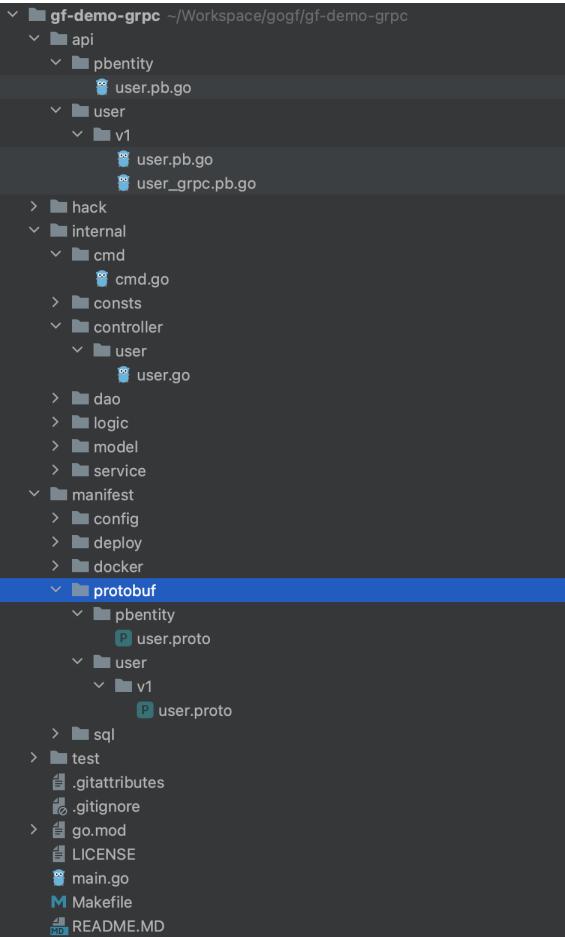


<https://github.com/gogf/gf-demo-grpc>

```
manifest  
/protobuf  
//xxx.  
proto v1  
/v2  
  
protobuf m  
anifest  
/pbentity  
  
protoapi  
  
gf gen  
pbentity  
/ make  
pbentity p  
rotobuf  
PB-gen  
pbentity  
  
proto  
  
gf gen  
pb /  
make pb pr  
oto  
  
proto
```



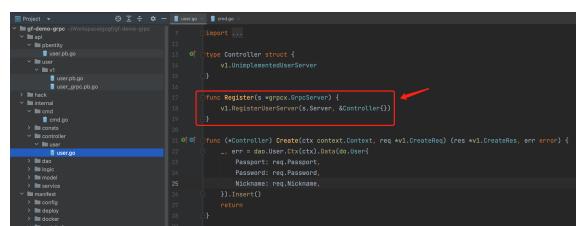
```
user.proto  
1 package user;  
2  
3 import "google/protobuf/timestamp.proto";  
4  
5 message User {  
6     string id = 1; //必填  
7     string name = 2; //必填  
8     string password = 3; //必填  
9     string nickname = 4; //必填  
10    google.protobuf.Timestamp created_at = 5; //必填  
11    google.protobuf.Timestamp updated_at = 6; //必填  
12 }  
13  
14 service UserService {  
15     option (io.grpc.ServiceConfig.http_status_code) = 200;  
16  
17     rpc Create(user.User) returns (User);  
18     rpc Update(user.User) returns (User);  
19     rpc Delete(string id) returns (User);  
20     rpc Get(string id) returns (User);  
21     rpc List(list.User) returns (User);  
22 }  
23  
24 message User {  
25     string id = 1; //必填  
26     string name = 2; //必填  
27     string password = 3; //必填  
28     string nickname = 4; //必填  
29 }  
30  
31 message User {  
32     string id = 1; //必填  
33     string name = 2; //必填  
34     string password = 3; //必填  
35     string nickname = 4; //必填  
36 }  
37  
38 message User {  
39     string id = 1; //必填  
40     string name = 2; //必填  
41     string password = 3; //必填  
42     string nickname = 4; //必填  
43 }  
44  
45 message User {  
46     string id = 1; //必填  
47     string name = 2; //必填  
48     string password = 3; //必填  
49     string nickname = 4; //必填  
50 }  
51  
52 message User {  
53     string id = 1; //必填  
54     string name = 2; //必填  
55     string password = 3; //必填  
56     string nickname = 4; //必填  
57 }  
58  
59 message User {  
60     string id = 1; //必填  
61     string name = 2; //必填  
62     string password = 3; //必填  
63     string nickname = 4; //必填  
64 }  
65  
66 message User {  
67     string id = 1; //必填  
68     string name = 2; //必填  
69     string password = 3; //必填  
70     string nickname = 4; //必填  
71 }  
72  
73 message User {  
74     string id = 1; //必填  
75     string name = 2; //必填  
76     string password = 3; //必填  
77     string nickname = 4; //必填  
78 }  
79  
80 message User {  
81     string id = 1; //必填  
82     string name = 2; //必填  
83     string password = 3; //必填  
84     string nickname = 4; //必填  
85 }  
86  
87 message User {  
88     string id = 1; //必填  
89     string name = 2; //必填  
90     string password = 3; //必填  
91     string nickname = 4; //必填  
92 }  
93  
94 message User {  
95     string id = 1; //必填  
96     string name = 2; //必填  
97     string password = 3; //必填  
98     string nickname = 4; //必填  
99 }
```

