PRACTICECOACHVERIFIED 2026-09-03 · ENSEMBLE SHARING RECORDED 2026-09-07 · DEVICE STORAGE 2026-10-09 · ERASURE CORRECTED 2026-10-12 · OWNER BLOCK 2026-10-31 · FCM AND REGIONS 2026-11-06 · PLANS AND PAYMENTS 2026-11-07 · TRIAL CLAIMS AND WEBHOOK LOG 2026-11-08 · FLAG COUNTS 2026-11-10

FOR YOUR DPO / DATA-PROTECTION LEAD

DPIA Information Pack

Information to support your Data Protection Impact Assessment — the school remains controller

DPIA Information Pack

Information to support *your* DPIA — the school remains the data controller and owns the Article 35 assessment. Prepared July 2026 against ICO guidance. This is not a completed vendor DPIA, and it is not legal advice.

What this is (and is not)

This pack gives your Data Protection Officer / data-protection lead the **data-flow facts** they need to complete the school's **own** Data Protection Impact Assessment for deploying PracticeCoach.

- **It is not a "completed DPIA."** Under UK GDPR **Article 35**, the DPIA duty sits with the **controller** — the school. A processor cannot discharge that duty on your behalf. This pack *feeds* your assessment; it does not replace it.
- **Why the shape of this pack.** It is deliberately **roles-explicit**, lists **named sub-processors** with region and transfer safeguard, and **maps the data flow**. That is a direct response to the ICO's June 2026 *Edtech examined* report, whose audits of edtech providers found exactly three recurring failures: controller/processor **role misclassification**, **insufficiently detailed contracts**, and **incomplete data-flow mapping**. This pack is built to let your DPIA avoid all three.
- **Roles.** The **school is the controller**; **PracticeCoach is the processor** (see the companion DPA / processor-terms template, which covers the eight Art. 28(3) clauses).
- **Model.** The shape follows real UK edtech vendor packs that feed a school-completed DPIA (e.g. school-published DPIAs for mainstream edtech tools).

Because children's practice data is involved and PracticeCoach is also offered direct-to-consumer, this processing is the kind the ICO treats as **higher-risk**, so a DPIA is expected — another reason this pack exists.

---

1. Purposes of processing

PracticeCoach is processed under the school deployment to:

1. Let pupils **plan and log** their instrumental/vocal practice (pieces, passages, sessions, confidence ratings, tempo progress).
2. Build **organisational practice plans** — the app schedules **what** to practise from the pupil's own data. **Plans are built by fixed rules in code, not by an AI model** (corrected 16 October 2026: the last AI plan path was retired). It is explicitly an **organiser, not an instructor**: it never gives technique or teaching advice. AI is used only for the assistant (Ask, the chat), for wording insights, session debriefs and the "where you left off" summary, for wording a single-piece mini-plan, for reading the first page of an imported score, and for voice transcription — see the OpenAI row in §5 and the public page `/how-we-use-ai`.
3. Let **teachers** in the department support pupils (see progress, set assignments, share resources) under an explicit consent object.
4. Where a guardian is linked, give **guardians** appropriate visibility of a minor child's activity.
5. Optionally, where the Head of Department has **knowingly enabled** it, support **teacher↔pupil direct messaging** (off by default for school studios; message history is retained, append-only, and disclosable/exportable to the school).

6. **Ensembles — sharing music within a consented group.** *(Recorded as a purpose on 7 September 2026, discharging §8 of this pack; the surface itself is still switched off — see the first bullet.)* An ensemble is a small private group (currently capped at 40 people) that one accountable owner creates and other people join by an unguessable link. It gives a pupil a way to share sheet music they uploaded with the people they play with, which is the first channel in the product where content reaches a pupil from someone who is **not their own teacher**. What the school's DPIA needs to know about it:
    - **It is off.** The feature ships behind a runtime kill switch that is false and has never been written; nothing described here is reachable by any user until an administrator deliberately enables it, which is a separate and recorded act — the row records who enabled it and when.
    - **Personal data disclosed to other members:** display name, role within the ensemble (owner / leader / member), and join date. **Never** an age, an age band, a date of birth or an email address, in any view, to anyone. Where an ensemble contains anyone under 18, non-owner members see only the owner, any leaders and a total count.
    - **Under-13 accounts cannot create or join one**, enforced server-side from the stored date of birth, not from anything the client sends.
    - **No group messaging.** An owner can post a one-way notice; there is no reply, comment, reaction or thread anywhere in the feature. A notice addressed to a member under 18 is delivered to that member's accepted guardian by the existing notification routing, not to the young person's own channels.
    - **Not public and not findable.** There is no public page, no directory and no link that resolves without signing in; joining requires a live, revocable, expiring code.
    - **How the music moves: a copy per recipient, not a shared reading of one file.** When a member takes a score — a part, or a full score, which they may keep for reference outside the planner — that has been shared with the ensemble, the pages, the passages and the rights attestation are **copied into that member's own account**. Nobody reads out of the sharer's library, and nothing is served from it. Where the sharer chose to include what they had written on the score, their markings arrive as a **frozen layer**: the recipient can show, dim or hide it and cannot edit or erase it, and the recipient's own annotations are a separate layer that survives every later update. For the school's DPIA the consequence is the one that matters — the copy is the recipient's personal data in the recipient's account, and leaving the ensemble does not retract copies already taken.

- **Guardian visibility is preserved *where a guardian link exists*:** a guardian of a minor member sees, in the existing guardian view, every piece that member imported from an ensemble, with the sharer, the ensemble and the rights basis claimed. **The condition matters for a school's DPIA and is stated rather than assumed (corrected 4 September 2026):** the guardian view requires an accepted guardian link with the young person's own consent, sign-up blocks under-13s only, and — **stated in the past tense on 7 September 2026, because it is no longer true; see the correction at the end of this bullet** — nothing *then* required a 13–17 account to have a guardian linked before joining an ensemble, so a self-registered teenager could have been in an ensemble with no guardian seeing what arrived. In a school deployment the school's own consent and safeguarding route is the counterpart control; for a personal account it is not. **Corrected 7 September 2026, because the sentence that stood here is now false and a DPO would have relied on it:** this bullet used to end *"Requiring or prompting a guardian link at join is assigned to Phase 84 in the assessment's rating-change table and must land before the feature is switched on"*. Phase 84 landed it, so the condition is **enforced at the join rather than hoped for**: an account aged 13–17 with no live, accepted, consented guardian link is refused, and an account whose age band cannot be read at all is refused and asked for a date of birth rather than admitted. A narrowly scoped administrator override exists for the case a school needs one — one account, one ensemble, a mandatory reason, the grantor recorded, and both the grant and any later revocation written to the moderation log. And the guardian is now **told** rather than merely able to look: a notice goes to the linked guardian when the account joins an ensemble, when a part is copied into their library, when an update to one is accepted, and when an ensemble date reaches their calendar.
- **The Online Safety Act assessment for this change has been re-run.** `docs/compliance/online-safety-risk-assessment.md` was re-assessed on **3 September 2026** against ensemble sharing as built — a pre-change assessment under its own trigger list, not the annual review, which remains due September 2027. Its outcome: every children's risk rating stays Low or Negligible and no recommended safety measure changed status; one Ofcom risk factor changed answer (users can now form closed groups) and is recorded as such rather than argued away; and the duty to report CSEA content to the National Crime Agency (s.66 Online Safety Act 2023, SI 2026/268) was determined to apply and is now operationalised in `content-moderation-runbook.md` §5. §6A of the assessment holds the dated pre-change design decisions, made before any ensemble table existed. **Still pending:** the transcription session in which both Ofcom toolkits are re-run under the existing reference codes and the sign-off at §7 is dated and signed; until then the re-answers are filed as "pending toolkit confirmation", and the feature stays switched off.

