Capsule Deployment — New Site Playbook

This guide covers moving Capsule to a **new site/domain** (e.g., different org, region, or environment) with minimal friction and zero credential leaks.

---

1) Decide your topology

- **New Vercel project** per site (recommended)
- **Supabase**: choose one
- **A. New Supabase project** per site (clean isolation)
- **B. Reuse existing project** with separate schemas (faster, less isolation)

> Tip: For vendors/clients, use **A**. For a second region within the same org, **B** is fine if RLS is strict.

---

2) New domain + DNS

- Point your **custom domain** to Vercel.
- Verify HTTPS (Vercel auto TLS) and HSTS enabled.

---

3) Environment variables (per site)

Create `.env.production` (Vercel → Project → Settings → Environment Variables):

```
NEXT_PUBLIC_SUPABASE_URL=
NEXT_PUBLIC_SUPABASE_ANON_KEY=
SUPABASE_SERVICE_ROLE_KEY=
APP_BASE_URL=https://new.example.com
OPENAI_API_KEY=
```

---

4) Google OAuth

In **Google Cloud Console → Credentials** for the new site:

- Add **Authorized redirect URI**:
- `https://new.example.com/auth/callback`
- (keep `http://localhost:3000/auth/callback` for local)
- In **Supabase → Authentication → Providers → Google**:

- Ensure the **same Client ID/Secret** or create a new pair if required by org policy.

  If using a **new Supabase project**, re-enable the Google provider there.

---

5) Supabase setup for the new site

If **new project**:

- Create tables: `meetings`, `transcripts`, `actions` (or run your schema migration pack)
- Run FTS migration: `supabase/migrations/*_add_fts_indexes.sql`
- Create Storage bucket `recordings/` (private)
- Verify **RLS** policies (see Security section)

If **reusing project**:

- Option 1: separate **schema** (e.g., `capsule_site2.*`)
- Option 2: add a `tenant_id` column to all tables and **enforce RLS** per tenant.

---

6) App configuration

- Set `APP_BASE_URL` to the new domain.
- Update any hardcoded links in emails/Slack messages.
- Confirm CORS: Supabase → **Auth** and **API** settings allow your new origin.

---

7) Build & deploy

- Create a **new Vercel project** from the same GitHub repo or a fork.
- Select the correct branch (main or release).
- Add env vars (Section 3).
- Deploy. Confirm `200` at `/` and OAuth redirect.

---

8) Post-deploy smoke tests

- **Login**: Sign in with Google on the new domain.
- **Upload**: Upload a small audio file; confirm it appears in Storage.
- **Transcribe**: Verify Whisper → transcript row created.
- **Summarize**: Check notes + actions are generated.
- **Search**: Try global search for a known keyword.
- **Integrations**: Send a test Email + Slack message.

---

9) RLS quick checklist (Supabase)

For each table: `meetings`, `transcripts`, `actions`

- Add `owner_id` (UUID) or `tenant_id` (UUID/text).
- Enable **RLS**.
- Policies:
- `select`: only members of the tenant/team or record owner
- `insert/update/delete`: owner or service role
- Storage: bucket `recordings` **private**, signed URLs only; expiry 60–120 mins

---

10) Content routing & branding

- Update **logo/wordmark** if the new site needs white-labeling.
- Change **email From** name/domain (SPF/DKIM records for deliverability).
- Optional: new Slack channel default.

---

11) Monitoring & alerts

- Vercel: enable logs, set up alerts on 5xx spikes
- Supabase: database logs + auth events
- Optional: Sentry for frontend & API route error tracking

---

12) Rollback plan

- Keep previous Vercel deployment as a **ready rollback** target.
- Database: snapshot before schema changes.
- Storage: ensure lifecycle rules retain objects for N days.

---

13) Migration checklist (copy-paste)

-

Appendix A — Staging vs Production

Recommended split:

- **staging.example.com** → staging Supabase project + OpenAI key (lower quota)
- **example.com** → production Supabase + OpenAI key

Use Vercel **Environments** (Preview, Production) with different env vars.

Appendix B — Supabase schema snippet

Example multi-tenant column and policy stub:

```
alter table meetings add column if not exists tenant_id uuid;
create policy tenant_read on meetings for select using (
  auth.uid() is not null and tenant_id = auth.jwt() ->> 'tenant_id'
);
```

> Replace policy with your auth strategy (e.g., map Google domain to tenant).