

***** Focus GoFrame""GoFrame

-
-
- Context:
-



Content Menu

-
-
- - dao
- - model
- - shared
-
- - api
- - define
- - service
-

```
app
dao
model
shared
system
  admin
  internal
  index
    internal
      api
      define
      service
config
document
library
packed
public
template
upload
Dockerfile
go.mod
main.go
```

/		
app		
- dao		CURD
- model		
- shared		service
- system		
- index		index
- internal		
- api		//
- define		
- service		
config		
docker		Docker
document		Documentation:
library		

packed		Goboot
public		
template		
Dockerfile		Docker
go.mod		Go Module
main.go		

GoFrameapp



1.

- dao

daomodeldao

- model

modelmodelcaptchacontextview

- ▼ app
 - > dao
 - ▼ model
 - > internal
 - 👤 captcha.go
 - 👤 category.go
 - 👤 content.go
 - 👤 context.go
 - 👤 file.go
 - 👤 interact.go
 - 👤 menu.go
 - 👤 reply.go
 - 👤 session.go
 - 👤 setting.go
 - 👤 user.go
 - 👤 view.go
 - > shared
 - ▼ system

2. model

- shared

sharedserviceservicesharedcontext

4.

apidefineservice

- api

apiservice

- define

defineGoFrameModelmodel

- service

service

GoFramebootrouter

```

37 // 前台系统路由注册
38 s.Group("/", func(group *ghttp.RouterGroup) {
39     group.Middleware(service.Middleware.Ctx)
40     group.ALLMap(g.Map{
41         "/": api.Index, // 首页
42         "/login": api.Login, // 登录
43         "/register": api.Register, // 注册
44         "/category": api.Category, // 栏目
45         "/topic": api.Topic, // 主题
46         "/topic/id": api.Topic.Detail, // 主题 - 详情
47         "/ask": api.Ask, // 问答
48         "/ask/id": api.Ask.Detail, // 问答 - 详情
49         "/article": api.Article, // 文章
50         "/article/id": api.Article.Detail, // 文章 - 详情
51         "/reply": api.Reply, // 回复
52         "/search": api.Search, // 搜索
53         "/captcha": api.Captcha, // 验证码
54         "/user/id": api.User.Index, // 用户 - 主页
55     })
56 // 权限控制路由
57 group.Group("/", func(group *ghttp.RouterGroup) {
58     group.Middleware(service.Middleware.Auth)
59     group.ALLMap(g.Map{
60         "/user": api.User, // 用户
61         "/content": api.Content, // 内容
62         "/interact": api.Interact, // 交互
63         "/file": api.File, // 文件
64     })
65 })
66
67 Init()

```

5.

```

15 // 应用启动
16 func Run() {
17     // 绑定Swagger PLogin
18     s := g.Server()
19     s.Plugin(&swagger.Swagger{})
20 // 静态目录设置
21 uploadPath := g.Trig().GetString("upload.path")
22 if uploadPath == "" {
23     g.Log().Fatal("文件上传配置路径不能为空")
24 }
25 s.AddStaticPath("upload", uploadPath)
26 // http, 开发阶段禁止浏览器缓存, 方便调试
27 if gmode.IsDevEnv() {
28     s.BindHookHandler("/*", ghttp.HookBeforeServe, func(r *ghttp.Request) {
29         r.Response.Header().Set("Cache-Control", "no-store")
30     })
31 }
32 // 业务系统初始化
33 admin.Init()
34 index.Init()
35 // 启动Http Server
36 s.Run()
37
38
39
40
41

```

6.

