**TDA-5 — Frontend Scaffolding & Build Pipeline**

**Epic:** Foundation & Infrastructure  
**Status:** Done  
**Owner:** Metehan / Core Team  
**Date:** *[fill in]*

**1) Summary (What & Why)**

We initialized the **frontend stack** using **React + Vite (TypeScript)** and delivered a minimal “Hello World” UI to validate the toolchain. The frontend reads the backend API base URL from environment variables (VITE\_API\_BASE\_URL) and can run locally and build for production without errors.  
On the backend side, we exposed a **/health** endpoint and configured **CORS** to allow the Vite dev origins. We also resolved database SSL issues at startup (asyncpg + SSL context), ensuring predictable backend startup and a clean health signal for the frontend.

**2) Scope of Work Completed**

* Bootstrapped **React + Vite (TS)** in /Frontend.
* Implemented basic component structure with App.tsx that renders:
  + Page title (“Turkish Diaspora App”)
  + “Hello World”
  + Display of VITE\_API\_BASE\_URL
  + A simple health check calling GET /health
* Added **environment configuration** with .env.development and .env.production.
* Created a tiny **API helper** (/Frontend/src/lib/api.ts) to centralize base URL access and the health call.
* Backend:
  + Added **/health** route
  + Added **CORS** for http://localhost:5173/5174 (Vite dev)
  + Fixed DB SSL verification error in dev (asyncpg with ssl.SSLContext)
* Optional root route returning a friendly JSON on / (prevents 404 on root).

**3) Directory & Key Files**

/Frontend

├─ index.html

├─ package.json (vite scripts: dev/build/preview)

├─ vite.config.ts

├─ .env.development (VITE\_API\_BASE\_URL=http://127.0.0.1:8000)

├─ .env.production (VITE\_API\_BASE\_URL=https://api.example.com) ← adjust later

└─ src/

├─ main.tsx

├─ App.tsx

└─ lib/api.ts (exports API\_BASE\_URL, getHealth)

Backend

└─ app/

├─ main.py (adds /health, CORS, startup DB ping)

└─ db.py (async SQLAlchemy + asyncpg, dev SSL context)

**4) Environment Configuration**

* **Frontend reads:** VITE\_API\_BASE\_URL
  + **Development:** Frontend/.env.development → http://127.0.0.1:8000
  + **Production:** Frontend/.env.production → set real API domain before deploy
* Note: Vite only exposes vars prefixed with VITE\_.

**5) Backend Integration**

* **CORS** allows:
  + http://localhost:5173, http://127.0.0.1:5173
  + http://localhost:5174, http://127.0.0.1:5174 (when 5173 is busy)
* **Health endpoint**: GET /health → { "status": "ok", "version": "…" }
* (Optional) **Root endpoint**: GET / → friendly JSON with links to /docs and /health.

**6) DB SSL Fix (Dev)**

* Resolved CERTIFICATE\_VERIFY\_FAILED by:
  + Converting Pydantic AnyUrl → str for SQLAlchemy
  + Stripping sslmode from the URL to avoid asyncpg kwarg issues
  + Passing an **SSLContext** (verification off) via connect\_args={"ssl": ssl\_ctx} in **dev only**
* For **production**, verification should be enabled (default SSL context).

**7) How to Run (Local)**

**Backend**

cd Backend

source .venv/bin/activate

uvicorn app.main:app --reload

# visit http://127.0.0.1:8000/health

**Frontend**

cd Frontend

npm run dev

# browser opens http://localhost:5173 (or 5174 if 5173 is occupied)

**Production Build**

cd Frontend

npm run build

# outputs to Frontend/dist without errors

**8) Acceptance Criteria (Traceability)**

* npm run dev starts local development server → **Met**
* npm run build creates production-ready build without errors → **Met**
* Frontend reads API base URL from VITE\_API\_BASE\_URL → **Met**
* Basic App.tsx renders page title → **Met**
* (Bonus) Frontend displays backend health status → **Met**

**9) Definition of Done**

* Frontend development server runs → **Done**
* Production build succeeds → **Done**
* Environment configuration working → **Done**
* Basic component structure in place → **Done**
* Backend /health reachable and CORS configured → **Done**

**10) Evidence / Screenshots (Observed)**

* Frontend shows:
  + Title + “Hello World”
  + API base URL: http://127.0.0.1:8000
  + Health: “Backend says: ok”
* Backend logs:
  + {"event":"Checking database connection..."}
  + {"event":"Database connection OK."}

**11) Risks, Notes & Follow-ups**

* **Dev SSL** currently disables verification; must be revisited before production (enable verification and/or load CA bundle).
* Consider adding **CI** to validate the frontend build on every push (GitHub Actions).
* Deployment next steps:
  + Frontend to Netlify/Vercel (static)
  + Backend to Render/Fly.io/Supabase functions/etc.
  + Set VITE\_API\_BASE\_URL via environment on the host.