# Full Project Flow Summary

## 1. Project Overview

This is an Offline-First Multi-Tenant Web Application built with Angular and NestJS.   
Users can manage customer data locally (offline) and sync it to a central PostgreSQL database (online).   
Each tenant works in complete isolation using a tenantId.

## 2. Technology Stack

|  |  |
| --- | --- |
| Frontend | Angular 12+, TypeScript, HTML, CSS |
| Local Storage | SQLite using sql.js (in-browser DB) |
| Backend | NestJS (v8+), TypeScript |
| Server Database | PostgreSQL |
| ORM | TypeORM |
| Sync Mechanism | Manual push to /customers/sync |
| Multi-Tenancy | Handled via tenantId and guards |

## 3. Setup Instructions

* Backend (NestJS):

1. Install dependencies: npm install

2. Configure .env for DB credentials.

3. Run the server: npm run start: dev

4. Port-http://localhost:3000

* Frontend (Angular):

1. 1. Install dependencies: npm install
2. 2. Run the Angular dev server: ng serve
3. 3. Access the app at: <http://localhost:4200>

## 4. Application Flow

* **Tenant Creation** - New tenants can be created using a simple form. Data is stored locally or synced when online.
* **Customer Management (CRUD)** - Add, Edit, Delete customers through a table UI. Uses Angular Reactive Forms and local SQLite. Email and Phone must be unique — enforced at the backend.
* **Offline Usage** - Application is fully usable offline. All data is saved in the browser using SQLite.
* **Syncing to Server** - Triggered manually via a Sync button. Only unsynced local records are sent. Conflicting records (e.g., duplicate email) are skipped and reported.
* **DB Online Status (In HomePage)-** The homepage shows whether the server database is online or not. Helps users know when it’s safe to sync.

For CREATE----->>Create ->Submit

For READ--------->>Get All Records

For UPDATE----->> Login->Login with email->edit->update

For DELETE----->> Get All Records->Delete

**For Sync->**

**Create a tenant offline**

You will get to see ---offline-sync disabled

**Come back online**

You will get to see ---Online ready to Sync--🡪Press on Sync, you will get an alert

## 5. Sync Process

1. Offline records are saved with isSynced = false.
2. On sync, frontend calls: POST /customers/sync.
3. Backend inserts records with proper createdByTenantId.
4. Uniqueness is enforced on email and phone: Duplicates are skipped and reported as failed.
5. Backend returns { synced, failed }.
6. Local DB updates successful records to isSynced = true.

## 6. Known Limitations

|  |  |
| --- | --- |
| Limitation | Notes |
| No Auth | No login/session mechanism |
| Conflict Resolution | No merge UI; failed syncs must be fixed manually |
| No Background Sync | Manual sync only (no auto sync) |

## 6. Additional Info

DB\_HOST=localhost

DB\_PORT=5432

DB\_USERNAME=postgres

DB\_PASSWORD=

DB\_DATABASE=tenant\_db

Table=customer