Applicant Import — Developer Handover (v2)

Importing Google Form applicants into Luther Thorne Primary & St. Giles Primary

Revision: one-step welcome email + automatic status-change notifications. Documents Requested keeps its existing token-based flow.

Mission

Take applicant rows + attachments out of the Google Form / Drive and stand them up in EduConnect for two schools so parents can log in immediately and track their application status. After import, every change to application status (except **Documents Requested**) automatically emails the parent.

HARD RULES	1. Always set **school_id** on inserted rows. RLS hides un-scoped rows. 2. Always insert with **status = 'pending'**. 3. Use **import-applicant-account** to create the auth user + send the welcome — never call auth.admin.createUser directly. 4. Do not modify the **documents_requested** branch — it owns its own email + token flow. 5. Welcome email is sent **once** at import. Status changes use a separate notifier.

Email Lifecycle (one welcome + status updates)

There are now **three** distinct email touchpoints. Do not introduce a fourth.

Trigger	Function	Email
Import (new applicant, no existing account)	import-applicant-account	Combined 'Application Received — Welcome' with Set Password &a
Import (parent already has an account)	bulk-send-applicant-welcomes (Group A)	Light 'New Application Received — Log In with existing credentials'.
Status change to documents_requested	registration-document-request	Tailored email containing a public token link to upload requested file
Any other status change (under_review, accepted, rejected, …)	send-application-status-update (NEW)	Short 'status updated' email with Log In button. Fires automatically o

Why the change	Previously, parents received a Set Password welcome at import but nothing afterward when the school moved them off Pending. v2 keeps the single welcome at import (so parents can log in immediately and see status = Pending) and adds an automatic status-update email on every transition. The Documents Requested status keeps its richer token-based flow.

Operational Runbook

Pre-flight

• Confirm school IDs: Luther Thorne f96a806f-…, St. Giles ca148d70-….
• Sanity-check CSV columns map to the import dialog's expected fields (child_name, parent_email, parent_name, phone, child_date_of_birth, relationship, plus any form_data.* fields).
• Drop attachments into the appropriate Storage bucket folder before import so URLs in form_data.attachments

resolve.

Step-by-step import

1. Open Admin → Pending Approvals → **Import Applicants**.
2. Pick the target school (one school per import run).
3. Upload CSV. Map columns. Review the preview — duplicates are flagged automatically.
4. Tick **Send welcome emails**. Click Import.
5. Verify in parent_registrations: rows have status='pending', the correct school_id, and a populated form_data.
6. Confirm welcome emails: query email_send_log filtering on template_name='auth_emails' for Group B and 'transactional' entries for Group A. Sent count should equal Group A + Group B from the dry-run.

Dry run first (always)

```
curl -X POST $FUNCTIONS_URL/bulk-send-applicant-welcomes \
  -H "Authorization: Bearer $ANON_KEY" -H "Content-Type: application/json" \
  -d '{"school_id":"<SCHOOL_ID>","dry_run":true}'

Returns counts and per-recipient classification (Group A vs Group B) without sending anything.
```

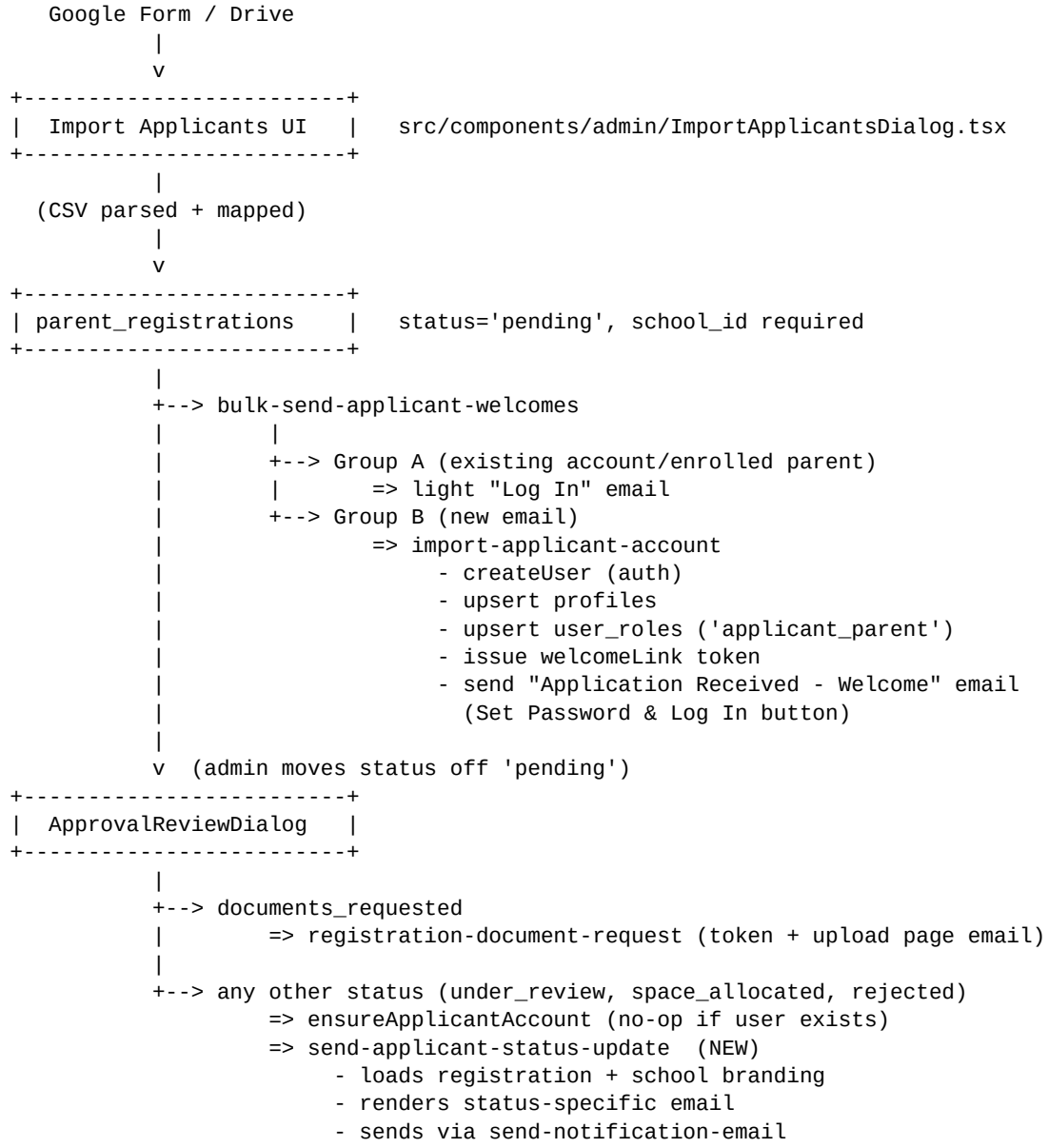
Status changes (automatic emails)

