Next.js with tRPC Project Structure

tRPC integration for end-to-end type-safe APIs with React Query.

#nextjs #react #typescript #trpc #api

PNG PDF Copy Prompt

Project Directory

```
my-app/
  app/
    layout.tsx
    page.tsx
    globals.css
    api/
      trpc/
        [trpc]/ tRPC HTTP handl…
          route.ts
    posts/ Example feature…
      page.tsx
      [id]/
        page.tsx
  server/ tRPC backend co…
    trpc.ts tRPC init and c…
    root.ts Root router
    routers/ Feature routers
      posts.ts
      users.ts
  lib/
    trpc.ts Client hooks (a…
    trpc-server.ts Server-side cal…
  components/
    providers/
      trpc-provider.tsx QueryClient + t…
    posts/ Feature compone…
      post-list.tsx
      post-form.tsx
  next.config.js
  package.json
  tsconfig.json
  .env.local
  .gitignore
```

Why This Structure?

tRPC provides end-to-end type safety between client and server without code generation. Define procedures in `server/routers/`, and TypeScript automatically infers types in client components. The `server/` folder keeps backend logic separate from the App Router.

Key Directories

`server/` - tRPC routers, context, and procedures

`server/routers/` - Feature-based router files

`lib/trpc.ts` - Client hooks: `api.posts.useQuery()`

`lib/trpc-server.ts` - Server caller for RSC

`app/api/trpc/[trpc]/` - HTTP handler route

Router Example

```
// server/routers/posts.ts
export const postsRouter = router({
  list: publicProcedure.query(async () => {
    return db.post.findMany()
  }),
  create: protectedProcedure
    .input(z.object({ title: z.string() }))
    .mutation(async ({ input }) => { ... })
})
```

Getting Started

```
npm install @trpc/server @trpc/client @trpc/react-quer
npm install @tanstack/react-query zod
npm run dev
```

When To Use This

- Full-stack TypeScript projects
- Teams wanting API type safety without GraphQL
- Rapid prototyping with instant API changes
- Projects with React Query for caching

Trade-offs

`TypeScript only` - Both client and server must use TS

`Tight coupling` - Frontend/backend share type definitions

`No REST/OpenAPI` - Not suitable for public APIs