1) Project setup & discovery (what to build / acceptance criteria)

1. **Create a one-page Product Spec** (deliverable): list the minimum features you'll ship for an MVP. Use the Questel page for inspiration — the product typically includes:
    - trademark portfolio records and dossiers (cases, owners, status, documents)
    - docketing / deadlines and automatic reminders
    - renewals / fee management and invoice tracking
    - watch/monitoring and search integrations
    - document management (PDFs, office actions, evidence of use)
    - reporting / dashboards and user roles (partner, attorney, paralegal)
    - (optional) AI helpers for drafting goods & services or reviewing search/watch results. Questel
2. **Define success / acceptance criteria** for the MVP (clear, testable). Example: "Create and store a trademark record with owner, status and three associated documents; generate a docket entry and send an email reminder X days before a deadline; run a USPTO TSDR query and display the results."
3. **Stakeholders / users:** define personas (attorney, paralegal, admin). Capture workflows (create file, docket deadline, evidence upload, run watch).

---

2) Tech stack recommendation (what you'll build with)

(Feel free to swap components — these are pragmatic choices for an iMac local server + Cursor-assisted dev.)

- **Frontend:** React + TypeScript + Tailwind CSS (component-driven, fast to iterate).
- **Backend:** Node.js + TypeScript with Express or NestJS (good ecosystem + easy to scaffold).
- **Database:** PostgreSQL (reliable relational DB for docketing & reporting).
- **Background jobs:** Redis + BullMQ (for scheduled reminders, watch polling).
- **Search / similarity:** Postgres full-text for MVP; add OpenSearch/Elasticsearch later if needed.
- **Storage:** local disk for dev; MinIO (S3-compatible) if you want S3-like behavior locally.
- **Auth:** JWT for API; use dev OAuth or local LDAP if needed for firm integration.
- **Dev / containerization:** Docker / Docker Compose to run Postgres, Redis, MinIO locally.
- **APIs to integrate:** USPTO TSDR / Open Data (for status & docs) and WIPO / Global Brand DB or EUIPO for international info. (You can use these programmatically where public APIs exist). developer.uspto.gov+1

You'll use **Cursor** as your coding assistant for scaffolding, multi-file edits, test generation, and refactors. Cursor supports macOS and can read your repo context to make suggestions. Cursor

---

3) Environment & local iMac server setup (concrete steps / commands)

Perform these on the iMac that will be the local server.

1. **Install core tools**
   - Install Homebrew (if not present):
     `/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"`
   - Install Git, Node, Docker, and mkcert:
     `brew install git node docker docker-compose mkcert`
   - Install Cursor (download from Cursor site or app): grab from https://cursor.com/downloads. Cursor
2. **Configure Docker**
   - Start Docker Desktop and enable the Docker daemon.
   - Create a `docker-compose.yml` with services: `postgres`, `redis`, `minio` (optional), and `app` (for the backend).
   - Example (skeleton):

```
version: '3.8'
services:
  db:
    image: postgres:15
    environment:
      POSTGRES_USER: app
      POSTGRES_PASSWORD: secret
      POSTGRES_DB: trademarks
    volumes:
      - db-data:/var/lib/postgresql/data
  redis:
    image: redis:7
  minio:
    image: minio/minio
    command: server /data
    environment:
      MINIO_ROOT_USER: minio
      MINIO_ROOT_PASSWORD: minio123
    ports: ["9000:9000"]
volumes:
  db-data:
```

24. **Set static local IP / DNS for dev**
    - If other devices need to access the iMac, configure a static local IP via macOS Network settings, or use a local name (e.g., `equinox.local`) with Bonjour.
    - For HTTPS in dev, use `mkcert` to create a local CA and certs, or rely on HTTP for strictly internal dev.
25. **Firewall & security**

  - Keep ports closed externally. If you expose services to LAN, ensure macOS firewall rules and user authentication are in place.
  - Regular backups: schedule `pg_dump` + copy to external disk or versioned directory.

---

4) Project structure & repository bootstrap

Create a mono-repo (recommended) with folders:

```
/equinox-mvp
  /services
    /api (Node/Nest/Express)
    /web (React)
    /jobs (worker scripts)
  /infra
    docker-compose.yml
    init-db.sql
  /docs
  /scripts
```

- Initialize Git and a clear README with run instructions.
- Add `.env.example` showing required env vars (DB URL, Redis URL, TSDR API key placeholder).

---

5) Build features incrementally — concrete dev steps (MVP → v1)

Follow small vertical slices (feature complete, testable end-to-end):

Slice A — Authentication + basic UI

- Scaffold backend API with users, roles, and JWT auth.
- Scaffold React app with login screen and a simple dashboard.
- Use Cursor to generate endpoints and React components (prompt examples below).

Slice B — Trademark record CRUD + document upload

- DB schema: trademarks (id, mark_text, owner_id, status, reg_num, class(es), filing_date, country), owners, documents, docket_entries.
- API endpoints: create/update/list/search trademark records; upload document endpoints (store metadata in DB, files in MinIO/local disk).
- Frontend: record page, upload widget, document viewer (embed PDF).

Slice C — Docketing & Reminders

- Implement docket entry model (type, due_date, created_by, linked trademark).
- Worker job that checks for upcoming deadlines and enqueues/send reminders (email via SMTP or local Mailcatcher).
- Add UI to create recurring deadlines and view upcoming tasks.

