Package: delphyr (via r-universe)

November 3, 2026

Title Reproducible Round-Based Delphi Studies

Version 0.0.2

Description Validated study protocols, deterministic consensus and longitudinal analysis, and transactional PostgreSQL services for round-based Delphi studies. Provides reproducible research exports and synthetic examples. Production use requires the documented security and operational acceptance gates.

License MIT + file LICENSE

Encoding UTF-8

Imports DBI, digest, jsonlite (>= 1.8.8), openssl, uuid

Suggests RPostgres, callr, withr, pkgload, testthat (>= 3.0.0), knitr, rmarkdown

Config/testthat/edition 3

VignetteBuilder knitr

URL https://github.com/CTTIR/delphyR

BugReports https://github.com/CTTIR/delphyR/issues

Config/Needs/website CTTIR/themakR, pkgdown

Config/roxygen2/version 8.1.1

Config/pak/sysreqs libssl-dev

Repository https://cttir.r-universe.dev

Date/Publication 2026-11-03 12:27:18 UTC

RemoteUrl https://github.com/CTTIR/delphyR

RemoteRef HEAD

RemoteSha cd2f51e7b12838597b79ff64b6d763fad8a94875

RemoteSubdir packages/delphyr

Contents

accept_panel_invitation . . . 4
add_panelist . . . 5
amend_protocol . . . 6
analyse_round . . . 7
assign_feedback . . . 7
authenticated_actor . . . 8
cancel_campaign . . . 9
change_round_deadline . . . 9
claim_job . . . 10
classify_consensus . . . 11
code_qualitative_source . . . 11
compare_rounds . . . 12
complete_study . . . 13
connect_repository . . . 13
content_hash . . . 14
create_feedback . . . 14
create_qualitative_theme . . . 15
create_study . . . 16
del_abort . . . 16
demo_actor . . . 17
demo_protocol . . . 17
demo_snapshot . . . 18
demo_study . . . 18
download_artifact . . . 19
enroll_panel . . . 19
freeze_round . . . 20
get_account_rights . . . 21
get_analysis . . . 21
get_capabilities . . . 22
get_feedback . . . 22
get_feedback_candidate . . . 23
get_item_comparability . . . 23
get_operation . . . 24
get_operations_status . . . 24
get_panel_import_receipt . . . 25
get_qualitative_provenance . . . 25
get_questionnaire . . . 26
get_retention_report . . . 26
get_round_instrument . . . 27
get_round_readiness . . . 27
get_snapshot . . . 28
get_study_documentation . . . 28
get_study_setup . . . 29
get_system_status . . . 29
hold_pending_messages . . . 30
import_panel . . . 31

import_round_contributions . . . 31
invitation_code . . . 32
issue_panel_invitation . . . 32
link_item_source . . . 33
list_audit_events . . . 34
list_campaign_enrollments . . . 35
list_campaign_rounds . . . 36
list_contribution_rounds . . . 36
list_enrollments . . . 37
list_item_decisions . . . 37
list_panel . . . 38
list_panel_invitations . . . 38
list_protocol_versions . . . 39
list_qualitative_reviews . . . 39
list_released_edits . . . 40
list_released_feedback . . . 40
list_rounds . . . 41
list_studies . . . 41
list_study_staff . . . 42
list_uncertain_deliveries . . . 42
log_event . . . 43
log_operation . . . 44
migrate_repository . . . 44
new_authentication_config . . . 45
new_correlation_id . . . 46
new_invitation_token . . . 46
new_message_adapter . . . 47
new_protocol . . . 47
new_scale . . . 48
new_snapshot . . . 48
parse_invitation_code . . . 49
prepare_campaign . . . 50
prepare_feedback . . . 51
prepare_report_data . . . 51
prepare_round . . . 52
prepare_study_report_data . . . 53
preview_campaign . . . 53
preview_panel_import . . . 54
preview_panel_invitation . . . 54
process_campaign_message . . . 55
process_campaign_sink . . . 56
provision_demo_principal . . . 56
publish_consent . . . 57
read_item_import . . . 57
record_consent . . . 58
record_item_comparability . . . 59
record_item_decision . . . 60
record_item_lineage . . . 60

record_qualitative_source . . . 61
record_study_documentation . . . 62
record_worker_heartbeat . . . 63
redact_qualitative_source . . . 64
register_invited_account . . . 65
register_staff_account . . . 66
release_campaign . . . 67
release_feedback . . . 68
release_feedback_correction . . . 68
release_qualitative_edit . . . 69
remove_expired_artifacts . . . 70
remove_unfinished_exports . . . 71
render_study_report . . . 71
renew_job_lease . . . 72
reproduce_export . . . 73
reproduce_study_export . . . 73
request_analysis . . . 74
request_export . . . 74
request_study_export . . . 75
resolve_delivery . . . 76
revoke_panel_invitation . . . 77
run_analysis . . . 77
save_response . . . 78
set_capability . . . 79
set_panel_group . . . 80
submit_round . . . 80
transition_round . . . 81
validate_feedback . . . 82
validate_items . . . 82
validate_protocol . . . 83
validate_response . . . 83
withdraw_participation . . . 84
worker_step . . . 84
write_participant_feedback . . . 85
write_report_data . . . 86

Index **87**

`accept_panel_invitation` *Accept an invitation after verified login and an explicit confirmed action*

Description

Accept an invitation after verified login and an explicit confirmed action

Usage

```
accept_panel_invitation(
  repo,
  actor,
  study_id,
  invitation_id,
  token,
  confirmed,
  command_id
)
```

Arguments

repo	Repository.
actor	Study coordinator.
study_id	Study UUID.
invitation_id	Invitation UUID, not a panelist identifier.
token	Private output of new_invitation_token(). Never persisted in plaintext.
confirmed	Explicit TRUE from the confirmation action; never call on GET.
command_id	Idempotency key.

Value

Invitation, membership and pseudonym receipt. No consent or round enrollment is created. Existing provisioned accounts are required.

---

add_panelist *Register a synthetic panel member*

Description

Register a synthetic panel member

Usage

```
add_panelist(repo, actor, study_id, principal_id, group_code, command_id)
```

Arguments

repo	Repository.
actor	Study manager.
study_id	Study UUID.
principal_id	Existing synthetic account UUID.
group_code	Declared stakeholder group.
command_id	Idempotency key.

Value

Study-specific pseudonym. A completed study refuses new members with DEL_CONFLICT (path study.state).

`amend_protocol` *Approve a new protocol version for future rounds*

Description

Existing rounds and snapshots retain their exact protocol. An amendment never changes the meaning of an existing scale code. Introduce a new scale code for changed scales and a new item version before using it. This service records a synthetic technical approval, not institutional or ethical approval.

Usage

```
amend_protocol(
  repo,
  actor,
  study_id,
  protocol,
  expected_hash,
  reason,
  command_id
)
```

Arguments

`repo`	Repository.
`actor`	Study manager.
`study_id`	Study UUID.
`protocol`	Validated complete replacement protocol.
`expected_hash`	Latest protocol hash shown during review.
`reason`	Nonempty amendment rationale.
`command_id`	Idempotency key.

Value

New protocol id, version and hash.

analyse_round *Analyse a frozen round independently of the application*

Description

Analyse a frozen round independently of the application

Usage

```
analyse_round(snapshot, rules = snapshot$protocol$analysis$consensus)
```

Arguments

snapshot	Validated delphyr_snapshot.
rules	Consensus rule; defaults to the frozen protocol.

Value

delphyr_analysis containing results, denominators, missingness, distributions, settings, warnings and scientific provenance.

Examples

```
summary(analyse_round(demo_snapshot()))
```

assign_feedback *Assign released feedback to an unopened subsequent round*

Description

Assign released feedback to an unopened subsequent round

Usage

```
assign_feedback(repo, actor, round_id, feedback_id, command_id)
```

Arguments

repo	Repository.
actor	Study manager.
round_id	Target round.
feedback_id	Released feedback in the same study.
command_id	Idempotency key.

Value

Target round UUID. A round shows one feedback: a second assignment, of the same or of another feedback, is refused with DEL_CONFLICT (path feedback.assigned). A retry under the same key returns the receipt.

`authenticated_actor` *Resolve a verified gateway subject to a current database principal*

Description

This adapter does not implement OIDC or verify browser credentials. It accepts only the transport and headers established by the separately configured, isolated authentication gateway. A header alone is not authentication.

Usage

```
authenticated_actor(repo, request, config)
```

Arguments

`repo`	Repository.
`request`	Shiny session$request received on the server, never browser input.
`config`	Server-only output of new_authentication_config().

Value

Short-lived delphyr_actor. Unknown, disabled, spoofed or missing identities fail closed with DEL_UNAUTHORIZED. No account is auto-provisioned.

Examples

```
if (FALSE) {
  config <- new_authentication_config(
    "https://identity.example/realms/study",
    Sys.getenv("DELPHYR_GATEWAY_SECRET"), "127.0.0.1"
  )
  # In a trusted session factory: authenticated_actor(repo, session$request, config)
}
```

cancel_campaign *Cancel remaining synthetic campaign delivery*

Description

Cancel remaining synthetic campaign delivery

Usage

```
cancel_campaign(repo, actor, campaign_id, reason, command_id)
```

Arguments

repo	Repository.
actor	Actor with coordinate capability.
campaign_id	Campaign UUID.
reason	Human cancellation rationale.
command_id	Idempotency key.

Value

Campaign UUID; already recorded sink receipts remain immutable. A campaign that is already cancelled is refused as ‘campaign.state‘.

change_round_deadline *Change the deadline of a round that is not closed*

Description

The deadline is not part of the reviewed instrument, so the approval of a round stays valid. Before a round opens the deadline may be any future time. Once the round is open it can only be moved to a later time: nobody loses time they were told they had. An open round whose deadline has passed accepts answers again after an extension. A closed round is never reopened this way. The change is recorded with its rationale in the round’s history and in the audit trail.