7. **Ensemble readiness — showing a consenting member's progress on that ensemble's music to the people they play with.** *(Recorded as a purpose on 15 September 2026, when the v4.1 readiness work was built; like the ensemble surface itself, it is still switched off behind its own separate kill switch — see below.)* This is a **second, narrower sharing purpose inside the first**, and it is worth stating separately because it discloses a different kind of data to a different class of reader. What the school's DPIA needs to know about it:
    - **What is disclosed:** for each piece of music shared with the ensemble, a consenting member's **confidence and tempo** — the confidence rating (1–5) and the tempo they are currently working at, on the piece and on each of its passages — and a derived on-track / not-on-track verdict. Nothing else — not practice history, not session durations, not recordings, not session notes, not plans, and nothing from the rest of that member's repertoire.
    - **It is opt-in, per ensemble, and revocable.** There is no row in `ensemble_readiness_consents` until the member deliberately turns sharing on for that one ensemble, and the projection returns nothing about a member without one. Turning it off stops the disclosure immediately. Consent is held by the member themselves, including members aged 13–17 (see §7).
    - **It is flag-gated independently of ensembles.** A second runtime kill switch (`ensemble_readiness`) governs this surface alone: an administrator can switch readiness off and leave ensembles running. Like the ensembles flag, it is false, and enabling it is a deliberate, recorded act.
    - **Who reads it is decided by the ensemble's age composition, not by preference.** In an all-adult ensemble, the other members read it (or, where the sharer chose *Leaders only*, just the owner and leaders). Where the ensemble contains **anyone under 18**, only the **owner and adult leaders** read other members' rows and every other member reads **only their own** — the same age switch that governs the roster. A member always sees their own readings in either mode.
    - **It is a projection, not a feed.** The view is computed on request from the member's existing practice data; it carries no timestamps of practice, no streaks and no recency signal, so a reader cannot reconstruct when or how much another member practised. The closed column list in §2 is what makes that a bounded claim rather than an intention.
    - **A linked guardian of a member under 18 is told on both transitions** — when the young person turns readiness sharing on, and when they turn it off. See §7.

No processing for advertising. No sale of personal data. No profiling beyond the practice-scheduling logic described above.

---

2. Data categories

Category	Examples	Special category?
Account / identity	Name or display name, email, age-band (teacher views use age-bands, not date of birth), role (pupil / teacher / guardian), studio membership	No
Children's practice data	Repertoire (pieces), user-defined passages, practice sessions and durations, confidence ratings (1–5, often at a working tempo), tempo history, practice plans, run-throughs, flagged problem spots, whether a piece is played from memory (and, per run-through, whether it was played from memory)	No
Teacher-authored records	Assignments, progress-report notes, annotations on a pupil's record	No
Messages (only if DMs enabled)	Teacher↔pupil direct-message threads and message history	No
Ensemble membership (only once ensembles are enabled)	Ensemble name, the member's role in it (owner / leader / member), join date, and the display names of co-members visible to them; one-way notices posted by the owner. No age, age band, date of birth or email is disclosed to any member.	No
Rights attestation (only once ensembles are enabled)	Recorded against the **sharer**, not against the reader: which of the three declared bases they chose (`rights-bases-decision.md` §1 — public domain, own work, or permission/licence), the source or licence reference and any note they typed, the version of the wording they were shown, who attested and when — one attestation per score, under the wording version current at the share (`2026-10-06` since the Phase 99 fixes, `2026-10-01` from v4.2). A snapshot of the same fields is frozen onto every copy taken, so a copy can be traced to the claim that justified it.	No
Ensemble readiness (only once ensembles *and* readiness are enabled, and only for a member who has consented)	Per shared piece and per passage: the member's confidence rating (1–5), current tempo, whether they have added the part, and a derived on-track verdict — plus the consent record itself (`ensemble_readiness_consents` : the ensemble, the member, `accepted` / `revoked` , `consented_at` , `revoked_at`). No practice history, durations, session notes, recordings or plans.	No

Category	Examples	Special category?
Technical / operational	Auth session data, device push-subscription tokens (if the pupil opts into reminders), error diagnostics; the device's time-zone name (e.g. `Europe/London`): detected on the device and stored as `user_profiles.timezone` for streaks and reminder quiet hours (long-standing, first recorded here 20 October 2026), and also sent with a plan request from release 3.19 so "today" and day counts in the plan follow the user's local day. That request copy is not stored separately and is never sent to OpenAI	No

Added 29 October 2026 (Phase 136): the memory mark. Whether a piece is played from memory (`pieces.played_from_memory`), when the "performance ready" suggestion was last offered (`pieces.performance_ready_offered_at`) and the per-run-through memory answer (`practice_run_throughs.from_memory`) are low-sensitivity practice data. A linked teacher, guardian or department reads them through the existing piece read access and no wider; the ensemble readiness allowlist never carries them, so a "from memory" mark is not disclosed to the people a member plays with.

Added 10 November 2026: flag counts on the teacher and department screens. A flag is a problem spot a pupil marks on their own music (the "flagged problem spots" in the table above). Two screens now show counts of them. On a pupil's page, a linked teacher's *Since last lesson* card shows how many flags the pupil **found, fixed and had come back** since the last lesson (or in the last 7 days, where no lesson is logged): a total, then a row per passage, with a flag outside any passage counted under its piece. The Head of Department's page shows one *Found and fixed* figure for the **whole department** for the last 7 days (flags found, fixed and come back, and how many pupils fixed at least one) beside time practised; it carries no per-teacher or per-pupil figure and no ranking. Both carry **counts and passage (or piece) names only: never the note a pupil wrote on a flag, and never its position on the score.** A teacher cannot fix, reopen or remove a pupil's flag. **No new data is collected and no recipient is added.** The counts are derived from timestamps already held on each flag (when it was placed, fixed and reopened), and they reach a teacher or the department only through the existing consent objects: an accepted teacher link that shares practice data, or the pupil's separate department consent (`is_linked_teacher` , `is_department_teacher` , `resolveDepartmentStudentIds`). **How the limit is applied, stated rather than assumed:** both screens are served by one server read (`readFlagCountRows` in `api/_lib/teacher/flagCountReads.ts`), which selects only the columns named in `FLAG_COUNT_COLUMNS` (`src/lib/flagCounts.ts`): the flag's id, the pupil, the piece, the passage and the three timestamps. It is not a column-level restriction in the database: the row-level read policies that have let a linked or department teacher read a consenting pupil's flags since migrations 075 and 115 (restated in 228) cover the whole row, so the note and the position are withheld by what the service reads and shows, not by the database itself. The rules every such screen follows are in `docs/FIND_FIX_PROGRESS.md` .

Additional recipient class, once ensembles are enabled (recorded 7 September 2026). Until now every recipient of a pupil's data was an adult the pupil or the school had linked — a teacher, a guardian, the department. Ensembles add one more: **other members of a group the data subject has joined**, who receive the member's display name, role and join date, and the content that member chooses to share, and nothing else. Where the member is under 18 their linked guardian is a recipient too, by design rather than by accident — see §7. No member ever receives another member's age, age band, date of birth or email address, in any view.