Slice D — Integrations: USPTO / WIPO lookups

- Implement a connector to call **USPTO TSDR** or USPTO Open Data to fetch status and documents (requires reading their API docs & signing up for any API key). Show fetched results on the trademark record page. developer.uspto.gov+1
- Consider a modular "connector" system so you can add EUIPO/WIPO later (WIPO Global Brand DB / Madrid Monitor for international checks). WIPO

Slice E — Watch & Monitoring (polling or webhook)

- Implement a watch resource: user chooses marks/owners to watch.
- Worker periodically calls watch sources (or mock for MVP) and creates events if new results found.
- UI: watch dashboard, alerts, and the ability to accept/reject a watch result.

Slice F — Reporting, exports, and admin

- CSV/XLSX exports of portfolios and upcoming deadlines.
- Basic reports (counts by status, upcoming renewals).

---

6) Use Cursor effectively (how it helps you)

Cursor can speed up many tasks — scaffold, multi-file edits, tests, documentation. Suggested workflow:

- **Scaffold**: ask Cursor to create a TypeScript express route + DTOs for a trademark CRUD endpoint.
- **Refactor**: ask Cursor to rename a model and update references across files.
- **Write tests**: prompt Cursor to generate Jest + Supertest integration tests for your endpoints.
- **Create UI components**: instruct Cursor to scaffold a React form component tied to your API.
- Example prompt (in Cursor):
  ```
  Create a NestJS controller file for "trademarks" with
  get/list/post/delete endpoints, DTOs for create and update, and unit
  test skeletons. Use TypeScript and Prisma schema model "Trademark".
  ```
- Cursor docs emphasize multi-file edits and natural-language editing (so feed it clear prompts and your repo context). Cursor+1

---

7) Testing, QA & acceptance

- **Automated tests:** unit tests for critical logic, integration tests for API endpoints, end-to-end tests (Playwright) for user flows.
- **Manual QA checklist:** create trademark, upload docs, create docketing item, simulate reminder, run USPTO lookup and ingest results.
- **Security reviews:** ensure file uploads are scanned/validated, SQL injection prevention (use parameterized queries/ORM), store secrets securely in `.env` not in repo.

---

8) Deployment & running on the iMac (how to run locally)

1. Clone repo on iMac.
2. Start services: `docker compose up -d`
3. Run migrations: `yarn workspace api prisma migrate deploy` (or `npm run migrate`).
4. Start backend: `yarn workspace api dev` (or `npm run dev`) — bind to `0.0.0.0` so other machines on LAN can reach it.
5. Start frontend: `yarn workspace web start` — configured to talk to the API host (use iMac IP).
6. Background worker: `yarn workspace jobs start` (or run via `pm2`/systemd if you want persistent process).

Add a `start.sh` script so a junior engineer can run a single command to bring up everything.

---

9) Data migration & import

- Provide CSV import for existing portfolios (map required columns, validate).
- Provide an “import dry run” mode that reports errors but doesn’t write to DB.

---

10) Logging, monitoring & backups

- Logging: centralize server logs (rotate logs); for dev use console + files.
- Monitoring: basic health endpoint `/health` and a simple status page.
- Backups: cron `pg_dump` to mounted external drive or backup folder.

---

11) Security & legal considerations

- **Data protection:** client data is sensitive. Encrypt backups, restrict access, and keep audit logs.
- **Third-party API terms:** when using USPTO, WIPO, EUIPO APIs, read their terms & rate limits. Some APIs require registration and rate limiting (e.g., USPTO TSDR Open Data). developer.uspto.gov+1
- **No copying UI or proprietary workflows**: use Questel's public page only as product inspiration — don't copy proprietary UI/assets or internal functionality that's behind a license.

---

12) Documentation, handover & next steps

- Maintain an up-to-date README with runbook for the iMac server.
- Create a short "How we used Cursor" doc with example prompts and shortcuts so the junior engineer can reproduce your AI-assisted workflow.
- Prepare onboarding notes: how to run migrations, run tests, and perform backups.

---

13) Suggested milestone checklist (deliverables)

- Repo initialized + dev docs + Docker compose
- Auth + user roles + basic dashboard
- Trademark CRUD + document upload + viewer
- Docketing + reminder engine + worker
- USPTO integration for status/docs
- Watch feature (basic)
- Tests + CI checks (GitHub Actions or local precommit hooks)
- README and runbook for the iMac server

---

14) Helpful prompts & Cursor tips (quick wins)

- "Create a React TS component `TrademarkForm` with fields: mark_text, owner, classes, filing_date, country, and integrate form submit to POST `/api/trademarks`."
- "Write a Jest integration test that creates a trademark, attaches a document, then fetches the trademark and asserts the document appears."

- Use Cursor to **open the repo** and ask it to "make me a migration that adds `docket_entries` table" — Cursor can make multi-file edits.

Sources & reference

- Questel Equinox Law Firm product page (features & areas to mirror). Questel
- Cursor — product and features (use Cursor to scaffold, multi-file edits, downloads). Cursor+1
- USPTO developer APIs (TSDR / Open Data) — for status, docs and programmatic access. developer.uspto.gov+1
- WIPO Global Brand Database / Madrid Monitor — international brand data reference. WIPO