Usage

```
change_round_deadline(
  repo,
  actor,
  round_id,
  deadline,
  expected_hash,
  reason,
  command_id
)
```

Arguments

repo	Repository.
actor	Study manager.
round_id	Round UUID.
deadline	New deadline, a timestamp with explicit offset.
expected_hash	Instrument hash of the round, as shown in its review.
reason	Nonempty rationale.
command_id	Idempotency key.

Value

Round id and the new deadline in UTC.

---

claim_job *Claim one available durable job with an exclusive lease*

Description

Claim one available durable job with an exclusive lease

Usage

```
claim_job(repo, lease_seconds = 120, study_id = NULL)
```

Arguments

repo	Trusted worker repository.
lease_seconds	Lease duration, between 1 and 3600 seconds.
study_id	Optional study UUID limiting a worker partition.

Value

Claimed row or empty data frame. Worker infrastructure API.

classify_consensus *Classify consensus from explicit counts*

Description

Classify consensus from explicit counts

Usage

```
classify_consensus(counts, rule)
```

Arguments

counts	Named list n_valid, n_agree, n_disagree.
rule	Validated consensus rule.

Value

Classification and unrounded proportions with numerator and denominator.

Examples

```
classify_consensus(
  list(n_valid = 10, n_agree = 7, n_disagree = 1),
  demo_protocol()$analysis$consensus
)
```

code_qualitative_source *Append a reasoned theme coding decision*

Description

Append a reasoned theme coding decision

Usage

```
code_qualitative_source(
  repo,
  actor,
  study_id,
  source_id,
  theme_id,
  decision,
  reason,
  command_id
)
```

Arguments

repo	Repository.
actor	Actor with edit capability.
study_id	Study UUID.
source_id	Original source UUID.
theme_id	Exact theme-version UUID.
decision	include or exclude. Later decisions never overwrite history.
reason	Coding rationale or explanation of disagreement.
command_id	Idempotency key.

Value

Coding-event UUID. Counts represent decisions, not unique people.

---

compare_rounds *Compare paired responses across compatible rounds*

Description

Compare paired responses across compatible rounds

Usage

```
compare_rounds(previous, current, mapping = NULL)
```

Arguments

previous, current	Validated snapshots with the same study code.
mapping	Optional table item_code, dimension_code, previous_version, current_version, comparable, reason. Explicit approval required for new versions.

Value

delphyr_comparison table, reporting attrition and unpaired means separately.

Examples

```
compare_rounds(demo_snapshot(c(5, 7, 8, 9)), demo_snapshot(c(6, 7, 7, 9)))
```

complete_study *Complete a study after finalization and explicit item decisions*

Description

Complete a study after finalization and explicit item decisions

Usage

```
complete_study(repo, actor, study_id, reason, command_id)
```

Arguments

repo	Repository.
actor	Study manager.
study_id	Study UUID.
reason	Human completion rationale, including persistent dissent.
command_id	Idempotency key.

Value

Completed study UUID.

connect_repository *Connect to an explicitly configured PostgreSQL database*

Description

Connect to an explicitly configured PostgreSQL database

Usage

```
connect_repository(
  ...,
  environment = c("development", "test"),
  artifact_root = tempfile("delphyr-artifacts-")
)
```

Arguments

...	Arguments passed to DBI::dbConnect(RPostgres::Postgres()).
environment	Development or test; production is not qualified.
artifact_root	Private directory for generated artifacts.

Value

A delphyr repository. Close its connection with DBI::dbDisconnect(repo$con).

content_hash *Canonical scientific content hash*

Description

Canonical scientific content hash

Usage

```
content_hash(x, version = "2")
```

Arguments

x	Declarative data; data frame row order is ignored.
version	Hash contract: "2" (default) preserves double precision; "1" reproduces the legacy 15-digit JSON contract for existing records only.

Value

SHA-256 string of sorted UTF-8 JSON, excluding no fields implicitly. Version 2 starts with 'v2:'; legacy version 1 is an unprefixed digest. A legacy hash cannot detect changes below its original numeric precision.

Examples

```
content_hash(list(b = 2, a = 1))
```

create_feedback *Create a reviewed feedback candidate from a frozen analysis*

Description

Qualitative content enters participant feedback only as an editorial version that another person has released. The service reads the released text itself; text supplied by the caller is not accepted. A version whose source is linked to item codes is shown beside those items.

Usage

```
create_feedback(
  repo,
  actor,
  analysis_id,
  qualitative = list(),
  command_id,
  released_edits = character()
)
```

Arguments

repo	Repository.
actor	Actor with analyse capability.
analysis_id	Analysis UUID.
qualitative	Must be empty; use released_edits.
command_id	Idempotency key.
released_edits	UUIDs of released editorial versions of this study.

Value

Feedback UUID and hash, requiring separate release.

---

create_qualitative_theme

Define an immutable version of a qualitative theme

Description

Define an immutable version of a qualitative theme

Usage

```
create_qualitative_theme(
  repo,
  actor,
  study_id,
  code,
  version,
  label,
  definition,
  command_id
)
```

Arguments

repo	Repository.
actor	Actor with edit capability.
study_id	Study UUID.
code	Stable theme code.
version	Positive integer version.
label	Short theme label.
definition	Coding definition or decision rule.
command_id	Idempotency key.

Value

Theme UUID.

`create_study` *Create an isolated synthetic study and its first immutable protocol*

Description

Create an isolated synthetic study and its first immutable protocol

Usage

```
create_study(repo, actor, protocol, command_id)
```

Arguments

`repo`	Repository.
`actor`	Authorized server actor.
`protocol`	Validated demo protocol.
`command_id`	Idempotency key.

Value

Study id and protocol id.

`del_abort` *Structured delphyr error*

Description

Structured delphyr error

Usage

```
del_abort(code, path = "")
```

Arguments

`code`	Stable condition code.
`path`	Field path, without sensitive values.

Value

Does not return; signals a structured error.

Examples

```
tryCatch(del_abort("DEL_VALIDATION", "scale"), error = function(e) e$code)
```

demo_actor *Create a short-lived, server-side demo actor*

Description

Create a short-lived, server-side demo actor

Usage

```
demo_actor(repo, principal_id, lifetime_seconds = 3600)
```

Arguments

repo	Development/test repository.
principal_id	Provisioned synthetic principal UUID.
lifetime_seconds	Positive session duration, at most one hour.

Value

Actor context for trusted server code, never accepted from browser payloads.

demo_protocol *Synthetic protocol example*

Description

Synthetic protocol example

Usage

```
demo_protocol()
```

Value

Complete demo configuration; thresholds are examples, not recommendations.

Examples

```
validate_protocol(demo_protocol())$valid
```

demo_snapshot *Synthetic offline snapshot*

Description

Synthetic offline snapshot

Usage

```
demo_snapshot(
  values = c(1, 4, 6, 7, 7, 8, 8, 9, 9, 9),
  groups = rep("professionals", length(values))
)
```

Arguments

values	Integer ratings; NA creates an explicit not_answered response.
groups	Optional group assignments.

Value

Validated synthetic snapshot.

Examples

```
analyse_round(demo_snapshot())$results
```

demo_study *Create a synthetic two-group development study*

Description

Create a synthetic two-group development study

Usage

```
demo_study(
  repo,
  n = 30L,
  item_count = 12L,
  code = paste0("DEMO-", substr(uid(), 1, 8))
)
```

Arguments

repo	Development/test repository.
n	Number of synthetic panel accounts.
item_count	Number of trilingual rating items.
code	Unique study code.

Value

Manager and panel actors, study/consent ids and unapproved round.

---

download_artifact *Resolve a private artifact only after a fresh rights check*

Description

Resolve a private artifact only after a fresh rights check

Usage

```
download_artifact(repo, actor, artifact_id)
```

Arguments

repo	Repository.
actor	Original requesting actor with current export capability.
artifact_id	Artifact UUID.

Value

Private directory path for trusted server download code. An export whose worker stopped after registering it but before giving it its final name is delivered from its temporary name, '<id>.partial'.

---

enroll_panel *Enroll eligible panel members who joined after a round was prepared*

Description

Applies the round's protocol entry policy and each member's current stakeholder group. Enrollment, withdrawal and group changes of one study take effect one after the other: a member who withdraws meanwhile is either not enrolled or enrolled and withdrawn. Enrollment is possible until the round closes. Feedback already assigned to an unopened round is assigned to the new enrollments as well; an open round with assigned feedback accepts no further enrollment, so that everyone rating in a round has the same information.

Usage

```
enroll_panel(repo, actor, round_id, command_id)
```

Arguments

repo	Repository.
actor	Study manager.
round_id	Round UUID.
command_id	Idempotency key.

Value

Round UUID and the number of enrollments added.

---

freeze_round *Freeze the exact submitted revision set of a closed round*

Description

Freeze the exact submitted revision set of a closed round

Usage

```
freeze_round(repo, actor, round_id, command_id)
```

Arguments

repo	Repository.
actor	Study manager.
round_id	Closed round UUID.
command_id	Idempotency key.

Value

Immutable snapshot UUID and content hash.

get_account_rights *Rights of the signed-in account outside any study*

Description

Rights of the signed-in account outside any study

Usage

```
get_account_rights(repo, actor)
```

Arguments

repo	Repository.
actor	Server actor.

Value

List with can_create_study. Creating a study is granted by the operator to an account, never chosen in the interface.

get_analysis *Read a persisted aggregate analysis*

Description

Read a persisted aggregate analysis

Usage

```
get_analysis(repo, actor, analysis_id)
```

Arguments

repo	Repository.
actor	Actor with analyse capability.
analysis_id	Analysis UUID.

Value

delphyr_analysis.

get_capabilities *Read current study capabilities*

Description

Read current study capabilities

Usage

```
get_capabilities(repo, actor, study_id)
```

Arguments

repo	Repository.
actor	Server actor.
study_id	Study UUID.

