

# 项目结构

MkSaaS 样板文件和文件夹组织概述

MkSaaS 采用基于 Next.js App Router 的模块化架构,注重可扩展性和可维护性。本指南介绍了样板文件中的关键目录和文件,以帮助您更有效地浏览代码库。

## 根目录结构

🗅 源码
口行动
🗅 分析
□ 应用程序
口资产
口成分
□ 配置
□分贝
口钩子
□ 国际化
□库
□邮件
口 通讯
口支付
🗅 贮存
□ 通知
口 商店
□样式
□类型
🗅 中间件.ts
□ 路线.ts
口内容
□民众
□消息
Dockerfile
drizzle.config.ts
□ 下一个.config.ts
□ 源.config.ts

# 关键文件

src/niddleware.ts: 包含路由保护逻辑和重定向
 src/routes.ts: 定义一致导航的核心路线
 drizzle.config.ts: 数据库的 Drizzle ORM 配置
 next.config.ts: Next.js 配置,包括重定向和重写
 source.config.ts: Fumadocs 文档配置
 env.example: 环境变量模板

# 关键目录

# /src/app 目录

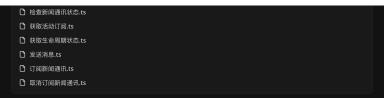
应用程序目录遵循 Next.js App Router 约定:

҈ [语言环境]	
□ (营销)	
□ (受保护)	
□ 授权	
🗅 文档	
□ api	
宀 布局.tsx	
↑ 未找到.tsx	
🗅 robots.ts	
🗅 站点地图.ts	
☐ 清单.ts	

### /src/actions 目录

按功能组织的服务器操作,由 next-safe-action 提供支持,用于类型安全的服务器操作:

根目录结构
关键文件
关键目录
/src/app 目录
/src/lib 目录
/src/components 目录
代码组织最佳实践
按功能分组,而不是按类型分组
保持组件集中且小巧
使用桶出口
明确分离客户端和服务器组件
视频教程



#### /src/lib 目录

核心应用程序逻辑和实用程序:

□ 接权.ts
 □ 身份验证客户端.ts
 □ 流ts
 □ 元数据.ts
 □ 常量.ts
 □ 格式化程序.ts
 □ 实用程序.ts
 □ 网址
 □ 文档

#### /src/components 目录

组件按功能和用途进行组织,以提高可维护性:

基于特征的组织共享组件

## 代码组织最佳实践

使用 MkSaaS 时,请遵循以下代码组织的最佳实践:

- 1 **按功能分组,而不是按类型分组** 将相关文件放在一起,无论其类型如何。
- 2 **保持组件集中且小巧** 每个组件都应该具有单一职责。
- 3 使用桶出口

创建 index.ts 文件以从目录重新导出组件,使导入更清晰。

```
// components/ui/index.ts
export * from './Button';
export * from './Card';
export * from './Input';

// Usage
import { Button, Card, Input } from '@/components/ui';
```

4 明确分离客户端和服务器组件

在客户端组件的顶部使用 'use client' 指令,并将服务器组件保留为默认值。

# 视频教程



## 后续步骤