The readiness disclosure is a CLOSED column list, and it is enforced rather than promised (recorded 15 September 2026). The wire format of the readiness view is bounded to **30 keys** across every level of its response — the envelope, the next event, the work, the work's passages, the member column and the member cell. The list is the exported constant `ALLOWED_READINESS_KEYS` (with its counterpart `BANNED_READINESS_KEYS`), declared in `api/_lib/ensemble/readinessEnvelopeKeys.ts` and copied verbatim into `online-safety-risk-assessment.md` §6B so this pack and that assessment quote one list rather than two; a key added to the response without being filed turns a test red in the same commit. **Added 12 October 2026 (Phase 105):** from migration 211 the reader's own row additionally carries `own_part_id`, saying which of their own scores was read for them; it is filed separately, in `api/_lib/ensemble/readinessOwnRowKeys.ts` and as its own list in §6B of that assessment, it appears on no other member's row, and it is served to no other reader — so the 30-key bound is unchanged for everything a reader is served about anyone else. **Added 13 October 2026 (105-REVIEW WR-03), on which score those keys are read from:** from the same migration a consenting member's row is also read from a score they attached to a listed work themselves, where before only a copy they had received was read. An attached score is read from the attacher's own piece and is not subject to a share-liveness test, so it is still read where the share of that score has been withdrawn, was never made, or is suspended by moderation or by the age rule, for as long as the work itself stays listed; a received copy in those cases is not read. What is shown is the attacher's own status, confidence and tempo, to the readers their consent names, and never the withheld material. No key and no reader is added by this. Whether the two suspended cases should be read is an open ruling for the owner, recorded in the change log of `online-safety-risk-assessment.md`. The module is the shared home deliberately, and a DPO following this citation should land there rather than in a test file: `api/__tests__/ensembleReadinessRoutes.test.ts` and `api/__tests__/readinessFilingParity.test.ts` both IMPORT the arrays from it, and a test file cannot be the source a second test file audits against without the audit becoming circular. (90-REVIEW WR-02: this paragraph cited the route suite, which only re-exports them.) `BANNED_READINESS_KEYS` **names seventeen keys, and naming them is the stronger half of the control (corrected 15 September 2026, 90-REVIEW WR-10).** Three are **withheld by design** because they would otherwise look harmless: `display_name` in self mode (which would name a person to a reader who is only entitled to their own row), and `member_count` and `minor_present`, which are disclosure-by-shape — they would tell a reader something about the roster's age composition that the age switch exists to withhold. Four are the age vocabulary itself (`minor`, `age`, `band`, `date_of_birth`), which may not reach this surface in any form. The remaining ten are named **because each is a real column on a table this projection reads**, and therefore one careless `select *` away: `tempo_history`, `sessions`, `minutes`, `run_through`, `updated_at`, `confidence_log`, `fragility` and `lastPracticed` are every practice timestamp and recency signal the schema holds, and `annotation` and `notes` are what a member wrote. So **practice timestamps, streaks and recency signals are not merely absent from the allowed list — they are on the banned one, by name** (the earlier wording here said they were "outside the list", which understated

the control): a reader who could watch them could reconstruct another member's practice diary from a view meant to carry only progress, and the test that would catch it fails on the key's NAME rather than on the allowlist's silence.

No special-category data (UK GDPR Art. 9) is intentionally collected — no health, ethnicity, religion, biometric or similar data is a required field. Practice data is about musical repertoire and progress, not the pupil's characteristics. If a pupil freely types special-category content into a free-text field (e.g. a message), that is incidental and not a designed data category; the school's DPIA should note this residual free-text risk.

---

3. Data subjects

- **Pupils (13+).** PracticeCoach is 13+; under-13 / KS2 is out of scope. Pupils are children, so the **Children's Code** applies.
- **Teachers** in the department (Head of Department and colleagues).
- **Guardians**, where a guardian is linked to a minor child.

---

4. Processing locations and the data flow

- **Primary datastore is EU-resident.** The Supabase project backing this deployment runs in region `eu-west-1` **(AWS Ireland, EU)** — confirmed 2026-07-10 and confirmed again 2026-11-06 (Supabase CLI). So pupil records in the primary database are **not** transferred to the US for storage. **Re-confirm the exact project region in the Supabase dashboard at signing.**
- Application hosting and serverless functions run on Vercel; functions run in Dublin (`dub1`), and static files are served from Vercel's global edge network.
- The AI calls go to the **currently-active AI providers** named in the sub-processor table: the assistant to **Anthropic**; wording, score-page reading and transcription to **OpenAI**. Plan building makes no AI call.
- **Native push** (the Android and iPhone apps, if a pupil turns notifications on) is sent through **Google Firebase Cloud Messaging**, which PracticeCoach engages with its own service account (`api/_lib/sendFcm.ts`), so it is a sub-processor (§5). Each message carries the device token, the notification title and body, and an in-app link. On iPhone, FCM hands the message to Apple Push Notification service for delivery.

- **Web push** (the browser/PWA, if a pupil opts in) transits the browsers' push gateways (Apple / Google / Mozilla) — a transport channel, not a data store (see footnote).
- **Optional Google sign-in.** A pupil or teacher may create or sign in to their account with Google instead of a password. Google acts as an identity provider in its own right (an independent controller for the Google account, not a PracticeCoach sub-processor): it confirms the user's identity and returns their name, email address and a Google account identifier, which Supabase Auth stores with the account. No practice data is sent to Google, and no Google data beyond those three items is requested (scopes `openid email profile`). Under-13 accounts cannot be created by any route, Google included: the date-of-birth gate comes before the Google button.
- **Optional Sign in with Apple** (added 3 November 2026). The same, with Apple as the identity provider (again an independent controller for the Apple Account, not a sub-processor). Apple returns an account identifier, an email address — the user's own, or a private relay address if they choose Hide My Email — and, on the first sign-in only and only if the user shares it, their name. Email to a relay address passes through Apple's private relay service, for which PracticeCoach's sending domain is registered with Apple. One further item is kept: an Apple refresh token, stored server-side only (no client can read it), whose only use is to revoke the account's Apple sign-in when the account is erased, as Apple requires. It is deleted with the account. The date-of-birth gate precedes the Apple button exactly as it precedes Google's.

Data-flow summary: pupil's browser/PWA → Vercel (app + functions) → Supabase (`eu-west-1`, primary store, row-level-security-scoped per user) → AI provider (only when the pupil uses an AI feature, and only the context that feature needs — never for building the plan) → back to the pupil. Teacher and guardian access to a pupil's data is gated by explicit consent objects, not open to the whole school.

---

5. Named sub-processors

Each sub-processor is engaged under the Art. 28 sub-processor clause of the DPA, with the **advance change-notification** mechanism described there. **Confirm every region and safeguard live at signing** — they can drift. Until each US provider's transfer mechanism is confirmed and recorded here (as PostHog's is), the public privacy policy describes the safeguards only in general terms ("for example, the UK–US data bridge where a provider is certified under it, or the UK's approved contract clauses") and names no provider as certified.

Sub-processor	Purpose	Data categories	Region	Transfer safeguard
Supabase	Primary datastore (Postgres), auth, storage — the system of record	Account/identity, children's practice data, teacher records, messages	`eu-west-1` — **AWS Ireland (EU-resident)**, confirmed 2026-07-10	EU-resident; no US transfer for the DB itself. Re-confirm region at signing.
Vercel	Application hosting + serverless functions (request processing in transit)	All categories, in transit during request handling	Functions `dub1` (Dublin, EU); static assets via Vercel's global edge	Standard controller-to-processor / international transfer terms per Vercel DPA; confirm at signing
Anthropic	**Currently-active** AI provider for the chat assistant (Ask), using Claude Haiku 5.5 since November 2026. **Not used to build practice plans.**	The practice-data context the chat needs, and the screen the user is on (its route) with every message and, when the screen chip names a piece, passage or event, its name and a few details about it such as its learning stage or the event's date (resolved server-side from ids, user-scoped; never a screenshot), with ✕ on the chip stopping both. **Never practice recordings, never audio.** (No advertising use.)	United States or global processing (Anthropic offers no EU-only processing for the API) — confirm at signing	Anthropic's Commercial Terms bar training on customer content; API inputs and outputs are deleted within 30 days unless flagged under its Usage Policy or required by law. Standard DPA and international transfer terms; confirm at signing. Swapped in through the DPA's Art. 28 change-notification clause.