Value

Character vector; this UI hint does not replace service authorization.

get_feedback *Read released aggregate feedback plus only the actor's prior answers*

Description

Read released aggregate feedback plus only the actor's prior answers

Usage

```
get_feedback(repo, actor, enrollment_id)
```

Arguments

repo	Repository.
actor	Panel actor.
enrollment_id	Own target enrollment in a round that was opened.

Value

Aggregate feedback, own prior response table, the comparability decisions for revised items that were in force when the target round opened, the released qualitative content and, where the feedback was corrected, the note for participants; or NULL.

`get_feedback_candidate` *Preview the exact reviewed feedback artifact before release*

Description

Preview the exact reviewed feedback artifact before release

Usage

```
get_feedback_candidate(repo, actor, feedback_id)
```

Arguments

`repo`	Repository.
`actor`	Study manager or analyst; only a manager releases it.
`feedback_id`	Feedback UUID.

Value

Content, hash and state of the candidate.

`get_item_comparability` *Read the comparability decisions of a study*

Description

Read the comparability decisions of a study

Usage

```
get_item_comparability(repo, actor, study_id)
```

Arguments

`repo`	Repository.
`actor`	Actor with manage, analyse, export, edit or audit capability.
`study_id`	Study UUID.

Value

Decision history; effective marks the latest decision per version pair.

get_operation *Retrieve an authorized operation state*

Description

Retrieve an authorized operation state

Usage

```
get_operation(repo, actor, operation_id)
```

Arguments

repo	Repository.
actor	Requesting actor.
operation_id	Operation UUID.

Value

State, result reference and safe error code.

get_operations_status *Operational status of a study's background work*

Description

Operational status of a study's background work

Usage

```
get_operations_status(repo, actor, study_id)
```

Arguments

repo	Repository.
actor	Actor with manage or coordinate capability.
study_id	Study UUID.

Value

Counts of jobs by type and state with the age of the oldest waiting job, counts of messages by delivery state with the age of the oldest waiting message, and the number of uncertain deliveries. No content.

get_panel_import_receipt

Retrieve a minimal panel import receipt through current coordination rights

Description

Retrieve a minimal panel import receipt through current coordination rights

Usage

```
get_panel_import_receipt(repo, actor, receipt_id)
```

Arguments

repo	Repository.
actor	Study coordinator.
receipt_id	Receipt UUID.

Value

Receipt metadata and invitation IDs without contact values.

get_qualitative_provenance

Read restricted qualitative provenance or reviewed export tables

Description

Read restricted qualitative provenance or reviewed export tables

Usage

```
get_qualitative_provenance(repo, actor, study_id, export = FALSE)
```

Arguments

repo	Repository.
actor	Actor with edit capability; export also requires export capability.
study_id	Study UUID.
export	FALSE includes originals for editorial review. TRUE excludes originals and unreleased redactions and returns authorized export tables.

Value

Named tables with explicit source, coding, item and review provenance. These are restricted research records, not participant feedback.

get_questionnaire *Retrieve only the actor's current questionnaire and consent information*

Description

Retrieve only the actor's current questionnaire and consent information

Usage

```
get_questionnaire(repo, actor, enrollment_id)
```

Arguments

repo	Repository.
actor	Panel actor.
enrollment_id	Own enrollment UUID in a round that was opened; any other enrollment is refused with DEL_NOT_FOUND, as an unknown one is.

Value

Instrument, own responses, receipt and required consent information.

get_retention_report *Inventory of a study's data for retention decisions*

Description

Lists each class of stored data with its count and the age of its oldest and newest record. With 'periods', a proposed number of days per class, it also states how many records would be older than that period. It is a dry run: nothing is deleted, and the software ships no retention periods. A period is the decision of the responsible organisation.

Usage

```
get_retention_report(repo, actor, study_id, periods = NULL)
```

Arguments

repo	Repository.
actor	Actor with manage capability.
study_id	Study UUID.
periods	Optional named numeric vector of days per data class.

Value

A table with data_class, records, oldest, newest, period_days and older_than_period.

get_round_instrument *Read the exact instrument of a round for review*

Description

Read the exact instrument of a round for review

Usage

```
get_round_instrument(repo, actor, round_id)
```

Arguments

repo	Repository.
actor	Study manager.
round_id	Round UUID.

Value

Round number, state, deadline, content hash, protocol version, study information and every item with all approved language versions. Contains no panel identities or responses.

get_round_readiness *Review whether a round may be approved or opened*

Description

Reports every finding at once instead of failing on the first. Errors block the transition; warnings describe a foreseeable methodological consequence, such as a required group below the minimum valid n, and leave the decision to the study team.

Usage

```
get_round_readiness(repo, actor, round_id)
```

Arguments

repo	Repository.
actor	Study manager.
round_id	Round UUID.

Value

delphyr_validation: valid, issues (severity, code, path, message_key, details) and schema_version. Approval ignores the deadline_passed and no_enrollments findings; opening requires none.

get_snapshot *Retrieve a frozen snapshot through current study authorization*

Description

Retrieve a frozen snapshot through current study authorization

Usage

```
get_snapshot(repo, actor, snapshot_id)
```

Arguments

repo	Repository.
actor	Actor with analyse capability.
snapshot_id	Snapshot UUID.

Value

Validated delphyr_snapshot.

get_study_documentation *Read the current study documentation and its version history*

Description

Read the current study documentation and its version history

Usage

```
get_study_documentation(repo, actor, study_id)
```

Arguments

repo	Repository.
actor	Actor with manage, analyse, export or audit capability.
study_id	Study UUID.

Value

Current version, hash and fields; a topic table that marks every reporting topic documented or not documented; and the version history.

`get_study_setup` *Read authorized setup information for the round editor*

Description

Analysts read it as well: the protocol holds the rules and the time zone of the analyses they work with.

Usage

```
get_study_setup(repo, actor, study_id)
```

Arguments

`repo`	Repository.
`actor`	Study manager or analyst.
`study_id`	Study UUID.

Value

Protocol and published information versions, without panel identities.

`get_system_status` *Read the operational state of an installation*

Description

For monitoring and the first look at an incident. The result holds counts and ages only: no study content, contact, account or study identifier. ‘ready‘ is TRUE when the database answers and its schema is exactly the one this software version ships. ‘attention‘ lists what a person should look at; it does not make the application unready. A database whose schema is not the one of this software, for example one of the previous release, is reported with its schema state alone: the tables of later migrations may not exist there yet. ‘artifacts‘ counts exports that can be downloaded, expired ones whose files were not yet removed, removed ones, and ‘unfinished‘: directories in the private export directory of ‘repo‘ that no registration covers, left by a stopped worker or an earlier release (see remove_unfinished_exports()).

Usage

```
get_system_status(
  repo,
  path = system.file("sql", package = "delphyr"),
  heartbeat_seconds = 60L
)
```

Arguments

repo	Repository of the application or worker role.
path	Directory of the packaged migrations.
heartbeat_seconds	Age after which a worker counts as silent.

Value

A list with ‘ready‘, ‘schema‘, ‘workers‘, ‘jobs‘, ‘messages‘, ‘artifacts‘ and ‘attention‘; only ‘ready‘, ‘schema‘ and ‘attention‘ when the schema is not current or the database does not answer.

---

hold_pending_messages *Put messages that wait for delivery on hold for a human decision*

Description

After a database was restored from a backup, a message that was waiting when the backup was taken may have been delivered before the loss. This function turns every message that is queued or in flight into an uncertain delivery with the cause ‘restored_from_backup‘. The worker then sends none of them, and a coordinator decides for each one in the interface: delivery confirmed, send again, or abandon. Run it before the worker is started on a restored database. Without ‘dry_run = FALSE‘ it changes nothing.

Usage

```
hold_pending_messages(repo, dry_run = TRUE, study_id = NULL)
```

Arguments

repo	Trusted worker repository.
dry_run	TRUE counts what would be put on hold.
study_id	Optional study UUID; otherwise every study.

Value

A list with the numbers of messages ‘queued‘ and ‘in_flight‘ that were found and the number ‘held‘, which is zero for a dry run.

`import_panel` *Atomically import exactly reviewed synthetic contacts as unbound drafts*

Description

Atomically import exactly reviewed synthetic contacts as unbound drafts

Usage

```
import_panel(repo, actor, study_id, preview, expected_hash, reason, command_id)
```

Arguments

`repo`	Development/test repository.
`actor`	Current study coordinator.
`study_id`	Study UUID.
`preview`	Output of preview_panel_import().
`expected_hash`	Exact preview hash explicitly reviewed by the coordinator.
`reason`	Approval rationale.
`command_id`	Idempotency key for this approved import.

Value

Receipt id, file/preview hashes, counts and opaque invitation-draft IDs.

`import_round_contributions` *Preserve the free-text answers of a frozen round as qualitative sources*

Description

Every submitted free-text answer of the snapshot becomes one immutable original source, referenced by round, item and an opaque suffix. The sources carry no pseudonym; editorial work, redaction and independent release follow as for any other source. Answers that already have a source are skipped, so the import can be repeated.

Usage

```
import_round_contributions(repo, actor, snapshot_id, command_id)
```

Arguments

repo	Repository.
actor	Actor with edit capability.
snapshot_id	Snapshot UUID of a frozen round.
command_id	Idempotency key.

Value

Snapshot UUID and the number of sources created.

---

invitation_code *Encode an invitation hand-over code*

Description

The code combines the study, invitation and private token so that the invited person enters one value after verified login. Treat the whole code as secret.

Usage

```
invitation_code(study_id, invitation_id, token)
```

Arguments

study_id, invitation_id	UUIDs returned by the issuing services.
token	Private output of new_invitation_token().

Value

A single private string with redacted printing; as.character() reveals it.

---

issue_panel_invitation *Issue a synthetic invitation for an explicitly approved existing account*

Description

Issue a synthetic invitation for an explicitly approved existing account

Usage

```
issue_panel_invitation(
  repo,
  actor,
  study_id,
  draft_id,
  principal_id,
  token,
  ttl_seconds = 900L,
  reason,
  command_id
)
```

Arguments

repo	Repository.
actor	Study coordinator.
study_id	Study UUID.
draft_id	Imported invitation draft UUID.
principal_id	Existing provisioned principal approved by stable issuer/subject.
token	Private output of new_invitation_token(). Never persisted in plaintext.
ttl_seconds	Lifetime in seconds, between one second and 24 hours.
reason	Documented account-binding approval; do not include credentials.
command_id	Idempotency key.

