET /v2/bookings/{booking_uid}/attendees`	Low	No
`cap:cal_diy:booking.attendee.get`	`GET /v2/bookings/{booking_uid}/attendees/{attendee_id}`	Low	No
`cap:cal_diy:booking.attendee.add`	`POST /v2/bookings/{booking_uid}/attendees`	Medium	Required
`cap:cal_diy:booking.attendee.delete`	`DELETE /v2/bookings/{booking_uid}/attendees/{attendee_id}`	High	Required
`cap:cal_diy:booking.guest.add`	`POST /v2/bookings/{booking_uid}/guests`	Medium	Required

Links, references, and meeting artifacts

Capability	Provider request	Risk	Approval
`cap:cal_diy:booking.calendar_links.get`	`GET /v2/bookings/{booking_uid}/calendar-links`	Low	No
`cap:cal_diy:booking.reference.list`	`GET /v2/bookings/{booking_uid}/references`	Low	No
`cap:cal_diy:booking.recording.list`	`GET /v2/bookings/{booking_uid}/recordings`	High	Required
`cap:cal_diy:booking.transcript.list`	`GET /v2/bookings/{booking_uid}/transcripts`	High	Required
`cap:cal_diy:booking.conferencing_session.list`	`GET /v2/bookings/{booking_uid}/conferencing-sessions`	Low	No

Recording and transcript operations are reads, but their restricted-media boundary makes them high risk and approval-required.

List or read attendees

List accepts only a non-empty booking uid:

```
JSON
{
  "booking_uid": "booking-provider-uid"
}
```

Read one attendee by adding its integer provider id:

```
JSON
{
  "booking_uid": "booking-provider-uid",
  "attendee_id": 73
}
```

Additional fields are rejected. The connector percent-encodes `booking_uid` into the provider path. `attendee_id` is an integer without an additional minimum in the published schema; Cal.diy can still reject an id outside the provider's domain.

These are low-risk, retry-safe reads. No capability-level human approval is required, but normal identity, tenant, account, and restricted-data policy still applies.

Add an attendee

Attendee add requires the booking uid, name, time zone, and email:

```
JSON
{
  "booking_uid": "booking-provider-uid",
  "name": "Avery Patel",
  "timeZone": "Europe/London",
  "email": "avery@example.com",
  "language": "en"
}
```

Field	Constraint
`booking_uid`	Non-empty string; required
`name`	Non-empty string; required
`timeZone`	Non-empty string; required
`email`	Email string; required
`phoneNumber`	`+` plus 7 through 15 digits
`language`	The 43-value Cal.diy locale enum

The locale enum is listed in the profile capability reference. Additional input fields are rejected.

Before the add request, the connector reads the booking. A passing preflight shows that the target was visible at that moment; it does not guarantee that the provider will accept the attendee.

The operation is medium risk and carries `write`, `external_network`, and `send_message` side effects. The connector description states that it updates connected calendar events. The marker does not prove that every participant received a notification or calendar update.

Remove an attendee

Attendee delete uses the same two-identifier input as attendee get:

```
JSON
{
  "booking_uid": "booking-provider-uid",
  "attendee_id": 73
}
```

The connector reads that exact attendee before deletion. The operation is high risk and destructive, and it can update connected calendar events or participant-visible state.

There is no rollback or compensating capability. Adding a person later creates a new provider change and does not restore prior invitation, attendance, or notification history.

Add guests

Guest add requires the `guests` field to contain 1 through 10 guest objects:

```
JSON
{
  "booking_uid": "booking-provider-uid",
  "guests": [
    {
      "email": "jordan@example.com",
      "name": "Jordan Lee",
      "timeZone": "America/New_York",
      "language": "en"
    }
  ]
}
```

Each guest requires only `email`. Optional fields are `name`, `timeZone`, `phoneNumber`, and `language`. The guest schema does not require non-empty `name` or `timeZone`; provider policy may impose stricter rules. Phone and language use the same constraints as attendee add. Additional guest fields are rejected.

The array does not declare `uniqueItems`, so duplicate guest objects can pass schema validation. Deduplicate intended recipients before approval rather than depending on provider behavior.

Guest add first reads the booking. It is medium risk and carries the `send_message` marker because the provider may send applicable notifications. It declares no compensation.

Read calendar links and references

Calendar-link get accepts only:

```
JSON
{
  "booking_uid": "booking-provider-uid"
}
```

It reads generated booking calendar links. The connector does not publish field names or expiry semantics for those provider-owned links.

Reference list requires the same booking uid and may add one `type` filter:

```
JSON
{
  "booking_uid": "booking-provider-uid",
  "type": "google_calendar"
}
```

The exact filter enum is:

```
TEXT
google_calendar | office365_calendar | daily_video | google_video
office365_video | zoom_video
```

The filter is optional. These references identify provider calendar or video systems associated with the booking; they are not proof that an external event or meeting remains reachable.

Both operations are low-risk, retry-safe reads. Treat returned links and external identifiers as restricted even when a URL itself is shareable.