Sub-processor	Purpose	Data categories	Region	Transfer safeguard
OpenAI	**Currently-active** AI provider for wording insights / session debriefs / the "where you left off" notes summary / a single-piece mini-plan, voice transcription, and reading title/composer details from an imported score. **Not used to build practice plans** (built by code since October 2026).	The practice-data context the feature in use needs (for the notes summary, the pupil's own practice notes on that piece); short voice clips for transcription (not retained), sent with the user's own piece titles, composers and passage names as spelling hints (where the device supports it, speech detection runs on the device: if it hears no speech within 8 seconds, the recording stops and nothing is sent); the **first page of an uploaded score** (as an image or its text layer) at import time, used only to read the title, composer and similar details and not retained by the provider. **Never practice recordings.** (No advertising use.)	Provider infrastructure — confirm at signing	The AI seam is **provider-swappable**; any change runs through the DPA's Art. 28 change-notification clause. Confirm transfer terms at signing.
Email provider	Transactional email (invites, notifications)	Account/identity (email), message content only insofar as an email notifies of it	**Zoho ZeptoMail, EU data centre** (`api.zeptomail.eu`), since September 2026	EU-resident processing; Zoho's standard DPA and transfer terms — confirm at signing
Sentry	Error monitoring / diagnostics	Technical/operational diagnostics; not a designed store of practice data	EU (Germany), `ingest.de.sentry.io`	Standard DPA + international transfer terms; confirm at signing

Sub-processor	Purpose	Data categories	Region	Transfer safeguard
Google Firebase Cloud Messaging	Push notifications to the Android and iPhone apps (on iPhone via Apple Push Notification service)	Device token; notification title, body and in-app link	Google global infrastructure	Google's Firebase data processing terms; transfer mechanism to confirm at signing

Footnote — web-push transit. If a pupil opts into reminders in a browser, notifications transit the **Apple / Google / Mozilla push gateways**. These are a transport channel for the push token and message payload, not a store of the pupil's practice record.

Footnote — session recordings (PostHog), not engaged for schools. Since 28 October 2026 an adult (18+, confirmed by date of birth) who uses PracticeCoach **on their own** may opt in to session recordings, processed by **PostHog** in its **EU cloud (Frankfurt)** and deleted after 30 days. The app never asks, and never records, an account that is under 18, of unknown age, linked to any teacher as a pupil, a guardian, or a member of a school department (`src/lib/sessionReplay.ts`, `isReplayEligible`). **No school's pupil or staff data therefore flows to PostHog**, and it is not a sub-processor for a school deployment. PostHog processes those adults' recordings under a DPA signed 28 October 2026 (`posthog-dpa-2026-10-28.pdf`, PandaDoc ref. KJHUJ-9MG7B-DK6SE-TYMM2), with UK transfers covered by the UK Extension to the EU-US Data Privacy Framework (clause 10.3) and the UK Addendum to the Standard Contractual Clauses as the fallback (clause 10.4). The Ask panel is never recorded: replay pauses while the panel is open (`?ask=1`), as it does on `/chat`. It is disclosed here for completeness; if that rule were ever widened to cover school accounts, PostHog would be added to the table above under the Art. 28 change-notification clause first.

Footnote — RevenueCat and Stripe are not engaged for schools. From the paid-plan launch (recorded 7 November 2026, §11), an adult who buys the full plan in the iPhone or Android app does so from Apple or Google, with **RevenueCat** keeping track of the subscription, and an adult who buys on the web pays through **Stripe**. School pupils and staff get the full plan through the school's invoice and are never shown a price, and nobody under 18 or of unknown age is ever offered a purchase, so **no school's pupil or staff data flows to RevenueCat or Stripe** and neither is a sub-processor for a school deployment. The one case that touches a pupil is a parent or guardian choosing to pay for a child's plan directly on the web: that purchase is between the parent and Clean Wave Software Limited as controller, outside the school's contract. If a school deployment were ever to use either provider, it would be added to the table above under the Art. 28 change-notification clause first.

Footnote — SMS (removed). SMS was removed from the product in release 4.15 (Phase 145): there is no Twilio sub-processor, no phone number is collected and no SMS is sent. A dormant `user_profiles.sms_phone` column from the earlier SMS work still holds one legacy value; it is to be cleared and the column dropped. Should SMS ever return, it would be added to the table above under the Art. 28 change-notification clause first.

---

6. Retention

Retention is described concretely in the companion **Data Retention Statement**. In summary: deletion requests get a **14-day reversible grace**, and the account is then erased **within a day of the grace period ending, by a scheduled job** (corrected 14 October 2026 from "*shortly after the grace period ends, by a job that runs hourly*": the job is scheduled hourly but in practice starts every four to eight hours) — the sign-in account, every row keyed to it and its stored files, through one routine that retries a failed attempt rather than dropping it (a failure found only after the account itself has been deleted cannot be retried and is reported to the operator instead; see the statement §1); teacher-authored records about a pupil (progress reports, the teacher's annotations on the pupil's pieces) are **deleted with the pupil's account**, and no automatic "pupil left a teacher" anonymisation flow exists; a pupil's erasure deletes their message threads, including the teacher's view of them; an owner cannot schedule deletion while their studio, department or ensemble has other members (they hand it over or close it first), and a group the person owns that has no other members is deleted with them; pupils have a **self-serve export**; backups are provider-managed. See that statement for the honest detail (including what is *not* automated).

Corrected 12 October 2026 (Phase 104). The summary above used to read "*deletion requests get a 14-day reversible grace then an hourly hard-purge; teacher-authored records are anonymised, not destroyed, when a pupil leaves*". Neither was true: the hourly job deleted practice data but never the sign-in account, profile, messages or files, and no flow redacted teacher records when a link ended. **What survives an erasure:** complaints the person filed and the moderation record, both de-identified; when an operator erases an account, a one-way SHA-256 fingerprint of its email kept with that audit row (pseudonymous personal data, so a later complaint can be matched); bare user ids on other people's rows (a moderation action's target, the rights attestation on a received copy, a quarantine ledger row's owner) with no other personal data; and guardians' copies of notices about a child, kept as the guardian's own record. **Refused, not erased:** operator accounts, accounts with admin access, and accounts under a CSEA statutory hold, which are handled by an operator step (`content-moderation-runbook.md` §5). **Added 12 October 2026 (WR-05):** a fourth case is refused the same way: an account whose storage folder holds files that pages belonging to *other* accounts point to, so that erasing it would blank their pages. No real account is in that state (it arises only in seeded demonstration data); the operator moves those pages first. **Added 12 October 2026 (WR-04):** a statutory hold is not permanent. It is lifted by two recorded operator steps — the hold on each held file is ended, then the hold on the account is recorded as released, each on the NCA's and the police's confirmation — after which the account is erased by the ordinary routes, including the person's own pending deletion request. The record of the release (operator, time, stated reason; no email address) is kept in the moderation record; the release row itself is deleted with the account. **Added 12 October 2026 (WR2-02):** the hourly job also does not erase an account that is **suspended** when its grace period ends. A person's own deletion request then waits until the review of the report about the account is finished, so that an unattended job cannot destroy what the review is about; an operator then lifts the suspension (the job erases the account on its next run) or erases the account from the admin screen. This delays an erasure request for the length of a review and is stated in the privacy policy. The *Data Retention Statement*'s 90-day post-deletion police metadata hold never existed and has been struck there (§3a).

Added 31 October 2026 (Phase 144) — the owner block, and closing a studio. Deletion is refused for an owner whose studio, department or ensemble has other members, until they hand it over to another member or close it. This protects the other members' data: before this date, deleting the owner deleted the group for everyone in it. The app refuses to schedule such a deletion, and the hourly job refuses it as a sixth case if members joined during the grace period. An operator's erasure from the admin screen is not refused, so a group with members can still go with its owner where an operator decides it must. Closing a studio or department deletes its lessons and payment records along with its memberships, student links and settings. Members keep their own pieces and practice history. Closing is refused while any of its library material is under a moderation hold. Closing also removes the studio's guardian-gate override records, as deleting an ensemble already did. This is recorded as a known gap in the *Data Retention Statement* §5.

