What to use (all free/near-free)

Frontend

- React + Vite + TypeScript + Tailwind
- Host: **Azure Static Web Apps (Free tier)**

Auth

- **Microsoft Entra External ID (B2C)** — free up to 50k MAU

Backend

- **Azure Functions (Python)** on **Consumption** (pay-per-use; stays ~€0 on light MVP)

Database

- **SQL Server Express (local)** for development (use SSMS)
- When you need cloud: **Azure SQL Database (Basic/S0)** using trial credits (migrate later)

Storage (images/files)

- **Azure Blob Storage** (covered by trial credits; cents/month later)

Observability

- **Application Insights** (works with Functions; small usage ~free)
- Optional: **Sentry Free** (client + server errors)

CI/CD

- **GitHub Actions** → deploy to Static Web Apps + Functions

Analytics

- **Power BI Desktop** (free) against your local SQL Express or Azure SQL

Minimal project blueprint

```
kidtokid/
  frontend/              # React + Vite + Tailwind
    .github/workflows/swa.yml
  api/                   # Azure Functions (Python / FastAPI-in-Function)
    host.json
    local.settings.json  # (dev only)
    HttpTrigger/__init__.py
    HttpTrigger/function.json
  db/
    scripts/seed.sql     # sample schema + seed
  infra/
    env.sample           # all env vars in one place
    README.md
```

Env (sample)

```
SQL_CONN_STR=Server=localhost\SQLEXPRESS;Database=KidToKid;Trusted_Connection=
True;Encrypt=False;
BLOB_CONN_STR=DefaultEndpointsProtocol=...  # dev: use local Azurite or real
Blob
ENTRA_TENANT_ID=...
ENTRA_CLIENT_ID=...
ENTRA_CLIENT_SECRET=...
```

Step-by-step “Today Plan” (quick and free)

1. **DB (local & free)**

- Install **SQL Server Express** (you have SSMS).
- Create DB and seed:
  - Open SSMS → New Query → run `db/scripts/seed.sql`.

2. **Backend (Functions on Consumption)**

- VS Code → Azure Functions extension → “Create New Project” → Python → HTTP trigger.
- In `HttpTrigger/__init__.py`, connect to SQL using `pyodbc` and return JSON.
- Test locally: `func start`.

3. **Frontend (Static Web Apps Free)**

- `npm create vite@latest frontend -- --template react-ts`
- Add Tailwind, call your Functions endpoint (e.g., `/api/HttpTrigger`).
- Test locally: `npm run dev`.

4. **Blob Storage (trial credits)**

- In Azure Portal, create **Storage Account** → Blob container `uploads`.
- In dev, either use this real account or **Azurite** locally (completely free).

5. **Auth (free)**

- Set up **Entra External ID (B2C)** tenant.
- Create an App Registration (SPA + web API).
- Frontend uses MSAL; Functions validates tokens (easy middleware).

6. **CI/CD (free)**

- Push to GitHub.
- Create **Azure Static Web App (Free)** → connect your repo → set `frontend/` build.
- Add an **API** path pointing to `api/` (Functions auto-wired).
- GitHub Actions will build & deploy on every push.

7. **Monitoring (free)**

- Turn on **Application Insights** for the Function App.
- Optional: add **Sentry** DSN to frontend/backend env.

8. **Power BI (free Desktop)**

- Connect **Power BI Desktop** to your **local SQL Express** (DirectQuery or Import).
- Build your first report → save `.pbix`.
- (If you later need the cloud service, that's a paid SKU—stay on Desktop for free.)

Why this works for you

- Uses your **SSMS + SQL Express** now (no DB cost).
- **Functions + Static Web Apps** keep hosting at ~€0 on the Trial and beyond for light traffic.
- Clear upgrade path: swap conn string to **Azure SQL** when ready; everything else stays.

If you want, I can spit out:

- a ready-to-run **Functions HTTP endpoint** hitting SQL Express,
- a **React page** calling it,
- and a **GitHub Actions** workflow for Static Web Apps + Functions.