Express with Drizzle Project Structure

Express with Drizzle ORM—SQL-like queries with full TypeScript inference and lightweight runtime.

#express #nodejs #typescript #drizzle #database #sql

PNG PDF Copy Prompt

Project Directory

```
myproject/
  src/ Application sou…
    index.ts Server startup
    app.ts Express setup
    db/ Database layer
      index.ts Drizzle client …
      schema.ts Table definitio…
    migrations/
      .gitkeep
  routes/
    index.ts
    user.routes.ts
    post.routes.ts
  controllers/
    user.controller.ts
    post.controller.ts
  services/
    user.service.ts Uses Drizzle qu…
    post.service.ts
  middleware/
    error-handler.ts
    validation.ts
  types/
    index.ts
  drizzle.config.ts Drizzle Kit con…
  tests/
    setup.ts
    user.test.ts
  package.json
  tsconfig.json
  .env.example
  .gitignore
```

Why This Structure?

Drizzle gives SQL-like syntax with full TypeScript inference—no code generation step. Schema defined in TypeScript means your IDE knows your tables. Lightweight runtime works great on serverless and edge.

Key Directories

src/db/schema.ts - Table definitions as TypeScript code

src/db/index.ts - Drizzle client configured and exported

src/db/migrations/ - SQL migrations generated by Drizzle Kit

drizzle.config.ts - Migration and push configuration

Getting Started

1. `npm install`
2. `cp .env.example .env` (set DATABASE_URL)
3. `npx drizzle-kit generate`
4. `npx drizzle-kit migrate`
5. `npm run dev`

Schema Definition

```ts
// src/db/schema.ts
import { pgTable, serial, varchar } from 'drizzle-orm/pg-c

export const users = pgTable('users', {
  id: serial('id').primaryKey(),
  name: varchar('name', { length: 255 }),
  email: varchar('email', { length: 255 }).unique(),
});
```

When To Use This

- Prefer SQL-like query syntax over methods
- Need lightweight runtime for serverless
- Want schema-as-code without generation step
- Deploying to edge runtimes
- Starting new projects with modern tooling

Trade-offs

Newer ecosystem - Fewer tutorials and community resources

SQL knowledge needed - Queries look like SQL—need to know SQL

Less abstraction - More explicit than Prisma