Added 1 November 2026 (Phase 147) — a reported Ask answer. A user can report an Ask answer (an answer the in-app assistant gave them) from the answer itself. The operator then reads only that answer, not the question or the rest of the chat, using the service key on the admin reports screen. The report stores no chat content: only the reporter's chosen reason and their own words, and a pointer to the answer. It is attributed to no user, because the app wrote the answer. If the chat is deleted, the answer goes with it and the report is closed as unverifiable.

Added 7 September 2026 — moderation retention, which the summary above did not cover. Content removed after a report is no longer deleted outright: it is held in a private bucket for **30 days** so that a wrong removal can be reversed, and an hourly job then deletes it permanently. The window is an operator setting rather than a constant, so if it is changed the figure changes with it. Material retained because a report was made to the National Crime Agency under the CSEA reporting duty is **not** part of that window: it runs on the statutory clocks instead — five years for the report reference, one year for the content and the uploader's associated data — and it is ended by a deliberate operator step, never by the hourly job. See §3a and §3b of the *Data Retention Statement*.

Added 15 September 2026 — the readiness consent record. A member's decision to share how they are getting on inside an ensemble is stored as a row in `ensemble_readiness_consents`, and its retention is **scoped to the life of the ensemble and of the account**, not to a period. Leaving an ensemble, or being removed from it, is a **soft revoke**: a database trigger flips the row to `revoked` and stamps `revoked_at`, so the record of what was shared and when it stopped survives, and a member who re-joins shares nothing until they choose again. Deleting the **ensemble** or the **account** is a **hard delete**: both foreign keys are `ON DELETE CASCADE`, so the rows go with whichever is deleted, which is the same posture as every other row keyed to a studio or to a user and what the account-deletion promise in the privacy policy requires. It is **not** retained indefinitely as an audit record that outlives either, and this pack says so explicitly because an earlier draft of the migration's own commentary claimed otherwise and was corrected under review (WR-07). See §3c of the *Data Retention Statement*, and `supabase/migrations/185_readiness_flag_and_consent.sql` §B and §E.

7. Children's Code applicability

Because PracticeCoach is offered direct-to-consumer to users aged 13+, the ICO's **Age Appropriate Design Code (Children's Code)** applies to the service. Shipped alignment includes: teen-held consent, protective defaults, data minimisation (age-bands not dates of birth in teacher views), live age transitions, and — for school studios — **teacher↔pupil DMs off by default**, enabled only knowingly by the Head of Department with the retention/disclosure understanding above. Your DPIA should record the Children's Code as an applicable framework. ICO guidance here is under review following the **Data (Use and Access) Act 2025** — re-check at pilot time.

Ensembles — the two children's decisions, recorded 7 September 2026 as §8 of this pack required. Both were filed at `docs/compliance/online-safety-risk-assessment.md` §6A before any ensemble table existed, and both are now enforced in the shipped system:

1. **Guardian visibility, and the gate that makes it reliable.** A linked guardian of a member under 18 sees every piece that reaches that member through an ensemble — with the sharer, the ensemble and the rights basis claimed — and is sent a notice on each of the four events that can put something there: joining an ensemble, a part copied into their library, an update to one accepted, and an ensemble date reaching their calendar. Because that promise is worth nothing where no guardian link exists, an account aged 13–17 cannot join an ensemble at all without a live, accepted, consented guardian link, and an account whose age band cannot be read is treated as a minor rather than waved through. The single exception is the audited administrator override described in §1: one account, one ensemble, a mandatory reason, and every grant and revocation in the moderation log.
2. **Roster visibility.** Where an ensemble contains anyone under 18, non-owner members see only the owner, any leaders and a total count — never a list naming younger members. The **owner is the exception** and receives the full roster (display name, role, join date and nothing else), because the owner is the accountable party who removes members and promotes leaders and cannot do either without seeing them. No age, age band, date of birth, email or per-member under-18 flag is disclosed to anyone in either mode; the only age-shaped value that leaves the server is a boolean property of the *ensemble* that identifies nobody.

Ensemble readiness — two further children's decisions, recorded 15 September 2026. Both follow the Code's teen-held-consent and protective-defaults expectations, and both are enforced in code rather than recorded here as intentions:

- **The consent is the young person's own, and a linked guardian is told on both transitions.** A member aged 13–17 turns readiness sharing on and off themselves — it is their own progress data, and the Code expects the choice to be theirs — but a **linked guardian is sent a notice when the young person turns it on, and a notice when the young person turns it off**. Both halves matter, and the off half is the one worth writing down: a guardian told only about the switch going on would be left believing sharing is still running after the young person had stopped it. The notices are bounded (a repeat of the same event is silent and a daily budget of ten per member caps a consent storm) and they say only that the setting changed — never the readings themselves, never an evaluation, never anything about another member. **What is promised is the sending and the recording, not an unconditional arrival**, and the published policy is worded to match (review CR-01, Phase 90): the notice rides the `announcement` category, whose frequency defaults to weekly, so it normally reaches the guardian within a week rather than at the moment of the tap; a guardian who has switched that category off receives none; once the daily budget is spent the notice is logged as skipped while the consent write itself stands; and a revoke whose pre-read of the consent id fails goes un-notified by construction, because the id is unrecoverable once the RPC has run. Each of those is a recorded limitation of the control rather than a silent one, and none of them changes what the member themselves sees: the card states the current setting either way.
- **The age switch decides who reads it, and the member's own row is always there.** Where an ensemble contains anyone under 18, only the owner and adult leaders read other members' readiness and everybody else reads only their own, so one minor's progress is never legible to another. The composition of the roster is not disclosed by the shape of the view either — see the withheld trio in §2. A member sees their own readings in either mode, before and after consenting, which is what makes the setting legible enough to consent to.

---

8. Ensemble sharing — pending item discharged (recorded 2026-09-07)

This section was a pending item and is now the record of what was done about it. It is rewritten rather than deleted, because a pack whose history has been tidied away cannot show a supervisory authority when it learned something.

What it said, filed 2026-09-03. That a member-to-member sharing path was built but not live and that this pack did not yet cover it; that Phase 79 of v4.0 had added "ensembles" — a closed, invitation-only group in which a member can publish a piece they uploaded so other members can read it, under one of three declared rights bases (`docs/compliance/rights-bases-decision.md`); that it was the first path by which one user's uploaded sheet music becomes readable by another user's account, and therefore a new sharing path in the sense `CLAUDE.md` requires to be documented in the same pull request; and that the whole surface sat behind the `ensembles` runtime flag, which was false and had never been written to true, so no personal data flowed through it and no processing described in §1–§6 changed. It named what would have to be written when the feature was turned on: *an additional purpose (sharing music within a consented group), an additional recipient class (other members of a group the data subject joined), the rights-attestation fields recorded against the sharer, and — for members under 18 — the guardian-visibility and roster-visibility decisions filed at* `docs/compliance/online-safety-risk-assessment.md` *§6A.* Due: Phase 85.

What was decided and done, 7 September 2026. The pack now covers the path, in the four places §8 itself named:

- **§1** carries the purpose — sharing music within a consented group — together with how the music actually moves (a copy per recipient; the sharer's markings frozen and read-only) and a dated correction of the guardian-gate sentence that Phase 84 overtook.
- **§2** carries the rights-attestation fields recorded against the sharer, and the additional recipient class.
- **§6** carries the 30-day moderation review hold, with the statutory CSEA clocks beside it.
- **§7** carries the guardian-visibility and roster-visibility decisions, with their §6A filing.