Value

Invitation UUID and expiry, without the token or contact fields.

---

link_item_source *Link an imported or proposed item version to its original source*

Description

Link an imported or proposed item version to its original source

Usage

```
link_item_source(
  repo,
  actor,
  study_id,
  item_code,
  item_version,
```

```
  source_id,
  reason,
  command_id
)
```

Arguments

repo	Repository.
actor	Actor with edit capability.
study_id	Study UUID.
item_code	Stable existing or proposed item code.
item_version	Positive version number.
source_id	Original qualitative source UUID.
reason	Rationale for derivation or consolidation.
command_id	Idempotency key.

Value

Source-link UUID. This does not import or approve an instrument item.

---

list_audit_events *Read the audit trail of a study*

Description

Every event names time, action, object and actor, with the recorded rationale where the action required one. Staff are referenced by their account UUID. Panel actions carry no account reference, and worker events reference the person who approved the work.

Usage

```
list_audit_events(
  repo,
  actor,
  study_id,
  from = NULL,
  to = NULL,
  actions = NULL,
  limit = 500L
)
```

Arguments

repo	Repository.
actor	Actor with audit or manage capability.
study_id	Study UUID.
from, to	Optional inclusive time limits with an explicit offset, for example "2026-10-02T00:00:00Z". A time that does not exist, such as 30 February or an offset beyond 15:59, is refused with DEL_VALIDATION (path audit.from or audit.to) before any statement.
actions	Optional action codes to include.
limit	Maximum number of events, newest first, from 1 to 100000.

Value

Data frame occurred_at, action, detail, object_ref, reason, actor_kind (staff, panel or worker) and actor_ref.

---

list_campaign_enrollments

Preview eligible campaign selection fields without contacts or answers

---

Description

Preview eligible campaign selection fields without contacts or answers

Usage

```
list_campaign_enrollments(repo, actor, round_id)
```

Arguments

repo	Repository.
actor	Actor with coordinate capability.
round_id	Round UUID.

Value

Enrollment UUID, study pseudonym, participation state and the number of approved reminders that count against the protocol's limit, including reminders delivered before their campaign was cancelled.

list_campaign_rounds *List rounds available to a campaign coordinator*

Description

List rounds available to a campaign coordinator

Usage

```
list_campaign_rounds(repo, actor, study_id)
```

Arguments

repo	Repository.
actor	Actor with coordinate capability.
study_id	Study UUID.

Value

Minimal round identifiers, numbers and states.

list_contribution_rounds *List frozen rounds with free-text contributions*

Description

List frozen rounds with free-text contributions

Usage

```
list_contribution_rounds(repo, actor, study_id)
```

Arguments

repo	Repository.
actor	Actor with edit capability.
study_id	Study UUID.

Value

Round number, snapshot UUID and the number of submitted free-text answers, with how many are already preserved as qualitative sources.

list_enrollments *List own round enrollments*

Description

Lists only rounds that were opened. A round that is still being prepared, reviewed or approved, or that was withdrawn, is not shown to panel members, and its instrument cannot be read through any panel service.

Usage

```
list_enrollments(repo, actor, study_id)
```

Arguments

repo	Repository.
actor	Panel actor.
study_id	Selected study.

Value

Enrollment table.

list_item_decisions *List the human item decisions of a study*

Description

List the human item decisions of a study

Usage

```
list_item_decisions(repo, actor, study_id)
```

Arguments

repo	Repository.
actor	Actor with manage or analyse capability.
study_id	Study UUID.

Value

Round number, item, disposition, rationale and time of each decision.

list_panel *List the panel of a study by pseudonym*

Description

List the panel of a study by pseudonym

Usage

```
list_panel(repo, actor, study_id)
```

Arguments

repo	Repository.
actor	Study manager.
study_id	Study UUID.

Value

Study pseudonym, stakeholder group, whether the member is active or withdrew, and counts of rounds enrolled and submitted. No account references, contacts or responses.

list_panel_invitations *List imported invitation drafts and their current invitation state*

Description

List imported invitation drafts and their current invitation state

Usage

```
list_panel_invitations(repo, actor, study_id)
```

Arguments

repo	Repository.
actor	Study coordinator.
study_id	Study UUID.

Value

One row per draft: source reference, display name, group, locale, latest invitation id, expiry and state (unbound, outstanding, expired, revoked or accepted). Never contains tokens, hashes, emails or pseudonyms.

`list_protocol_versions` *Read the immutable study protocol history*

Description

Read the immutable study protocol history

Usage

```
list_protocol_versions(repo, actor, study_id)
```

Arguments

`repo`	Repository.
`actor`	Study manager.
`study_id`	Study UUID.

Value

Version table with content hashes, configurations and amendment reasons.

`list_qualitative_reviews` *List exact redacted versions for independent management review*

Description

List exact redacted versions for independent management review

Usage

```
list_qualitative_reviews(repo, actor, study_id)
```

Arguments

`repo`	Repository.
`actor`	Actor with manage capability.
`study_id`	Study UUID.

Value

Reviewed content candidates and release state; excludes originals.

list_released_edits *List released editorial versions available for participant feedback*

Description

List released editorial versions available for participant feedback

Usage

```
list_released_edits(repo, actor, study_id)
```

Arguments

repo	Repository.
actor	Actor with analyse or manage capability.
study_id	Study UUID.

Value

Edit UUID, source reference, kind, released text, release time and the item codes linked to the source. Originals are not included.

list_released_feedback *Read the released feedback of a round and its corrections*

Description

Read the released feedback of a round and its corrections

Usage

```
list_released_feedback(repo, actor, study_id)
```

Arguments

repo	Repository.
actor	Study manager or analyst.
study_id	Study UUID.

Value

One row per released feedback: round, UUID, hash, the round it is assigned to and, where it was corrected, its replacement with rationale.

list_rounds *List study round states and frozen result references*

Description

List study round states and frozen result references

Usage

```
list_rounds(repo, actor, study_id)
```

Arguments

repo	Repository.
actor	Study manager or analyst. Analysts read the list to find the frozen snapshots and analyses of their work; it holds no panel data.
study_id	Study UUID.

Value

Round table.

list_studies *List studies available to the current actor*

Description

List studies available to the current actor

Usage

```
list_studies(repo, actor)
```

Arguments

repo	Repository.
actor	Server actor.

Value

Study table without contacts or response data.

list_study_staff *List study staff and their current rights*

Description

List study staff and their current rights

Usage

```
list_study_staff(repo, actor, study_id)
```

Arguments

repo	Repository.
actor	Study manager.
study_id	Study UUID.

Value

One row per staff account: principal UUID, issuer, subject, whether the account and membership are active, and granted and revoked rights. Panel membership is not part of this list.

list_uncertain_deliveries *List deliveries whose outcome is uncertain*

Description

List deliveries whose outcome is uncertain

Usage

```
list_uncertain_deliveries(repo, actor, study_id)
```

Arguments

repo	Repository.
actor	Actor with coordinate capability.
study_id	Study UUID.

Value

Message UUID, campaign kind, round number, study pseudonym, the cause, the adapter and the time of the uncertain attempt.

log_event *Write one entry of the technical log*

Description

An entry holds the time, a correlation ID, the component, the operation, its duration and, for a refused or failed operation, the condition class and field path. Arguments, results, condition messages, account references and request headers are never written.

Usage

```
log_event(
  operation,
  error = NULL,
  correlation_id = new_correlation_id(),
  component = "service",
  duration_ms = NULL,
  reference = NULL
)
```

Arguments

operation	Stable operation name, for example "save_response".
error	Condition of a refused or failed operation; NULL for success.
correlation_id	Output of new_correlation_id().
component	Part of the system writing the entry, for example "app".
duration_ms	Duration in milliseconds, if measured.
reference	UUID of the job or message concerned, if any.

Details

Entries are written to standard error as one JSON object per line. Set ‘options(delphyr.log_sink = function(line) ...)‘ to route them elsewhere. ‘options(delphyr.log_level =)‘ or the environment variable ‘DELPHYR_LOG_LEVEL‘ selects ‘off‘, ‘error‘ (failures), ‘warning‘ (refusals and failures; the default) or ‘info‘ (every operation).

Value

The correlation ID, invisibly.

Examples

```
options(delphyr.log_sink = function(line) NULL)
log_event("example", error = simpleError("not written"))
options(delphyr.log_sink = NULL)
```

log_operation *Run one operation with an entry in the technical log*

Description

Measures the operation and writes one entry, see log_event(). A refused or failed operation signals its original condition again, extended by the field ‘correlation_id‘, so that the message shown to a person can quote it.

Usage

```
log_operation(
  operation,
  code,
  correlation_id = new_correlation_id(),
  component = "service"
)
```

Arguments

operation	Stable operation name, for example "save_response".
code	Function without arguments performing the operation.
correlation_id	Output of new_correlation_id().
component	Part of the system writing the entry, for example "app".

Value

The value of code().

Examples

```
options(delphyr.log_sink = function(line) NULL)
log_operation("example", function() 1 + 1)
options(delphyr.log_sink = NULL)
```

migrate_repository *Apply checksummed forward migrations*

Description

Apply checksummed forward migrations

Usage

```
migrate_repository(repo, path = system.file("sql", package = "delphyr"))
```

Arguments

repo	Repository connected as migration owner.
path	Directory of SQL migrations, normally installed package resources.

Value

Invisibly TRUE. Changed historical migration files are rejected.

