Phoenix Minimal Project Structure

Standard Phoenix app with MVC structure. Ideal for learning Phoenix or small web applications.

#phoenix #elixir #web #backend #mvc

PNG PDF Copy Prompt

Project Directory

```
my_app/
    mix.exs  Project definit…
    mix.lock
    .formatter.exs  Code formatter …
    .gitignore
  config/  Runtime configu…
    config.exs  Shared config
    dev.exs
    prod.exs
    test.exs
    runtime.exs  Runtime secrets
  lib/  Application code
    my_app/  Business logic …
      application.ex  OTP application
      repo.ex  Ecto repository
      accounts/  Context module
        accounts.ex  Context API
        user.ex  Ecto schema
    my_app_web/  Web layer
      endpoint.ex  HTTP entry point
      router.ex  Route definitio…
      telemetry.ex
      controllers/
        page_controller.ex
        page_html.ex  View module
        page_html/
          home.html.heex
      components/  Reusable UI
        core_components.ex
        layouts.ex
        layouts/
          app.html.heex
          root.html.heex
  priv/  Private assets
    repo/
      migrations/
      seeds.exs
    static/  Static files
      favicon.ico
      robots.txt
  assets/  Frontend source
    css/
    js/
    tailwind.config.js
  test/
    test_helper.exs
    my_app/
      accounts_test.exs
    my_app_web/
      controllers/
        page_controller_test.exs
    support/
      conn_case.ex
      data_case.ex
```

Why This Structure?

Phoenix 1.7+ structure with the new verified routes and HEEx templates. The `lib/` directory splits cleanly: `my_app/` for business logic (contexts, schemas) and `my_app_web/` for web concerns (controllers, components). Contexts like `Accounts` encapsulate domain logic behind a clean API.

Key Directories

`lib/my_app/` - Business logic, contexts, and Ecto schemas

`lib/my_app_web/` - Controllers, components, router, endpoint

`priv/repo/` - Database migrations and seeds

`config/runtime.exs` - Production secrets and environment config

Getting Started

1. `mix phx.new my_app`
2. `cd my_app && mix deps.get`
3. `mix ecto.create`
4. `mix phx.server`

When To Use This

- Learning Phoenix fundamentals
- Small to medium web applications
- Traditional request/response apps
- CRUD-heavy applications
- Teams new to Elixir

When To Upgrade

- Need real-time features (use LiveView template)
- Multiple distinct applications (use umbrella)
- Complex domain requiring bounded contexts
- Deploying multiple services from one codebase

Naming Conventions

Modules - PascalCase matching directory: `MyApp.Accounts`

Files - snake_case: `accounts.ex`, `user.ex`

Contexts - Plural nouns: `Accounts`, `Products`, `Orders`

Schemas - Singular nouns: `User`, `Product`, `Order`