Read recordings and transcripts

Recording and transcript list each accept only the booking uid:

```
JSON
{
  "booking_uid": "booking-provider-uid"
}
```

These capabilities list restricted download metadata. They do not declare a media upload, streaming, or byte-download contract, and the connector does not publish the provider metadata field shape.

Both operations are high-risk reads and require approval because a response may expose restricted meeting media or transcript data. Approval uses the tenant-policy selector, a 900,000 ms validity window, and reason, input snapshot, and policy-decision evidence.

Approval must bind the actor, tenant, booking, capability, and exact input. A retry-safe marker does not bypass approval expiry, changed authorization, or a new booking uid.

Despite their high risk and approval requirement, both operations retain the read contract:

- optional action-scoped idempotency;
- connector retry support;
- retry safety `safe`;
- synchronous execution only.

Retry only within the current approval, deadline, and data-access policy. Do not automatically follow or download a URL found in provider metadata.

Read conferencing sessions

Conferencing-session list accepts only the booking uid and returns provider-owned session metadata:

```
JSON
{
  "booking_uid": "booking-provider-uid"
}
```

It is a low-risk, approval-free capability at the contract level. The result remains restricted and PII-bearing. Session presence does not prove that a meeting is active, joinable, recorded, or qualified for a particular provider.

Shared read contract

All seven reads are synchronous, retry-safe, and eligible for connector retry. Their idempotency key is optional, action-scoped, and collision-checked against the input hash.

Five reads are low risk and do not require capability-level approval. Recording and transcript list are the two high-risk exceptions described above. Risk and approval are independent contract fields; neither changes these operations into mutations.

Shared mutation contract

Attendee add, attendee delete, and guest add require:

Contract field	Value
Human approval	Required
Approval selector	Tenant policy
Approval validity window	900,000 ms

Contract field	Value
Approval evidence	Reason, input snapshot, policy decision
Idempotency	Required; external-account scope
Collision behavior	Revalidate input hash
Connector retry	Not supported
Retry safety	Unsafe
Execution	Synchronous only
Transaction modes	`dry_run`, `plan`, `commit`, `reconcile`
Plan before commit	Not required
Dry-run fidelity	Policy and schema only

All three mutations carry `send_message`. None declares a compensating capability. Attendee delete is destructive and uses the stricter destructive approval reason; add operations use the customer-visible-change reason.

An uncertain provider dispatch must be reconciled from the original durable operation. Do not issue the same people-changing action under a replacement key.

Result and redaction contract

The connector returns the validated provider JSON body as AIP output. A non-empty string `status` is required. `data`, `pagination`, and additional provider fields are optional; their shapes are not guaranteed by the connector schema.

The connector recursively replaces secret-, key-, and token-named values with `[REDACTED]`. This targeted redaction does not remove booking names, email addresses, phone numbers, URLs, transcripts, recording metadata, or other PII unless their field names match the secret redactor. Keep the full output under restricted-data controls.

Failures and recovery

Failure	Relevant cause	Safe response
`connector.cal_diy.invalid_action`	Missing id, invalid attendee or guest, unsupported reference type, or unknown field	Correct the exact input before another call
`connector.cal_diy.policy`	Mutation omitted its required idempotency key	Preserve the approved input and add the missing key
`connector.cal_diy.idempotency_collision`	One mutation key owns different input	Stop and inspect the original action
`connector.cal_diy.in_flight`	A matching people mutation is executing	Read durable state and wait
`connector.cal_diy.outcome_unknown`	Prior provider dispatch cannot be proved	Reconcile; never create a speculative replacement key
`connector.cal_diy.authentication`	Provider returned `401` or `403`	Verify account binding, credential revision, booking ownership, and media authorization
`connector.cal_diy.state_conflict`	Provider returned `409` or `412`	Refresh booking or attendee state and review the requested change
`connector.cal_diy.rate_limited`	Provider returned `429`	Retry reads within current approval and policy; reconcile mutations
`connector.cal_diy.remote_temporary`	Provider returned a server error	Retry reads within policy; reconcile mutations
`connector.cal_diy.transport`	Provider exchange failed	Retry reads only; treat dispatched mutations as uncertain
`connector.cal_diy.invalid_provider_output`	Successful response lacked a valid `status`	Preserve evidence and reconcile a mutation

Permanent provider rejection, invalid credentials, oversized responses, and durable settlement failures use the common Cal.diy error family. Gateway and lifecycle errors are described in the global error reference.

Verify attendee or artifact handling

For attendee changes, verify the approval and idempotency key, wait for a completed durable result, then list attendees or read the exact attendee. For guest add, read the booking and compare only the intended guest change.

For artifact reads, verify the exact capability, booking uid, actor, tenant, external account, approval when required, response status, and data-handling decision. Do not expose metadata merely because the operation completed.

Retain the provider request id when present. Reconcile uncertain mutations and never use a follow-up read as authorization to repeat a people-changing action.

These checks validate application handling. They do not prove notification delivery, media accessibility, live-provider qualification, or production readiness.

Related documentation

- Cal.diy capability index
- Core booking capabilities
- Authentication and tenant routing
- Transactions and compensation
- Error reference