---

new_authentication_config
Configure the trusted authentication gateway boundary

Description

Configure the trusted authentication gateway boundary

Usage

```
new_authentication_config(
  issuer,
  gateway_secret,
  trusted_proxy_addresses,
  max_session_seconds = 900L
)
```

Arguments

issuer	Exact OIDC issuer configured in the independently validating proxy.
gateway_secret	Private random secret overwritten by the internal gateway.
trusted_proxy_addresses	Exact transport peer addresses observed by httpuv.
max_session_seconds	Maximum lifetime of one Shiny actor, at most one hour.

Value

Server-only delphyr_authentication_config. Never serialize to a browser.

new_correlation_id *Create a correlation ID for the technical log*

Description

The ID is random. It is derived from no account, study or content and can be shown to a person and quoted in a support request.

Usage

```
new_correlation_id()
```

Value

Twelve lowercase hexadecimal characters.

Examples

```
new_correlation_id()
```

new_invitation_token *Generate a private synthetic invitation token*

Description

Generate a private synthetic invitation token

Usage

```
new_invitation_token()
```

Value

A 256-bit secret with redacted printing. Retain privately; only its hash is persisted. Never put the token in logs, audit records or research exports.

new_message_adapter *Define a message provider adapter*

Description

The contract for a delivery provider. ‘send‘ receives one message as a list with message_id, idempotency_key, kind, locale, subject, body, to and display_name, and must return ‘list(status = "accepted", provider_ref =)‘, ‘list(status = "rejected", reason =)‘ for a permanent refusal or ‘list(status = "retry", reason =)‘ for a transient failure. The same idempotency key is passed on every attempt for a message. An error, a timeout or any other return value leaves the delivery uncertain; it is then never repeated automatically. No production adapter is shipped or approved: contact addresses are restricted to the reserved ‘.invalid‘ domain.

Usage

```
new_message_adapter(name, send)
```

Arguments

name	Short adapter name recorded with each delivery.
send	Function of one message, as described.

Value

A delphyr_message_adapter for process_campaign_message().

new_protocol *Construct a protocol*

Description

Construct a protocol

Usage

```
new_protocol(config)
```

Arguments

config	Complete declarative configuration.

Value

delphyr_protocol; invalid inputs raise DEL_VALIDATION.

Examples

```
new_protocol(demo_protocol())$schema_version
```

new_scale *Construct a validated response scale*

Description

Construct a validated response scale

Usage

```
new_scale(
  type,
  levels = integer(),
  labels = as.character(levels),
  missing_options = c("unable_to_judge", "abstained")
)
```

Arguments

type	One of ordinal_integer, binary, free_text.
levels	Ordered integer categories within R's non-missing integer range; empty for text.
labels	Category labels, one per level.
missing_options	Allowed explicit missing categories.

Value

A delphyr_scale. Invalid input raises DEL_VALIDATION.

Examples

```
new_scale("ordinal_integer", 1:9, as.character(1:9))
```

new_snapshot *Construct an immutable analysis input*

Description

Construct an immutable analysis input

Usage

```
new_snapshot(
  responses,
  assignments,
  items,
  protocol,
  round_number = 1L,
  snapshot_id = NULL,
  hash_version = "2"
)
```

Arguments

responses	Long data frame: panelist_id, item_code, item_version, dimension_code, answer_status, value_integer, value_text. Only submitted data.
assignments	All assigned panelists: panelist_id, group_code, submitted.
items	Instrument table: item_code, item_version, dimension_code, scale_code. Item versions must be positive and within R's integer range.
protocol	Validated protocol.
round_number	Positive round number.
snapshot_id	Stable identifier, or content hash when NULL.
hash_version	"2" for new snapshots; "1" only to reconstruct an existing legacy record without changing its scientific identity.

Value

delphyr_snapshot, with all assigned fields materialized, including missingness.

Examples

```
s <- demo_snapshot()
s$content_hash
```

---

parse_invitation_code *Decode an invitation hand-over code*

Description

Decode an invitation hand-over code

Usage

```
parse_invitation_code(code)
```

Arguments

code	Value produced by invitation_code(); surrounding space is ignored.

Value

List study_id, invitation_id and token; DEL_UNAUTHORIZED when malformed.

---

prepare_campaign *Prepare an exact synthetic communication campaign*

Description

Prepare an exact synthetic communication campaign

Usage

```
prepare_campaign(
  repo,
  actor,
  round_id,
  enrollment_ids,
  kind,
  subject,
  body,
  locale = "en",
  template_version = 1L,
  command_id
)
```

Arguments

repo	Development/test repository.
actor	Actor with coordinate capability.
round_id	Study round UUID.
enrollment_ids	Exact recipient enrollment UUIDs; no addresses accepted.
kind	invitation, round_start, reminder, deadline_change or completion.
subject, body	Final plain text without unresolved template placeholders.
locale	en, fr or de.
template_version	Positive immutable template version.
command_id	Idempotency key.

Value

Campaign id, exact content/recipient hash and recipient count.

prepare_feedback *Prepare disclosure-controlled feedback*

Description

Prepare disclosure-controlled feedback

Usage

```
prepare_feedback(
  analysis,
  qualitative = list(),
  policy = list(group_statistics = FALSE, minimum_display_cell_n = 5L,
    complementary_suppression = TRUE)
)
```

Arguments

analysis	A delphyr_analysis.
qualitative	List of reviewed summaries, each with text, reviewed and source_ref, and optionally kind, edit_id, content_hash and item_codes.
policy	List group_statistics, minimum_display_cell_n, complementary_suppression.

Value

delphyr_feedback_draft. Suppressed data are removed, not hidden in markup.

Examples

```
prepare_feedback(analyse_round(demo_snapshot()))$hash
```

prepare_report_data *Prepare an authorized aggregate report and numeric export supplement*

Description

Prepare an authorized aggregate report and numeric export supplement

Usage

```
prepare_report_data(repo, actor, snapshot_id)
```

Arguments

repo	Repository.
actor	Actor with current export capability.
snapshot_id	Immutable numeric snapshot UUID.

Value

A delphyr_report_data with allowlisted tables and a content hash.

---

prepare_round *Prepare a frozen-instrument candidate and enroll active panelists*

Description

Prepare a frozen-instrument candidate and enroll active panelists

Usage

```
prepare_round(
  repo,
  actor,
  study_id,
  items,
  consent_version_id,
  deadline,
  command_id
)
```

Arguments

repo	Repository.
actor	Study manager.
study_id	Study UUID.
items	Validated multilingual import data frame.
consent_version_id	Required study information version.
deadline	UTC timestamp string with explicit offset.
command_id	Idempotency key.

Value

Round UUID and exact instrument hash.

prepare_study_report_data *Prepare the data of a study-level report*

Description

Covers every frozen round: design, protocol versions, recruitment and participation, instruments, results, comparisons between rounds, human decisions, lineage, released feedback, final items and the author-supplied documentation. Topics without an entry stay marked "not documented".

Usage

```
prepare_study_report_data(repo, actor, study_id, profile = "study_summary")
```

Arguments

repo	Repository.
actor	Actor holding the capability of the profile.
study_id	Study UUID.
profile	study_summary (aggregates with small-cell suppression) or research_pseudonymized (full denominators and decision rationales).

Value

A delphyr_report_data with a content hash, for render_study_report().

preview_campaign *Preview exact campaign text and pseudonymous recipients*

Description

Preview exact campaign text and pseudonymous recipients

Usage

```
preview_campaign(repo, actor, campaign_id)
```

Arguments

repo	Repository.
actor	Actor with coordinate capability.
campaign_id	Campaign UUID.

Value

Plain text, hash, the study's reminder rules, the approved send window, pseudonyms and delivery states; no contacts or answers.

preview_panel_import *Preview a synthetic panel CSV with blocking duplicate review*

Description

Preview a synthetic panel CSV with blocking duplicate review

Usage

```
preview_panel_import(
  repo,
  actor,
  study_id,
  csv,
  schema_version = "1.0",
  delimiter = ","
)
```

Arguments

repo	Development/test repository.
actor	Current study coordinator.
study_id	Study UUID.
csv	UTF-8 CSV bytes or one CSV text string, not a file path.
schema_version	Explicit supported schema, 1.0.
delimiter	Explicit comma or semicolon.

Value

Restricted preview with rows, row/column issues, exact file hash and approval hash.

preview_panel_invitation *Preview an invitation without creating membership or consuming its token*

Description

Preview an invitation without creating membership or consuming its token

Usage

```
preview_panel_invitation(repo, actor, study_id, invitation_id, token)
```

Arguments

repo	Repository.
actor	Study coordinator.
study_id	Study UUID.
invitation_id	Invitation UUID, not a panelist identifier.
token	Private output of new_invitation_token(). Never persisted in plaintext.

Value

Study title and invitation expiry only, for the exact approved account.

---

process_campaign_message *Process one approved message through a delivery adapter*

Description

Claim, eligibility check and result are separate short transactions; no database transaction is held while the provider is called. Immediately before sending, the worker rechecks cancellation, approval authority, participation, consent, submission and round state, which can only reduce the approved audience. A transient failure is retried with a growing delay up to three attempts.

Usage

```
process_campaign_message(
  repo,
  adapter = sink_adapter(),
  study_id = NULL,
  lease_seconds = 30L
)
```

Arguments

repo	Trusted separate development/test worker repository.
adapter	Adapter from new_message_adapter(); the default is the local database sink.
study_id	Optional study UUID restricting the worker partition.
lease_seconds	Claim duration from 1 to 300 seconds.

Value

FALSE when idle, or message id, state and reason.

process_campaign_sink *Process one approved message into a local database sink*

Description

Process one approved message into a local database sink

Usage

```
process_campaign_sink(repo, study_id = NULL, lease_seconds = 30L)
```

Arguments

repo	Trusted separate development/test worker repository.
study_id	Optional study UUID restricting the worker partition.
lease_seconds	Claim duration from 1 to 300 seconds.

Value

FALSE when idle or a message id/state/reason. Never sends external mail.

provision_demo_principal
Provision a synthetic development principal

Description

Provision a synthetic development principal

Usage

```
provision_demo_principal(repo, subject, can_create = FALSE)
```

Arguments

repo	Development/test repository, using a provisioning connection.
subject	Synthetic account name, prefixed demo-.
can_create	Allow creating demo studies.