Content Menu

-
-
-
- proto
-
-
-
-

cmd<https://github.com/gogf/gf-demo-grpc/blob/main/internal/cmd/cmd.go>

protoRegister



```
1 package main  
2  
3 import "..."  
4  
5 func Register(s *grpc.Server) {  
6     if err := v1.RegisterUserServer(s, &Controller{}); err != nil {  
7         log.Fatalf("failed to register User server: %v", err)  
8     }  
9 }  
10  
11 type Controller struct {  
12     v1.UnimplementedUserServer  
13 }  
14  
15 func (Controller) Create(ctx context.Context, req *v1.CreateReq) (*v1.CreateRes, error) {  
16     user := NewUser()  
17     user.Name = req.Name  
18     user.Password = req.Password  
19     user.Nickname = req.Nickname  
20     user.Insert()  
21     return &v1.CreateRes{}, nil  
22 }  
23  
24 func (Controller) Update(ctx context.Context, req *v1.UpdateReq) (*v1.UpdateRes, error) {  
25     user := NewUser()  
26     user.Id = req.Id  
27     user.Name = req.Name  
28     user.Password = req.Password  
29     user.Nickname = req.Nickname  
30     user.Update()  
31     return &v1.UpdateRes{}, nil  
32 }  
33  
34 func (Controller) Delete(ctx context.Context, req *v1.DeleteReq) (*v1.DeleteRes, error) {  
35     user := NewUser()  
36     user.Id = req.Id  
37     user.Delete()  
38     return &v1.DeleteRes{}, nil  
39 }  
40  
41 func (Controller) Get(ctx context.Context, req *v1.GetReq) (*v1.GetUserRes, error) {  
42     user := NewUser()  
43     user.Id = req.Id  
44     user.Get()  
45     return &v1.GetUserRes{User: user}, nil  
46 }  
47  
48 func (Controller) List(ctx context.Context, req *v1.ListReq) (*v1.ListRes, error) {  
49     users := NewUserList()  
50     users.List()  
51     return &v1.ListRes{Users: users}, nil  
52 }
```

```

package cmd

import "github.com/gofrs/gofrframe"

func Main() {
    // Main is the main command.
    func(c *gofrframe.Context, parser *cmd.Parse) (err error) {
        c := gofrframe.NewConfig()
        c.Options = append(c.Options, []igrpc.ServerOption{
            gRPCServerOption(grpc.ServerUnaryValidate,
                grpcx.ServerUnaryValidate),
        })
        s := grpcx.Server.New(c)
        user.Register(s)
        s.Run()
        return nil
    }
}

```

gf gen pb/make pb prototodc xxx:yyy xxx

```

message CreateReq {
    string Passport = 1; // v: required
    string Password = 2; // v: required
    string Nickname = 3; // v: required
}

message CreateRes {}

message GetOneReq {
    uint64 Id = 1; // v: required
}

message GetOneRes {
    entity.User User = 1;
}

message GetListReq {
    int32 Page = 1;
    int32 Size = 2;
}

message GetListRes {
    repeated entity.User Users = 1;
}

message DeleteReq {
    uint64 Id = 1; // v: required
    string User = 2; // v: required
}

message DeleteRes {}

```

```

type CreateReq struct {
    State      protopb.MessageState
    Passport   protopb.String
    Password   protopb.String
    Nickname   protopb.String
}

func (req *CreateReq) ProtoMessage() interface{} {
    return req
}

func (req *CreateReq) GoString() string {
    return req.String()
}

func (req *CreateReq) String() string {
    return req.Passport
}

func (req *CreateReq) SetPassport(passport string) {
    req.Passport = passport
}

func (req *CreateReq) GetPassport() string {
    return req.Passport
}

func (req *CreateReq) SetPassword(password string) {
    req.Password = password
}

func (req *CreateReq) GetPassword() string {
    return req.Password
}

func (req *CreateReq) SetNickname(nickname string) {
    req.Nickname = nickname
}

func (req *CreateReq) GetNickname() string {
    return req.Nickname
}

```



GRPCHTTP

```

var (
    // Main is the main command.
    func(c *gofrframe.Context, parser *cmd.Parse) (err error) {
        c := gofrframe.NewConfig()
        c.Options = append(c.Options, []igrpc.ServerOption{
            gRPCServerOption(grpc.ServerUnaryValidate,
                grpc.ServerUnaryValidate),
        })
        s := grpcx.Server.New(c)
        user.Register(s)
        s.Run()
        return nil
    }
)

```