What has not changed: the feature is off. The `ensembles` flag is still false and has still never been written to true. This pack now describes the processing as built so that the description precedes the switch rather than following it, which is the order a DPIA is supposed to run in. The switch itself, the `TERMS_VERSION` bump and the Ofcom toolkit sign-off at `online-safety-risk-assessment.md` §7 remain the accountable person's own acts and are not discharged here.

Sibling documents for the same change: `src/pages/PrivacyPolicy.tsx`, the user-facing statement of the same facts, updated 7 September 2026; and `docs/compliance/data-retention-statement.md` §3a–§3b, the retention detail this pack summarises.

9. Device storage and sign-out (recorded 9 October 2026)

Why this section exists. On 8 October 2026 the operator found that one account's offline downloads and cached data stayed on a device after sign-out and were shown to the next account signed in on it. Schools have shared tablets and families share devices, so this matters for a school's DPIA. The fix shipped in Phase 101. Every claim below is filed from the automated test that holds it; behaviour with no test behind it is listed as such at the end. The Online Safety Act record of the same fix is `online-safety-risk-assessment.md` §7 ("Offline data surviving sign-out — closed 9 October 2026").

9.1 What a device holds

For offline use, the app keeps copies of the signed-in account's data on the device. **The model is one owner per device:** at any time the offline stores belong to at most one account, recorded in a device-level owner stamp (`pc:offline-owner`).

Store	What it holds	Platforms
Offline database (IndexedDB, `practice-coach-offline`), seven tables	`queryCache` — the offline read cache of the account's data; `rqCache` — the persisted in-app query cache; `syncQueue` — changes made offline, waiting to reach the server; `offlineDownloads` — manifests of music downloaded for offline use; `offlinePageBlobs` — downloaded page images on the native apps; `pendingUploads` — recordings made offline, waiting to upload; `annotationHistory` — the annotation editor's undo history	All
Browser caches (Cache API)	`performance-pages-v1` — downloaded page images on the web; `sheet-music-images` — the service worker's image cache	Web and Android (the service worker runs in the Android WebView); guarded on iOS, where it is believed absent (unverified)
Native files (Capacitor Filesystem)	`Data/offline-pages` — downloaded page images; `Cache/shared-out` — files the user exported or shared (data exports, backups, recordings, PDFs, calendar files), plus export files older builds wrote to the cache root	Android and iOS
Web Storage (localStorage, sessionStorage)	A registry classifies every key as one of four classes. **Device** keys are device preferences such as theme, sidebar state, metronome and pen settings, and the owner stamp. **Pre-auth** keys are sign-up hand-offs such as an invite token, class code or terms tick. **Auth session** keys hold the signed-in session. **Account** keys are everything else, such as cached plans, the practice checkpoint, the active studio and chat history.	All
Partner and sign-up hand-offs (added 7 November 2026, §11)	`pc_partner_code` (localStorage, 30 days): which partner link brought the visitor, so the account can be attributed when it is created. `pc_pending_signup_heard_about` (sessionStorage): the "How did you hear about us?" answer, held until the account exists, then saved with it and removed.	All

The **registry fails safe.** A key that nobody has classified counts as account data and is removed, so a new feature's key cannot leak by being forgotten. A source scan fails the build's tests if any storage key in the code is not classified. Filed from `src/lib/__tests__/deviceStorageKeys.test.ts` ("every storage key in src is classified", "no key is listed in two classes").

9.2 When it is removed — three purge modes

One module (`src/lib/deviceDataIsolation.ts`) is the only place account data leaves the device. Every mode empties all seven tables, both caches and both native directories. The modes differ only in which Web Storage classes survive:

Mode	When it runs	Keeps
`sign-out`	The user signs out. This covers every sign-out control, the password-reset completion, an invitation-only refusal at sign-up, and an account confirmed deleted.	Device keys only
`identity-change`	A different account arrives on a device stamped with another owner, or on an unstamped device holding another account's data.	Device keys, plus the arriving account's session and sign-up hand-offs
`reset-data`	The user chooses Settings → reset all data, after the server deletion succeeds.	Device keys, the session, and the push token, so the same user stays signed in with reminders working

Filed from `src/lib/__tests__/deviceDataIsolation.test.ts` :

- "every Dexie table is purged — the runtime table list equals PURGED_TABLE_NAMES". A table added later fails this test until it is consciously classified.
- The three "mode '…'" cases.
- "rejects on any other Filesystem error — but only after every other store was cleared".
- "rejects Dexie writes while the purge is running".
- "the purge removes shared-out whole and only the legacy export files in the Cache root".

At sign-out, in order:

1. The app releases this device's push subscription (§9.4).
2. It signs out of the server. If that fails, or takes longer than 4 seconds, it removes the stored session itself, so the account is not silently signed back in on the next launch.
3. It runs the purge, and retries once if it fails.
4. On success it stamps the device as having no owner. If the purge fails twice, it leaves the departing account's stamp in place, so the next different account purges first, and tells the user on the sign-in screen.

Filed from `src/contexts/__tests__/AuthContext.test.tsx` , under "sign-out orchestrator (101-09)". A source scan fails if any code other than this one function calls the server sign-out (`src/lib/__tests__/signOutEntryPoints.test.ts`).

Before anything is lost, the user is asked. If changes are still queued, the app first tries to send them. If it cannot, or if music is downloaded, one dialog states what signing out will remove ("3 changes haven't reached the server yet because you're offline. If you sign out now, they'll be lost.", "Signing out removes the 12 pieces downloaded for offline use from this device.") and defaults to staying signed in. Filed from `src/hooks/__tests__/useSignOutFlow.test.tsx` and `src/components/__tests__/SignOutDialog.test.tsx`.

9.3 The owner check

Each time an account arrives — at boot, on sign-in, or when an offline session is restored — the app compares it with the owner stamp **before the account's screens render**:

Stamp	Result
Same account	Nothing happens.
No owner	The app adopts the arriving account.
A different account	The app purges behind a blocking screen.

If the purge fails, the blocking screen stays with "Try again" and "Sign out". The app is never shown over the previous account's data. Filed from `AuthContext.test.tsx`: "identity guard at every arrival site", including "SIGNED_IN for a different account shows the gate in the same render as the user", and "a rejected isolation shows the failed gate and never fetches the profile".

Devices updated from an earlier version have no stamp. On those, the app looks for any other account's identifier in the stored data: the stored session, the cached profile, queued changes, download manifests and the page addresses in them, persisted queries, pending recordings and chat history. If it finds any, or cannot read a store, it purges. Otherwise it adopts the arriving account, so a single-user device keeps its downloads through the update. Filed from `deviceDataIsolation.test.ts`: the fifteen "legacy:" cases.

Defence in depth. Each queued change records the account that made it. The queue never sends, counts or lists another account's entry, and the two per-account settings writes refuse an entry from another account (`syncQueue.test.ts` "owner binding (D-09)"; `syncHandlers.test.ts`). Download manifests record their owner, and the *Available offline* list shows only the signed-in account's downloads (`performanceDownloads.test.ts`, "never lists another account's downloads, nor a legacy row with no owner"). The offline read cache will not write a result that arrives after the owner has changed (`offlineCache.test.ts`, "owner-captured write-through").