Value

Principal UUID. This is not a production authentication adapter.

`publish_consent` *Publish immutable synthetic study information*

Description

Publish immutable synthetic study information

Usage

```
publish_consent(repo, actor, study_id, text, locale, command_id)
```

Arguments

`repo`	Repository.
`actor`	Study manager.
`study_id`	Study UUID.
`text`	Displayed information, explicitly synthetic in development.
`locale`	en, fr or de.
`command_id`	Idempotency key.

Value

Consent version id. A completed study refuses new versions with DEL_CONFLICT (path study.state).

`read_item_import` *Read an instrument import file without guessing types*

Description

Reads a UTF-8 CSV item file. Every value is read as text, so codes keep their leading zeros and a text such as "NA" stays text. Required columns are ‘item_code‘, ‘item_version‘, ‘locale‘, ‘text‘, ‘dimension_code‘, ‘scale_code‘, ‘source_ref‘, ‘required‘ and ‘display_order‘. This differs from the v1 item import contract of the specification in two ways: ‘item_version‘ is an additional column, because a version names the rated wording across rounds, and ‘source_ref‘ is required, not optional. A file that lacks either column is refused, as is the specification’s v1 example, which has no ‘item_version‘. ‘item_version‘ and ‘display_order‘ must be written as whole numbers and ‘required‘ as true or false. The optional columns ‘definition‘ and ‘domain_code‘ are accepted and not stored. Check the result with validate_items() before preparing a round.

Usage

```
read_item_import(file, delimiter = ",")
```

Arguments

file	Path of the CSV file, at most 5 MB.
delimiter	"," (the canonical format) or ";".

Value

Data frame for validate_items() and prepare_round(). A file that does not follow the contract is refused with DEL_VALIDATION, path items.file, items.encoding, items.columns, items.rows or the column of the first value that is not of its type, such as items.display_order.

---

record_consent *Record an explicit consent decision*

Description

Record an explicit consent decision

Usage

```
record_consent(repo, actor, study_id, consent_version_id, decision, command_id)
```

Arguments

repo	Repository.
actor	Panel actor.
study_id	Study UUID.
consent_version_id	Displayed version UUID.
decision	TRUE for consent, FALSE for refusal or withdrawal.
command_id	Idempotency key.

Value

Consent event id.

`record_item_comparability` *Decide whether two versions of an item may be compared directly*

Description

A changed item version interrupts paired comparison by default. This records the study team's explicit decision for one version pair, with its rationale. A later decision for the same pair supersedes the earlier one; the history is kept and exported.

Usage

```
record_item_comparability(
  repo,
  actor,
  study_id,
  item_code,
  dimension_code,
  previous_version,
  current_version,
  comparable,
  reason,
  command_id
)
```

Arguments

`repo`	Repository.
`actor`	Study manager.
`study_id`	Study UUID.
`item_code`, `dimension_code`	Item and rating dimension.
`previous_version`, `current_version`	Distinct versions used in rounds of this study.
`comparable`	TRUE if ratings of both versions may be paired.
`reason`	Rationale for this version.
`command_id`	Idempotency key.

Value

Decision UUID.

record_item_decision *Record a human decision separately from analytic classification*

Description

Record a human decision separately from analytic classification

Usage

```
record_item_decision(
  repo,
  actor,
  analysis_id,
  item_code,
  disposition,
  reason,
  command_id
)
```

Arguments

repo	Repository.
actor	Study manager.
analysis_id	Evidence reference.
item_code	Item being decided.
disposition	retain, revise, remove, split, merge, rerate or finalize.
reason	Human rationale.
command_id	Idempotency key.

Value

Decision UUID. A completed study refuses new decisions with DEL_CONFLICT (path study.state).

record_item_lineage *Record a complete item split or merge without deleting its parents*

Description

Record a complete item split or merge without deleting its parents

Usage

```
record_item_lineage(
  repo,
  actor,
  study_id,
  parents,
  children,
  relation,
  reason,
  command_id
)
```

Arguments

repo	Repository.
actor	Actor with manage capability.
study_id	Study UUID.
parents, children	Data frames with item_code and item_version.
relation	split (one to at least two) or merge (at least two to one).
reason	Human decision rationale.
command_id	Idempotency key.

Value

Lineage-event UUID and hash. Child codes are new semantic identities.

---

record_qualitative_source

Preserve an original synthetic qualitative contribution

Description

Preserve an original synthetic qualitative contribution

Usage

```
record_qualitative_source(
  repo,
  actor,
  study_id,
  text,
  source_ref,
  command_id,
  response_revision_id = NULL
)
```

Arguments

repo	Repository.
actor	Actor with edit capability.
study_id	Study UUID.
text	Original text, preserved without redaction.
source_ref	Explicit provenance reference.
command_id	Idempotency key.
response_revision_id	Optional immutable free-text response revision.

Value

Source UUID and content hash. This is restricted editorial data.

---

record_study_documentation *Record author-supplied study documentation for reports*

Description

Reports never invent authorship, funding, conflicts of interest, approvals, interpretation or deviations. Study staff record these statements here; each change is a new immutable version, and topics without an entry stay marked "not documented". Recording a reference to an approval does not grant one.

Usage

```
record_study_documentation(
  repo,
  actor,
  study_id,
  fields,
  expected_version,
  reason,
  command_id
)
```

Arguments

repo	Repository.
actor	Study manager.
study_id	Study UUID.

`fields`	Named list of plain text. Allowed names: authors_responsibilities, panel_criteria_recruitment, funding, conflicts_of_interest, institutional_approval, methodological_interpretation, protocol_deviations, reporting_guideline_review and data_software_availability. The list replaces the previous version completely; empty entries are omitted.
`expected_version`	Version shown during editing; 0 for the first.
`reason`	Rationale for this version.
`command_id`	Idempotency key.

Value

Documentation id, version and content hash.

`record_worker_heartbeat` *Record that a worker process is alive*

Description

Record that a worker process is alive

Usage

```
record_worker_heartbeat(repo, worker_id, started = FALSE)
```

Arguments

`repo`	Trusted worker repository.
`worker_id`	Stable name of the worker slot, for example "worker-1".
`started`	TRUE when the process has just started.

Value

Invisibly TRUE.

`redact_qualitative_source` *Create a separate redacted qualitative version*

Description

Create a separate redacted qualitative version

Usage

```
redact_qualitative_source(
  repo,
  actor,
  study_id,
  source_id,
  text,
  reason,
  command_id,
  kind = "redaction"
)
```

Arguments

`repo`	Repository.
`actor`	Actor with edit capability.
`study_id`	Study UUID.
`source_id`	Original source UUID.
`text`	Redacted text or a faithful editorial summary.
`reason`	Rationale for all omissions or reformulation.
`command_id`	Idempotency key.
`kind`	redaction or summary; summaries must not be presented as quotations.

Value

Edit UUID and exact hash, requiring independent release.

`register_invited_account` *Register a verified external account before issuing its invitation*

Description

Account onboarding is an explicit coordinator decision. The stable issuer and subject come from the institution's identity provider, never from an email address. The registered principal has no study rights until its own invitation is accepted. Existing disabled accounts are not reactivated.

Usage

```
register_invited_account(
  repo,
  actor,
  study_id,
  issuer,
  subject,
  reason,
  command_id
)
```

Arguments

`repo`	Repository.
`actor`	Study coordinator.
`study_id`	Study UUID.
`issuer`	Exact OIDC issuer URL of the qualified authentication gateway.
`subject`	Stable provider subject of the invited person.
`reason`	Documented account-binding approval; do not include credentials.
`command_id`	Idempotency key.

Value

Principal UUID for issue_panel_invitation(); no rights are granted.

`register_staff_account` *Resolve or register a staff account by its stable identity*

Description

Rights are granted to an exact issuer and subject, never to an email address. An account of a verified gateway issuer (an http or https URL) is registered if it does not exist yet; any other issuer, such as synthetic development accounts, must already be provisioned. The account receives no right by this step. A disabled account is not reactivated.

Usage

```
register_staff_account(
  repo,
  actor,
  study_id,
  issuer,
  subject,
  reason,
  command_id
)
```

Arguments

`repo`	Repository.
`actor`	Study manager.
`study_id`	Study UUID.
`issuer`	Exact issuer of the account.
`subject`	Stable subject of the account.
`reason`	Why this account is to receive rights.
`command_id`	Idempotency key.

Value

Principal UUID for set_capability().

`release_campaign` *Release an exact synthetic campaign and atomically create its outbox*

Description

The approval covers the exact text, the exact recipients and the earliest send time. Quiet hours of the protocol are fixed with the approval. A reminder is refused when any recipient would exceed the protocol's maximum number of reminders or receive it sooner than the minimum interval after the previous one; the limits never shrink an approved audience silently. Approvals of reminders for one round are decided one at a time. The worker keeps the interval as well: a reminder that would reach a person sooner after the previous one, because quiet hours or a stopped worker held both, waits until the interval has passed.

Usage

```
release_campaign(
  repo,
  actor,
  campaign_id,
  expected_hash,
  reason,
  command_id,
  not_before = NULL
)
```

Arguments

`repo`	Repository.
`actor`	Actor with coordinate capability.
`campaign_id`	Campaign UUID.
`expected_hash`	Hash reviewed in preview.
`reason`	Human approval rationale.
`command_id`	Idempotency key.
`not_before`	Earliest send time with an explicit offset, for example "2026-10-05T09:00:00+02:00"; NULL means now. At most 90 days ahead. A time that does not exist in the calendar is refused as ‘campaign.not_before‘ before any statement.

Value

Campaign id, approved recipient count and the approved earliest send time. No message is sent.

release_feedback *Release an exact reviewed feedback candidate*

Description

Release an exact reviewed feedback candidate

Usage

```
release_feedback(repo, actor, feedback_id, expected_hash, command_id)
```

Arguments

repo	Repository.
actor	Study manager.
feedback_id	Feedback UUID.
expected_hash	Exact reviewed content hash.
command_id	Idempotency key.