Once imported, the registration team works through Pending → Under Review → Space Allocated/Rejected via the Approval dialog. Every transition (except Documents Requested) automatically invokes send-applicant-status-update and the parent gets a short notification email with a Log In button.

For **Documents Requested**: the admin types the request message in the dialog, and registration-document-request creates a token-protected upload page and emails the parent the link. **Do not** wire send-applicant-status-update into this branch — it would duplicate the email.

Technical Appendix

Architecture



Edge functions touched

Function	Role	Touch?
import-applicant-account	Creates auth user + sends combined welcome (Group B). Existing short-circuit prevents duplicate em	Do not change
bulk-send-applicant-welcomes	Classifies emails into Group A/B and routes accordingly. Has dry_run.	Do not change
registration-document-request	Owns the documents_requested email + token + public upload page.	Do not change
send-applicant-status-update	NEW. Sends a short status-change email for every transition except documents_requested.	Created by this change
send-notification-email	Generic email transport used by all of the above.	Do not change

send-applicant-status-update inputs

```
POST /functions/v1/send-applicant-status-update
Body: {
  "registration_id": "<uuid>",
  "new_status": "under_review" | "space_allocated" | "rejected" | ...,
  "school_id": "<uuid>"
}

Behavior:
  - Returns {skipped:true} when new_status === "documents_requested"
  - Returns {skipped:true} when registration has no parent_email
  - Otherwise renders branded HTML and sends via send-notification-email
```

Database surface

• **parent_registrations** — application rows. Always (school_id, status='pending', form_data JSONB).
• **profiles** — upserted by import-applicant-account. id = auth user id, school_id set, registration_approved=true, profile_completed=false.
• **user_roles** — applicant_parent role assigned at import. Enrolled parents already hold a 'parent' role at the same school — that's how Group A is detected.
• **schools** — used for branding (name, logo_url, primary_color) in both welcome and status-update emails.

RLS & school-scoping (do not relax)

Parent registrations are only visible to admins/principals/teachers scoped to the matching school_id (plus admins with NULL school_id who get island-wide access). Inserting a row without school_id hides it from everyone. All edge functions use the service role internally — never expose the service role key client-side.

Do-not-touch list

DO NOT	• Do not send marketing/bulk emails to applicant lists from these functions. • Do not change the userExisted short-circuit in import-applicant-account — it is what guarantees one welcome email per parent. • Do not invoke send-applicant-status-update for documents_requested — that branch already emails the parent. • Do not bulk-send without first running dry_run:true. • Do not modify registration-document-request or its registration-followup-documents bucket. • Do not store the service role key in client code or in DB tables.

Common failure modes

• **Welcome email never arrived**: check email_send_log for the recipient. If status='dlq' the queue exhausted retries — usually a hard bounce or invalid address.
• **Status email did not fire**: check edge function logs for send-applicant-status-update; verify school_id was passed and the registration has parent_email.
• **Parent can't see their application after login**: confirm profiles.id matches the auth user id and user_roles has applicant_parent for that user.
• **Group B parent received the light Group A email instead**: a profile already existed for that email — expected. They use existing credentials.