9.4 Push notifications

At sign-out, the app releases this device's push subscription using the departing user's token. This is best effort, bounded to 3 seconds, and does not change the user's notification preference on the account. As a server backstop, each time a device subscribes, any other user's row for the same endpoint or FCM token is deleted, so a device delivers notifications to one account only, even after an offline sign-out. Filed from:

- `api/_lib/__tests__/pushEndpointOwnership.test.ts`: "deletes every row for the endpoint held by a different user…", "never touches user_profiles"
- `src/lib/__tests__/pushDevice.test.ts`: "with a subscription and keepPreference true: posts release-device", "a failing fetch still unregisters and never rejects"

9.5 Residuals — what is not removed, stated rather than left to be inferred

- **Involuntary sign-out keeps the data, locked to the same account.** If a session ends without the user choosing it (an expired or revoked token, or a sign-out in another tab), nothing is purged and the stamp stays. This lets a pupil whose session lapses offline still have their downloads when they sign back in. A different account purges the data before it opens. The same applies to an account deleted on the server while a device is still signed in. The data is purged when the app confirms, online, that the account is gone, or when another account arrives. Until then it stays at rest, and no one can open it in the app without that account's credentials. Filed from `AuthContext.test.tsx`: "an involuntary SIGNED_OUT keeps the data and the stamp…" and the four "account gone" cases.
- **The browser's or WebView's own HTTP cache.** Page images can also sit in the browser's ordinary HTTP cache, which page code cannot clear. These are the public, unguessable sheet-music addresses already described in the privacy policy. They carry no titles, and the browser expires them under its own rules.
- **Service-worker expiration metadata** (`workbox-expiration`) is not deleted, because the worker holds it open. It contains URLs and timestamps only.
- **Deletion is not secure erasure.** Stores are removed through the platform's normal deletion calls. Freed storage is not overwritten, and the offline stores are not encrypted at rest.
- **iOS is unverified.** The code guards every iOS difference, but the only iOS test device cannot run current builds, so none of this has been exercised on iOS.
- **Not traced to a test.** The Settings "reset all data" screen calling the `reset-data` purge is code-reviewed, not test-covered. The mode itself is tested; the call from Settings is not.

10. Sign-up and first-run records (recorded 5 November 2026, Phase 161)

Phase 161 adds three things. None of them adds a sub-processor: all three are held by Supabase in the EU (§4, §5), and the sign-up email that carries the confirmation link already goes through ZeptoMail.

First-run and Home usage events. The app writes first-party events named `firstrun.*` (setting-up steps, such as adding a first piece or finishing a first session) and `home.*` (which Home cards are tapped, and which Home prompts are shown, tapped, dismissed or snoozed) to the existing `usage_events` table (migration 067). Their properties are short enum values only: no titles, notes, chat text or other content a person types, and no pupil content. Clients can insert their own events and cannot read any; only the admin dashboard reads them, through the service key. The inventory of every event and its allowed properties is `docs/analytics/USAGE_EVENTS.md`. Retention: see `data-retention-statement.md` §3e.

Transient sign-up metadata. When a person signs up with an email address, the details they enter (date of birth, any invite token, studio join code or guardian token, and the Terms version they accepted) are put in the sign-in record's user metadata, so that sign-up survives email confirmation on another device. When the app creates the profile, it re-validates these details on the server side rather than trusting them (the date of birth must be a real date and 13 or over; otherwise it is left out and the existing date-of-birth prompt applies), then scrubs them with a best-effort update. That update sets each key to null rather than deleting it, so the keys stay present with null values; this is accepted as minimisation, because no value remains. Until the scrub, the values also ride in the person's access token, which is why the privacy policy says they are held "briefly … until your account is set up, then removed".

Added 5 November 2026 (Phase 161 code review CR-S01). Two groups never reach that scrub, because no profile is ever created: a sign-up refused while registration is Invite only, and an address that never opens its confirmation link. For the first, the app clears the same keys from the sign-in record before it signs the refused account out (best effort, bounded to three seconds). For both, a retention sweep in the existing hourly notifications job (`api/_lib/cron/signupMetadataSweep.ts`) clears the keys from any sign-in record that is more than seven days old and still has no profile, so the privacy policy says that if the account is never set up the details are "removed from it once seven days have passed". The sweep is non-destructive by decision: it does not delete the sign-in record, so the name and email address remain with it (an invite sent later still signs the same account in). Deleting never-set-up sign-in records outright was considered and not adopted for now.

`signup_admissions` **(migration 252).** One row per admitted account: the user id, the admission route (`open` , `platform_invite` , `studio_invite` , `guardian` or `join_code`) and the time. It lets the profile insert be refused for anyone not admitted while registration is Invite only, and feeds the daily sign-up count to the operator. It is server-only (service role grant only, RLS on with no policies), is deleted with the account by `ON DELETE CASCADE` on `auth.users` , and is not part of the self-serve export or a personal backup (`src/lib/backup/schema.ts` , 2026-11-05 note), because the route names an invite or studio a restored account was never admitted through.

11. Plans and payments (recorded 7 November 2026, Phase 175)

Phase 175 records, for paid plans, the data stores built in phases 169, 173, 174, 176 and 177 and the providers a purchase uses. **No sub-processor is added for a school deployment** (§5 footnote): pupils and staff get the full plan through the school's invoice, and a minor never sees a price. A parent paying for a child directly on the web is outside the school contract, with Clean Wave Software Limited as controller. Every table below lives in Supabase in the EU (§4). None is part of the self-serve export or a personal backup (`src/lib/backup/schema.ts` , 2026-11-07 notes), because a restore must never grant a new account a plan nobody paid for, or attribute it to a partner who never referred it.

`entitlements` **(migration 254).** One row per paid subscription (Apple, Google or Stripe) or admin grant of the full plan: status, source, plan, period end, grace end, whether it renews, the provider's customer, subscription or transaction ids, and, for an admin grant, the operator's reason (200 characters at most) and the ids of the admins who granted, changed or ended it (`granted_by` , `changed_by` , `ended_by` , set to null if that admin's account is deleted). Server-only: RLS on with no policies and only a `service_role` grant, so a person sees their plan only through the account overview function. Deleted with the account (`ON DELETE CASCADE`). The plan's name is mirrored on the profile (`user_profiles.subscription_tier` , `pro` or `free`), which the server refreshes and never reads for a decision. Retention: *Data Retention Statement* §3f.

`studio_access_endings` **(migration 254).** One row when a person's studio access ends because the studio was deleted or their membership removed: the studio's id, kind and name, the role, the end date and the end of the 14-day grace. Server-only (RLS on, no policies, `service_role` grant). The hourly downgrade sweep deletes rows **30 days** after the end date (`free_tier_prune_access_endings` , migration 260, called from `api/_lib/cron/downgrades.ts`); rows also cascade with the account.

`downgrade_choices` **and** `downgrade_notices` **(migration 260).** When the full plan is ending, `downgrade_choices` holds the works and pieces a person chose to keep active (`keep_work_ids`, `keep_piece_ids`, a server-stamped `chosen_at` and `applied_at`); the owner can read and write only the keep lists. `downgrade_notices` is server-only and records what the person was told and when: the kind (`upcoming`, `reminder`, `applied`), the access end date, `shown_at`, `confirmed_at`, `floor_from` and, for an applied downgrade, the archived piece ids, their count and whether a saved choice steered it. Both cascade with the account; "Start over" clears the choice. `downgrade_notices` is **kept until account deletion**: applied rows are the person's trace of what moved to their archive. The downgrade reminder is a push notification in a new `plan_change` category, sent to adults only, naming no piece and showing no price.

`user_attribution`, `signup_survey` **and** `partner_code_redemptions` **(migration 261).** Which partner and code brought an account, how (link or typed code) and when; the "How did you hear about us?" answer; and which limited partner offer the account redeemed. All server-only and deleted with the account (`ON DELETE CASCADE`). **Pupils see both questions:** every sign-up form, including the school `JoinStudio` page, shows the optional partner-code box and the heard-about question (`signupExtras.scan.test.ts`). A pupil's answers are collected and stored, and **never used to market anything to them**; they are counted in totals for partner statements and to learn where people find the app. Partner contact details and bank-transfer notes are business contacts, readable only by admins.