Value

Feedback UUID.

release_feedback_correction *Replace released feedback by a corrected version*

Description

Released feedback is never edited. A correction releases a second feedback candidate of the same analysis and records that it replaces the first: later views show the corrected version with the note for participants, while earlier display events keep referring to the version that was shown.

Usage

```
release_feedback_correction(
  repo,
  actor,
  feedback_id,
  replacement_id,
  expected_hash,
  reason,
  impact_note,
  participant_note,
  command_id
)
```

Arguments

repo	Repository.
actor	Study manager.
feedback_id	Released feedback that contains the error.
replacement_id	Reviewed candidate of the same analysis.
expected_hash	Exact hash of the replacement shown during review.
reason	Internal rationale for the correction.
impact_note	Internal assessment of the effect on ratings already given under the earlier version.
participant_note	Text shown to participants with the corrected feedback.
command_id	Idempotency key.

Value

The replacement feedback UUID and the feedback it replaces.

---

release_qualitative_edit

Release an exact qualitative version after independent review

---

Description

Release an exact qualitative version after independent review

Usage

```
release_qualitative_edit(
  repo,
  actor,
  study_id,
  edit_id,
  expected_hash,
  reason,
  command_id
)
```

Arguments

repo	Repository.
actor	Manager different from the redaction author.
study_id	Study UUID.
edit_id	Redaction UUID.
expected_hash	Exact hash shown to the reviewer.
reason	Review rationale, including representation of dissent.
command_id	Idempotency key.

Value

Immutable release UUID and content hash.

`remove_expired_artifacts` *Remove the files of exports whose download period has ended*

Description

An export is a temporary private copy of data that stays in the database. After ‘expires_at‘ it can no longer be downloaded; this function removes its directory and records the time. It never touches research data, and without ‘dry_run = FALSE‘ it removes nothing. An export whose directory is absent from this private directory is reported as not present and stays recorded as not removed: its files may lie in another installation's export directory that uses the same database. With ‘record_missing = TRUE‘ the operator confirms that they are gone, and the removal is recorded.

Usage

```
remove_expired_artifacts(
  repo,
  dry_run = TRUE,
  study_id = NULL,
  record_missing = FALSE
)
```

Arguments

`repo`	Trusted worker repository with the artifact root of the worker.
`dry_run`	TRUE lists what would be removed.
`study_id`	Optional study UUID; otherwise every study.
`record_missing`	TRUE also records exports without a directory as removed.

Value

A table of the expired exports: identifier, profile, expiry, whether files are present and whether the removal was recorded.

`remove_unfinished_exports` *Remove the directories of exports that were never registered*

Description

A worker writes an export under a temporary name, '<id>.partial', and gives it its final name only after the export is registered. A worker that was stopped while it wrote, for example by the operating system, leaves the temporary directory behind. An earlier release, which named an export before registering it, and a study export whose worker stopped just before the registration leave a complete directory under the final name, '<id>', that no registration covers. Nothing can download such directories and nothing else removes them. This function removes both kinds once they have not changed for 'older_than_hours'. Without 'dry_run = FALSE' it removes nothing. The export directory belongs to one database: a directory whose registration this database does not hold, for example after the database was restored from an older backup, is treated the same way.

Usage

```
remove_unfinished_exports(repo, dry_run = TRUE, older_than_hours = 1)
```

Arguments

`repo`	Trusted worker repository with the artifact root of the worker.
`dry_run`	TRUE lists what would be removed.
`older_than_hours`	Hours since the last change, at least one; a job that is still writing changes its directory far more often.

Value

A table of the directories: name, hours since the last change and whether it was removed.

`render_study_report` *Render a trusted packaged Quarto report without executing study text*

Description

Render a trusted packaged Quarto report without executing study text

Usage

```
render_study_report(data, output_dir, timeout = 60L)
```

Arguments

data	Report data returned by prepare_report_data().
output_dir	Existing private output directory. report.html must not exist.
timeout	Render timeout in seconds, from 1 to 300.

Value

Path of the rendered report.html; DEL_DEPENDENCY if Quarto is absent.

---

renew_job_lease *Renew the reservation of a running job*

Description

A worker process that is still working on a job keeps it reserved: the reservation is extended to ‘lease_seconds‘ from now, as long as it has not ended yet. The requester’s rights are checked again when the result is recorded, with an identity that lasts as long as the reservation.

Usage

```
renew_job_lease(repo, job_id, lease_token, lease_seconds = 600L)
```

Arguments

repo	Trusted worker repository.
job_id	Job UUID of a claim, see the ‘on_claim‘ hook of worker_step().
lease_token	Lease token of that claim.
lease_seconds	New reservation from now, between 1 and 3600 seconds.

Value

TRUE when the reservation was renewed; FALSE when it had ended or the job is no longer held by that claim. Worker infrastructure API.

reproduce_export *Verify and reproduce a portable research export without a database*

Description

Verify and reproduce a portable research export without a database

Usage

```
reproduce_export(path)
```

Arguments

path	Export directory under the user's control.

Value

Recomputed delphyr_analysis after all listed checksums match.

reproduce_study_export *Verify and reproduce a study-level research export without a database*

Description

Verify and reproduce a study-level research export without a database

Usage

```
reproduce_study_export(path)
```

Arguments

path	Extracted research_pseudonymized export under the user's control.

Value

The recomputed analysis of every round and the comparisons between rounds, after all checksums, snapshot hashes and result hashes matched.

request_analysis *Queue a durable analysis operation*

Description

Queue a durable analysis operation

Usage

```
request_analysis(repo, actor, snapshot_id, command_id)
```

Arguments

repo	Repository.
actor	Actor with analyse capability.
snapshot_id	Immutable snapshot UUID.
command_id	Idempotency key.

Value

Operation id.

request_export *Queue a private pseudonymized numeric research export*

Description

Queue a private pseudonymized numeric research export

Usage

```
request_export(repo, actor, snapshot_id, command_id, purpose = NULL)
```

Arguments

repo	Repository.
actor	Actor with export capability.
snapshot_id	Immutable snapshot UUID.
command_id	Idempotency key.
purpose	What the export is for, 10 to 500 characters. It is recorded as the rationale of the request in the audit trail and written into the manifest of the export, under ‘purposes‘. A request merged into a pending export of the same snapshot adds its purpose there.

Value

Operation id. Free-text exports require a separate reviewed profile.

`request_study_export` *Queue a private study-level export for one profile*

Description

Profiles separate what leaves the system and who may request it: research_pseudonymized (export capability) holds every frozen round with pseudonyms, decisions, lineage and reviewed qualitative records; study_summary (analyse or manage) holds aggregates only; audit_restricted (audit or manage) holds events and approvals without response content; contacts_restricted (contacts_export) holds contact fields without pseudonyms. A public release is not produced by the software.

Usage

```
request_study_export(
  repo,
  actor,
  study_id,
  profile,
  command_id,
  purpose = NULL
)
```

Arguments

`repo`	Repository.
`actor`	Actor holding the capability of the requested profile.
`study_id`	Study UUID.
`profile`	research_pseudonymized, study_summary, audit_restricted or contacts_restricted.
`command_id`	Idempotency key.
`purpose`	What the export is for, 10 to 500 characters; see request_export().

Value

Operation id and state.

`resolve_delivery` *Resolve an uncertain delivery by a documented human decision*

Description

An uncertain delivery is never repeated automatically, because the message may already have been delivered. A coordinator decides with a rationale: ‘confirmed_delivered‘ records that delivery was verified; ‘requeue‘ sends it again and accepts a possible duplicate; ‘abandon‘ stops it. Before a requeued message is sent, eligibility is checked again.

Usage

```
resolve_delivery(
  repo,
  actor,
  study_id,
  message_id,
  resolution,
  reason,
  command_id
)
```

Arguments

`repo`	Repository.
`actor`	Actor with coordinate capability.
`study_id`	Study UUID.
`message_id`	Message UUID from list_uncertain_deliveries().
`resolution`	confirmed_delivered, requeue or abandon.
`reason`	Rationale, including how delivery was or was not verified.
`command_id`	Idempotency key.

Value

Message UUID and its new delivery state.

revoke_panel_invitation *Revoke an unconsumed synthetic invitation*

Description

Revoke an unconsumed synthetic invitation

Usage

```
revoke_panel_invitation(
  repo,
  actor,
  study_id,
  invitation_id,
  reason,
  command_id
)
```

Arguments

repo	Repository.
actor	Study coordinator.
study_id	Study UUID.
invitation_id	Invitation UUID.
reason	Documented account-binding approval; do not include credentials.
command_id	Idempotency key.

Value

Invitation UUID.

run_analysis *Analyse and persist a frozen snapshot*

Description

Analyse and persist a frozen snapshot

Usage

```
run_analysis(repo, actor, snapshot_id, command_id)
```

Arguments

repo	Repository.
actor	Actor with analyse capability.
snapshot_id	Snapshot UUID.
command_id	Idempotency key.

Value

Analysis UUID and scientific result hash.

---

save_response *Save a response with optimistic concurrency and a durable receipt*

Description

Save a response with optimistic concurrency and a durable receipt

Usage

```
save_response(
  repo,
  actor,
  enrollment_id,
  round_item_id,
  response,
  expected_revision,
  command_id
)
```

Arguments

repo	Repository.
actor	Panel actor.
enrollment_id	Own enrollment UUID.
round_item_id	Assigned response field UUID.
response	List with value and status.
expected_revision	Last confirmed revision; zero for first save.
command_id	Idempotency key, reused only for identical retries.

Value

Committed revision id, revision and server time.

`set_capability` *Assign or revoke a study capability*

Description

Assign or revoke a study capability

Usage

```
set_capability(
  repo,
  actor,
  study_id,
  principal_id,
  capability,
  enabled,
  command_id,
  reason = NULL
)
```

Arguments

`repo`	Repository.
`actor`	Study manager.
`study_id`	Study UUID.
`principal_id`	Principal UUID.
`capability`	One of manage, analyse, export, coordinate, edit, panel, audit or contacts_export. The contact export is never implied by a role.
`enabled`	Grant if TRUE; revoke otherwise.
`command_id`	Idempotency key.
`reason`	Optional rationale, recorded in the audit trail.

