

Run this **exactly from A → Z** to get everything working locally (server + client + DB + seed + tests).

0) Prerequisites

Install:

- **Node.js 18+** (recommended)
- **npm** (comes with Node)

Check versions:

```
bash
node -v
npm -v
```

1) Create the project folder + files

You already created the structure in Part 1. If not, re-run the PowerShell creation commands from Part 1, then paste all files from Parts 1–5.

Your final folder should look like:

```
Text
collab-playlist-100/
  package.json
  .env.example
  README.md
  server/...
  client/...
```

2) Install dependencies (root + server + client)

From **project root**:

```
bash
npm run install:all
```

This runs:

- `npm install (root)`

- `npm --prefix server install`
 - `npm --prefix client install`
-

3) Create the server env file (REQUIRED)

Create this file:

```
server/.env
env
PORT=4000
DATABASE_URL="file:./dev.db"
(You can copy from server/.env.example.)
```

4) Initialize Prisma + create SQLite tables

From **project root**:

```
bash
npm run db:migrate
```

This runs `prisma migrate dev` inside `server/` and creates:

- `server/dev.db` (SQLite database)
 - migration history under `server/prisma/migrations/`
-

5) Seed the database (40 tracks + 10 playlist items)

From **project root**:

```
bash
npm run db:seed
```

This loads:

- 40 diverse tracks
 - 10 playlist items
 - vote variety
 - exactly one now playing
-

6) Start the project (API + UI)

From **project root**:

```
bash
npm run dev
You should have:
```

- **Client UI:** <http://localhost:3000>
- **API server:** <http://localhost:4000>
- **SSE stream:** <http://localhost:4000/api/stream>

Sanity checks:

```
bash
# should return a message
curl http://localhost:4000/

# should return JSON list of tracks
curl http://localhost:4000/api/tracks

# should return JSON list of playlist items
curl http://localhost:4000/api/playlist
```

7) Realtime test (manual)

Open **two tabs** at:

<http://localhost:3000>
Try in Tab A:

- vote a track
- drag reorder
- click Play

Tab B should update instantly.

8) Offline queue test (manual)

1. Stop the server (Ctrl+C in terminal), or disconnect internet.
2. In the UI, try:

- vote
 - remove
 - reorder
3. You'll see the UI status go offline and the **Queue count** increase.
 4. Restart server:

```
bash
```

```
npm run dev
```

5. When status goes online, queued actions replay.

9) Run all tests (server + client) — single command

From **project root**:

```
bash
```

```
npm test
```

This runs:

- server tests (Jest + Supertest + SSE test)
- client tests (Vitest position test)

If tests fail the first time because DB isn't migrated/seeded, run:

```
bash
```

```
npm run db:migrate
```

```
npm run db:seed
```

```
npm test
```

10) Common commands you will use often

Start only the server

```
bash
```

```
cd server
```

```
npm run dev
```

Start only the client

```
bash
```

```
cd client
```

npm run dev

Reset database completely (clean slate)

Stop server first, then:

```
bash
# from project root
rm -f server/dev.db
npm run db:migrate
npm run db:seed
(Windows PowerShell alternative)
```

```
PowerShell
Remove-Item -Force server\dev.db
npm run db:migrate
npm run db:seed
```

If you want, I can verify your setup quickly

Send me:

1. A screenshot of npm run dev terminal output (server + client)
2. Output of:

```
bash
curl http://localhost:4000/api/playlist
```