1，代码记录

按照大明老师的提示，在ginx的handler_func里写全函数：

```
func WrapReq[T any](fn func(ctx *gin.Context, req T, uc jwt.UserClaims) (Result,
error), l logger.LoggerV1, method string) gin.HandlerFunc {
 return func(ctx *gin.Context) {
  // 顺便把 userClaims 也取出来
  var req T
  var uc jwt.UserClaims
  if err := ctx.Bind(&req); err != nil {
   return
  }
  res, err := fn(ctx, req, uc)
  if err != nil {
   l.Error(method, logger.Field{
    Key:   "error",
    Value: err.Error(),
   }, logger.Field{
    Key:   "response",
    Value: res,
   })
  }
 }
}

type Result struct {
 // 这个叫做业务错误码
 Code int    `json:"code"`
 Msg  string `json:"msg"`
 Data any    `json:"data"`
}
```

在里面Bind req，同时回调业务函数，如果fn返回有错，则用传进来的logger打印出来。

user.go里调用，这里主要改的是SendLoginSMSCodeV1函数

```
func (u *UserHandler) SendLoginSMSCodeV1(ctx *gin.Context) {
 type Req struct {
  Phone string `json:"phone"`
 }

 fn := func(ctx *gin.Context, req Req, uc ijwt.UserClaims) (ginx.Result, error) {
  if req.Phone == "" {
   return ginx.Result{
    Code: 4,
    Msg:  "输入错误",
   }, errors.New("输入错误")
  }

  err := u.codeSvc.Send(ctx, biz, req.Phone)
  switch err {
  case nil:
   return ginx.Result{
    Code: 0,
    Msg:  "发送成功",
   }, nil
  case service.ErrCodeSendTooMany:
   return ginx.Result{
    Code: 5,
    Msg:  "发送太频繁，请稍后再试",
   }, errors.New("发送太频繁，请稍后再试")
  default:
   return ginx.Result{
    Code: 4,
    Msg:  "系统错误",
   }, errors.New("系统错误")
  }
 }

 ginx.WrapReq[Req](fn, u.l, "SendLoginSMSCodeV1")(ctx)
}
```

同时在UserHandler里传入一个logger。

```
type UserHandler struct {
 svc         service.UserService
 codeSvc     service.CodeService
 emailExp    *regexp.Regexp
 passwordExp *regexp.Regexp
 ijwt.Handler
 cmd redis.Cmdable
 l   logger.LoggerV1
}

func NewUserHandler(svc service.UserService,
 codeSvc service.CodeService, jwtHdl ijwt.Handler, l logger.LoggerV1) *UserHandler {
 const (
  emailRegexPattern    = "^\\w+([-+.]\\w+)*@\\w+([-.]\\w+)*\\.\\w+([-.]\\w+)*$"
  passwordRegexPattern = `^(?=.*[A-Za-z])(?=.*\d)(?=.*[$@$!%*#?&])[A-Za-z\d$@$!%*#?&]{8,}$`
 )
 emailExp := regexp.MustCompile(emailRegexPattern, regexp.None)
 passwordExp := regexp.MustCompile(passwordRegexPattern, regexp.None)
 return &UserHandler{
  svc:         svc,
  emailExp:    emailExp,
  passwordExp: passwordExp,
  codeSvc:     codeSvc,
  Handler:     jwtHdl,
  l:           l,
 }
}
```

重新生成wire_gen.go后运行。

2，测试

程序跑起来后发一个Send SMS请求，可以看到打出来一个ERROR信息，记录了发生错误的函数名称，和返回的Error和response。

```
[GIN-debug] Listening and serving HTTP on :8077
[791007]
[791007]
[791007]
2023-10-17T18:04:30.983+0800 ERROR logger/zap_logger.go:28 SendLoginSMSCodeV1 {"error": "系统错误", "response": {"code":4,"msg":"系统错
误","data":null}}
github.com/bolognagene/geektime-gocamp/geektime-gocamp/webook/webook/pkg/logger.
(*ZapLogger).Error
C:/Work/IT/GO/src/github.com/bolognagene/geektime-gocamp/geektime-
gocamp/webook/webook/pkg/logger/zap_logger.go:28
github.com/bolognagene/geektime-gocamp/geektime-
gocamp/webook/webook/pkg/ginx.WrapReq[...].func1
C:/Work/IT/GO/src/github.com/bolognagene/geektime-gocamp/geektime-
gocamp/webook/webook/pkg/ginx/handler_func.go:19
github.com/bolognagene/geektime-gocamp/geektime-gocamp/webook/webook/internal/web.
(*UserHandler).SendLoginSMSCodeV1
C:/Work/IT/GO/src/github.com/bolognagene/geektime-gocamp/geektime-
gocamp/webook/webook/internal/web/user.go:263
github.com/gin-gonic/gin.(*Context).Next
C:/Work/IT/GO/pkg/mod/github.com/gin-gonic/gin@v1.9.1/context.go:174
github.com/bolognagene/geektime-gocamp/geektime-
gocamp/webook/webook/pkg/ginx/middlewares/logger.(*MiddlewareBuilder).Build.func1
C:/Work/IT/GO/src/github.com/bolognagene/geektime-gocamp/geektime-
gocamp/webook/webook/pkg/ginx/middlewares/logger/builder.go:81
github.com/gin-gonic/gin.(*Context).Next
C:/Work/IT/GO/pkg/mod/github.com/gin-gonic/gin@v1.9.1/context.go:174
github.com/gin-gonic/gin.CustomRecoveryWithWriter.func1
C:/Work/IT/GO/pkg/mod/github.com/gin-gonic/gin@v1.9.1/recovery.go:102
github.com/gin-gonic/gin.(*Context).Next
C:/Work/IT/GO/pkg/mod/github.com/gin-gonic/gin@v1.9.1/context.go:174
github.com/gin-gonic/gin.LoggerWithConfig.func1
C:/Work/IT/GO/pkg/mod/github.com/gin-gonic/gin@v1.9.1/logger.go:240
github.com/gin-gonic/gin.(*Context).Next
C:/Work/IT/GO/pkg/mod/github.com/gin-gonic/gin@v1.9.1/context.go:174
github.com/gin-gonic/gin.(*Engine).handleHTTPRequest
C:/Work/IT/GO/pkg/mod/github.com/gin-gonic/gin@v1.9.1/gin.go:620
net/http.serverHandler.ServeHTTP
C:/Software/Real/GO/src/net/http/server.go:2936
net/http.(*conn).serve
C:/Software/Real/GO/src/net/http/server.go:2936
net/http.(*conn).serve
C:/Software/Real/GO/src/net/http/server.go:1995
2023-10-17T18:04:31.018+0800 DEBUG logger/zap_logger.go:16 HTTP请求 {"al":
{"Method":"POST","Url":"/users/login_sms/code/send","Durat
C:/Software/Real/GO/src/net/http/server.go:1995
2023-10-17T18:04:31.018+0800 DEBUG logger/zap_logger.go:16 HTTP请求 {"al":
```

```
{"Method":"POST","Url":"/users/login_sms/code/send","Durat
ion":"1m2.0894352s","ReqBody":"{\r\n "phone": "15817211905"\r\n}","RespBody":"","Status":0}}
[GIN] 2023/10/17 - 18:04:31 | 200 | 1m2s | ::1 | POST "/users/login_sms/code/send"

Debugger finished with the exit code 0
```