`payment_events` **(migration 261).** The financial record of each payment, refund, chargeback and chargeback reversal: channel, type, date, the period paid for, gross, tax, fees or commission, net, currency, whether it was a trial or a promotional free period, the provider's reference, and the partner it is attributed to. Append-only and server-only. When the account is deleted the row stays with `user_id` **set to null** (`ON DELETE SET NULL`), because UK law requires company accounting records to be kept for **6 years** from the end of the financial year they relate to. Partner statements show only counts and an **8-character reference** per payment, never a name or email.

`trial_claims` **(migration 263, built in Phase 176).** To allow one free trial per person: the channel (web or store), for a web trial a one-way SHA-256 hash of the card's Stripe fingerprint (never the card number and never the raw fingerprint), for a store trial the store's original transaction id, the provider reference and the time of the claim. The lawful basis is legitimate interest in preventing repeated free trials. Server-only: RLS on with no policies, `-- server-only` marker, and only `service_role` holds SELECT, INSERT and DELETE. When the account is deleted the row stays with `user_id` **set to null** (`ON DELETE SET NULL`), so a deleted-and-recreated account cannot claim a second trial with the same card. Kept **3 years from the claim**, then deleted by the nightly billing reconcile (`api/_lib/cron/billingReconcile.ts`). Not in the export or a personal backup. Retention: *Data Retention Statement* §3f.

`billing_webhook_events` **(migration 262, built in Phase 176).** One row per message from Stripe or RevenueCat, so each is processed exactly once: the provider, the event id and type, the environment (sandbox or production), the subject reference (the Stripe subscription or charge id, or the RevenueCat app user id), the account id, the processing status and any error text. The message payload is stored only until the event is processed, with names, email addresses, postal addresses and every RevenueCat subscriber attribute except `partner_code` and `prompt` removed before it is written. The payload is **deleted on success**; a failed event's payload is kept **at most 30 days** while the reconcile retries it, then nulled; the event rows themselves are deleted after **90 days** by the nightly billing reconcile. Server-only (RLS on, no policies, `service_role` grant only). `user_id` is set to null if the account is deleted (`ON DELETE SET NULL`). Not in the export or a personal backup.

Plan-record and studio additions (migration 262). `entitlements.provider_environment` records whether a store purchase was a sandbox test, so App Review and tester purchases can be told apart from real ones and never reach `payment_events` or the partner ledger; `entitlements.last_reconciled_at` records when the nightly reconcile last re-read the provider. A paid studio subscription stores its band, plan, period, source and the plan record it comes from on `studios` (`billing_band`, `billing_plan`, `billing_period`, `billing_source`, `billing_entitlement_id`), written only by the server; no studio member can change them, and they go with the studio.

`stripe_customers` **(migration 268, Phase 177).** The Stripe customer reference for each account that has paid on the web: the account id, Stripe's customer id and whether it is a live or test customer. Purpose: reuse one Stripe customer per account so a person's plans and invoices stay together. Lawful basis: contract. Recipient: Stripe, which issued the reference. Server-only (RLS on, no policies, `-- server-only` marker, `service_role` grant only). Deleted with the account (`ON DELETE CASCADE`); the erasure job reads it first to cancel any active Stripe subscription before the account is deleted. Stripe keeps its own customer and financial records under its own terms. Not in the export or a personal backup. Retention: *Data Retention Statement* §3f.

A parent or guardian paying for a child (Phase 177, `entitlements.payer_user_id`, migration 268). The child's plan record names the paying account (payer) alongside the account that receives the plan (beneficiary). Purpose: only the paying account can manage or cancel the plan, and the child sees no payment details. Lawful basis: contract (with the paying adult). Recipient: Stripe, through the payment itself. The child is never shown a price; the guardian opens checkout from their own account through a link the child can pass on. The link is removed when either account is deleted: the plan record cascades with the child's account, and `payer_user_id` is set to null (`ON DELETE SET NULL`) when the paying account is deleted, at which point the child's subscription is set to cancel at the end of the paid period and the plan lasts until then. This is the only route by which a purchase relates to a pupil, and it sits outside a school's contract.

RevenueCat (in-app subscriptions). RevenueCat receives the account's id as its app user id, the store's purchase record, and a `partner_code` attribute when the account came through a partner. **No email address is sent.** From Phase 178 the app identifies the store customer to RevenueCat only by that Practice Coach account id (the Supabase user id, never an anonymous or device id), may also set the subscriber attributes `offer_id` and `patreon_offer` (which store offer was used for the purchase, and whether it was the Patreon members' rate), and leaves RevenueCat's device identifier collection off. Processing may be in the United States, under the safeguards UK and EU law requires (confirm the mechanism at signing). **Apple and Google** are independent sellers of in-app purchases and process the payment under their own privacy policies.

Stripe (web payments). Stripe receives the payer's name, email, card details (held by Stripe; Clean Wave never sees the full number), billing country and address, and tax. Stripe acts either as Clean Wave's processor or, under Stripe Managed Payments, as merchant of record (the seller); which applies is not yet settled, and the privacy policy says both and that it will be updated. In either case Stripe is an independent controller for its own fraud-prevention and legal duties. Processing may be in the United States, under the safeguards UK and EU law requires.

Device storage. The two partner and sign-up hand-off keys, `pc_partner_code` (30 days) and `pc_pending_signup_heard_about` (until the account exists), are listed in §9.1.

Emails. Trial-ending and annual-renewal reminders and cancellation confirmations are sent by email through ZeptoMail, the existing email provider (§5), to the adult who bought the plan.

This pack provides information to support the school's own DPIA under UK GDPR Art. 35; the school remains the controller and owns that assessment. Prepared July 2026 against ICO guidance (including the June 2026 "Edtech examined" report). It is not a completed DPIA and not legal advice. Regulatory positions and sub-processor facts verified 2026-07-10; re-verify against current ICO guidance and each provider's live terms at signing. Amended 7 September 2026 to record ensemble sharing in §1, §2, §6 and §7, to discharge §8, and to correct the guardian-gate sentence in §1 that Phase 84 overtook. Amended 9 October 2026 to add §9, device storage and sign-out (Phase 101). Amended 14 October 2026 to correct the erasure timing in §6 to "within a day". Amended 28 October 2026 to record, in §5, that the opt-in session recordings (PostHog) are never engaged for school accounts, and, in §4, the optional Google sign-in. Amended 1 November 2026 to record, in §6, that a user can report an Ask answer and what the operator then reads. Amended 3 November 2026 to record, in §4, the optional Sign in with Apple. Amended 5 November 2026 to add §10, sign-up and first-run records (Phase 161): the first-run and Home usage events, the transient sign-up metadata and its scrub, and `signup_admissions` *. Amended 6 November 2026 to record Google Firebase Cloud Messaging in §4 and §5 and the Vercel and Sentry regions. Amended 7 November 2026 to add §11 (plans and payments), the §5 footnote that RevenueCat and Stripe are not engaged for schools, and the partner and sign-up hand-off keys in §9.1. Amended 8 November 2026 (Phase 176) to record in §11 that* `trial_claims` *is built (migration 263, 3 years, kept past deletion with the person removed) and to add* `billing_webhook_events` *(migration 262, redacted payloads, 30 days on failure, 90-day event log),* `entitlements.provider_environment` *and the studio billing fields. Amended 8 November 2026 (Phase 177) to add to §11 the Stripe customer reference (* `stripe_customers` *) and the guardian-payer link (* `entitlements.payer_user_id` *). Amended 8 November 2026 (Phase 178) to record in §11 that the RevenueCat app user id is the account id, the* `offer_id` *and* `patreon_offer` *subscriber attributes, and that device identifier collection is off. Amended 10 November 2026 to record, in §2, the flag counts a linked teacher and the department see: counts and passage names only, under the existing consent objects.*