Value

Membership id. The last management right of a study cannot be revoked: a study always keeps one person who can manage it.

set_panel_group *Change stakeholder group only for future round preparation*

Description

Change stakeholder group only for future round preparation

Usage

```
set_panel_group(
  repo,
  actor,
  study_id,
  panelist_id,
  group_code,
  reason,
  command_id
)
```

Arguments

repo	Repository.
actor	Study manager.
study_id	Study UUID.
panelist_id	Study-specific pseudonym UUID.
group_code	Group declared by the current protocol.
reason	Nonempty rationale.
command_id	Idempotency key.

Value

Immutable group-change receipt; existing enrollments stay unchanged.

submit_round *Submit the exact confirmed revision set atomically*

Description

Submit the exact confirmed revision set atomically

Usage

```
submit_round(repo, actor, enrollment_id, expected_revision_set, command_id)
```

Arguments

repo	Repository.
actor	Panel actor.
enrollment_id	Own enrollment UUID.
expected_revision_set	Named integer vector: round-item UUID to revision.
command_id	Idempotency key.

Value

Durable submission receipt id and timestamp.

---

transition_round *Advance an authorized round through its approval lifecycle*

Description

Advance an authorized round through its approval lifecycle

Usage

```
transition_round(
  repo,
  actor,
  round_id,
  target,
  expected_hash,
  reason,
  command_id
)
```

Arguments

repo	Repository.
actor	Study manager.
round_id	Round UUID.
target	review, approved, open, closed, finalized or cancelled. Approval and opening are refused with DEL_VALIDATION while get_round_readiness() reports blocking findings. Only a round that was never opened can be cancelled; a cancelled round is final and frees its round number.
expected_hash	Exact instrument hash shown in review.
reason	Nonempty human rationale.
command_id	Idempotency key.

Value

Round id and new state.

validate_feedback *Validate a feedback draft and its immutable hash*

Description

Validate a feedback draft and its immutable hash

Usage

```
validate_feedback(draft)
```

Arguments

draft	Output of prepare_feedback().

Value

TRUE or DEL_VALIDATION.

Examples

```
validate_feedback(prepare_feedback(analyse_round(demo_snapshot())))
```

validate_items *Validate an instrument import before any database writes*

Description

Validate an instrument import before any database writes

Usage

```
validate_items(items, protocol)
```

Arguments

items	Data frame with the columns that read_item_import() returns, including item_version and source_ref. Codes, locales, texts and references must be text, as read_item_import() returns them; a code read as a number is refused.
protocol	Study protocol.

Value

TRUE or a structured validation condition.

validate_protocol *Validate a declarative study protocol*

Description

Validate a declarative study protocol

Usage

```
validate_protocol(protocol)
```

Arguments

protocol	Configuration list.

Value

delphyr_validation with valid, issues, and schema_version.

Examples

```
validate_protocol(demo_protocol())
```

validate_response *Validate and normalize an answer*

Description

Validate and normalize an answer

Usage

```
validate_response(value, status, scale)
```

Arguments

value	Scalar integer or text, or NULL for missing responses.
status	Answer status.
scale	A scale created by new_scale().

Value

List with status, value_int and value_text; DEL_VALIDATION on error.

Examples

```
validate_response(7, "answered", new_scale("ordinal_integer", 1:9))
```

`withdraw_participation` *Withdraw the current participant from future synthetic study activity*

Description

Stops further collection and delivery eligibility while retaining previously committed research data. This explicit synthetic policy is not a deletion workflow or a policy for use with real participants.

Usage

```
withdraw_participation(repo, actor, study_id, retention_policy, command_id)
```

Arguments

`repo`	Repository.
`actor`	Participant's own current server identity.
`study_id`	Study UUID.
`retention_policy`	Must explicitly be synthetic_retain_prior_data.
`command_id`	Idempotency key.

Value

Withdrawal event id and policy receipt.

`worker_step` *Process one leased job using its requester's current study rights*

Description

The requester's identity is delegated for exactly as long as this worker holds the job's reservation, as the database measures it: renewing the reservation (renew_job_lease()) extends it, and it ends when the reservation ends or another worker takes the job up. A refusal that a retry cannot change (‘DEL_FORBIDDEN‘, ‘DEL_UNAUTHORIZED‘, ‘DEL_NOT_FOUND‘, ‘DEL_VALIDATION‘) ends the job; any other failure, including a report that could not be rendered (‘DEL_RENDER‘) and a reservation that ended before the result was recorded (‘DEL_CONFLICT‘ with path ‘job.lease‘), is retried after ten seconds, up to three attempts. A rights check refused because the reservation had ended counts as such a lost reservation (‘DEL_CONFLICT‘), not as a refusal of the requester. An export is written under a temporary name and gets its final name only after it is registered; one that cannot be registered is removed, whatever failed.

Usage

```
worker_step(
  repo,
  study_id = NULL,
  lease_seconds = 600L,
  download_hours = 24,
  on_claim = NULL
)
```

Arguments

repo	Trusted development/test worker repository.
study_id	Optional study UUID limiting a worker partition.
lease_seconds	Time the job is reserved for this worker. A job whose worker stopped is taken up again after that time; a job that needs longer cannot record its result unless its reservation is renewed. The default leaves a wide margin over the analysis and export of a round of 300 members and 300 rating fields.
download_hours	Hours an export can be downloaded after it was registered, from 1 to 720; one day by default. The period is the decision of the responsible organisation.
on_claim	Optional function called with the claimed job (a one-row data frame with 'id' and 'lease_token') before it runs, for example to renew its reservation from another process.

Value

FALSE when idle, otherwise a safe operation state.

---

write_participant_feedback

Write a panel member's own feedback for download

Description

The participant-feedback profile: the released aggregate of the previous round and only this person's own previous answers.

Usage

```
write_participant_feedback(repo, actor, enrollment_id, output_dir)
```

Arguments

repo	Repository.
actor	Panel actor.
enrollment_id	Own enrollment UUID with assigned released feedback.
output_dir	Existing private directory controlled by trusted server code.

Value

Written file names; DEL_NOT_FOUND when no feedback is assigned.

`write_report_data` *Write allowlisted numeric report supplements to a private staging directory*

Description

Write allowlisted numeric report supplements to a private staging directory

Usage

```
write_report_data(data, output_dir)
```

Arguments

`data`	Report data returned by prepare_report_data().
`output_dir`	Existing private staging directory under caller control.

Value

Written basenames, to include in the enclosing export manifest.

Index

accept_panel_invitation, 4
add_panelist, 5
amend_protocol, 6
analyse_round, 7
assign_feedback, 7
authenticated_actor, 8

cancel_campaign, 9
change_round_deadline, 9
claim_job, 10
classify_consensus, 11
code_qualitative_source, 11
compare_rounds, 12
complete_study, 13
connect_repository, 13
content_hash, 14
create_feedback, 14
create_qualitative_theme, 15
create_study, 16

del_abort, 16
demo_actor, 17
demo_protocol, 17
demo_snapshot, 18
demo_study, 18
download_artifact, 19

enroll_panel, 19

freeze_round, 20

get_account_rights, 21
get_analysis, 21
get_capabilities, 22
get_feedback, 22
get_feedback_candidate, 23
get_item_comparability, 23
get_operation, 24
get_operations_status, 24
get_panel_import_receipt, 25
get_qualitative_provenance, 25
get_questionnaire, 26
get_retention_report, 26
get_round_instrument, 27
get_round_readiness, 27
get_snapshot, 28
get_study_documentation, 28
get_study_setup, 29
get_system_status, 29

hold_pending_messages, 30

import_panel, 31
import_round_contributions, 31
invitation_code, 32
issue_panel_invitation, 32

link_item_source, 33
list_audit_events, 34
list_campaign_enrollments, 35
list_campaign_rounds, 36
list_contribution_rounds, 36
list_enrollments, 37
list_item_decisions, 37
list_panel, 38
list_panel_invitations, 38
list_protocol_versions, 39
list_qualitative_reviews, 39
list_released_edits, 40
list_released_feedback, 40
list_rounds, 41
list_studies, 41
list_study_staff, 42
list_uncertain_deliveries, 42
log_event, 43
log_operation, 44

migrate_repository, 44

new_authentication_config, 45
new_correlation_id, 46
new_invitation_token, 46

new_message_adapter, 47
new_protocol, 47
new_scale, 48
new_snapshot, 48

parse_invitation_code, 49
prepare_campaign, 50
prepare_feedback, 51
prepare_report_data, 51
prepare_round, 52
prepare_study_report_data, 53
preview_campaign, 53
preview_panel_import, 54
preview_panel_invitation, 54
process_campaign_message, 55
process_campaign_sink, 56
provision_demo_principal, 56
publish_consent, 57

read_item_import, 57
record_consent, 58
record_item_comparability, 59
record_item_decision, 60
record_item_lineage, 60
record_qualitative_source, 61
record_study_documentation, 62
record_worker_heartbeat, 63
redact_qualitative_source, 64
register_invited_account, 65
register_staff_account, 66
release_campaign, 67
release_feedback, 68
release_feedback_correction, 68
release_qualitative_edit, 69
remove_expired_artifacts, 70
remove_unfinished_exports, 71
render_study_report, 71
renew_job_lease, 72
reproduce_export, 73
reproduce_study_export, 73
request_analysis, 74
request_export, 74
request_study_export, 75
resolve_delivery, 76
revoke_panel_invitation, 77
run_analysis, 77

save_response, 78
set_capability, 79
set_panel_group, 80
submit_round, 80

transition_round, 81

validate_feedback, 82
validate_items, 82
validate_protocol, 83
validate_response, 83

withdraw_participation, 84
worker_step, 84
write_participant_feedback, 85
write_report_data, 86