Exercício 2

Cliente/Servidor em Go

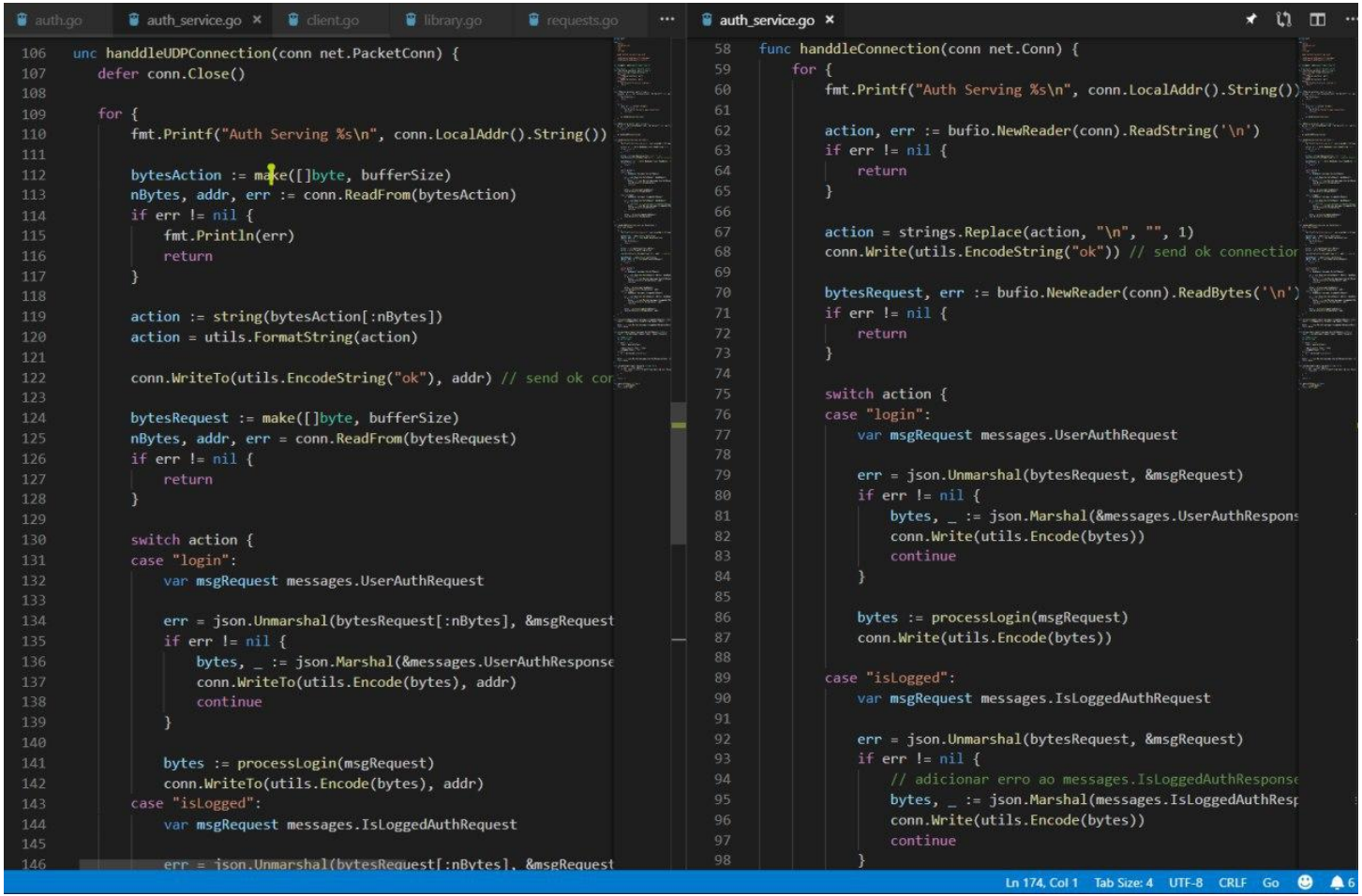
A aplicação – Biblioteca

- Servidor 1 (autenticação): realiza o *login* do cliente, gera um *token* para o mesmo que fica salvo e verifica se um determinado token é válido.

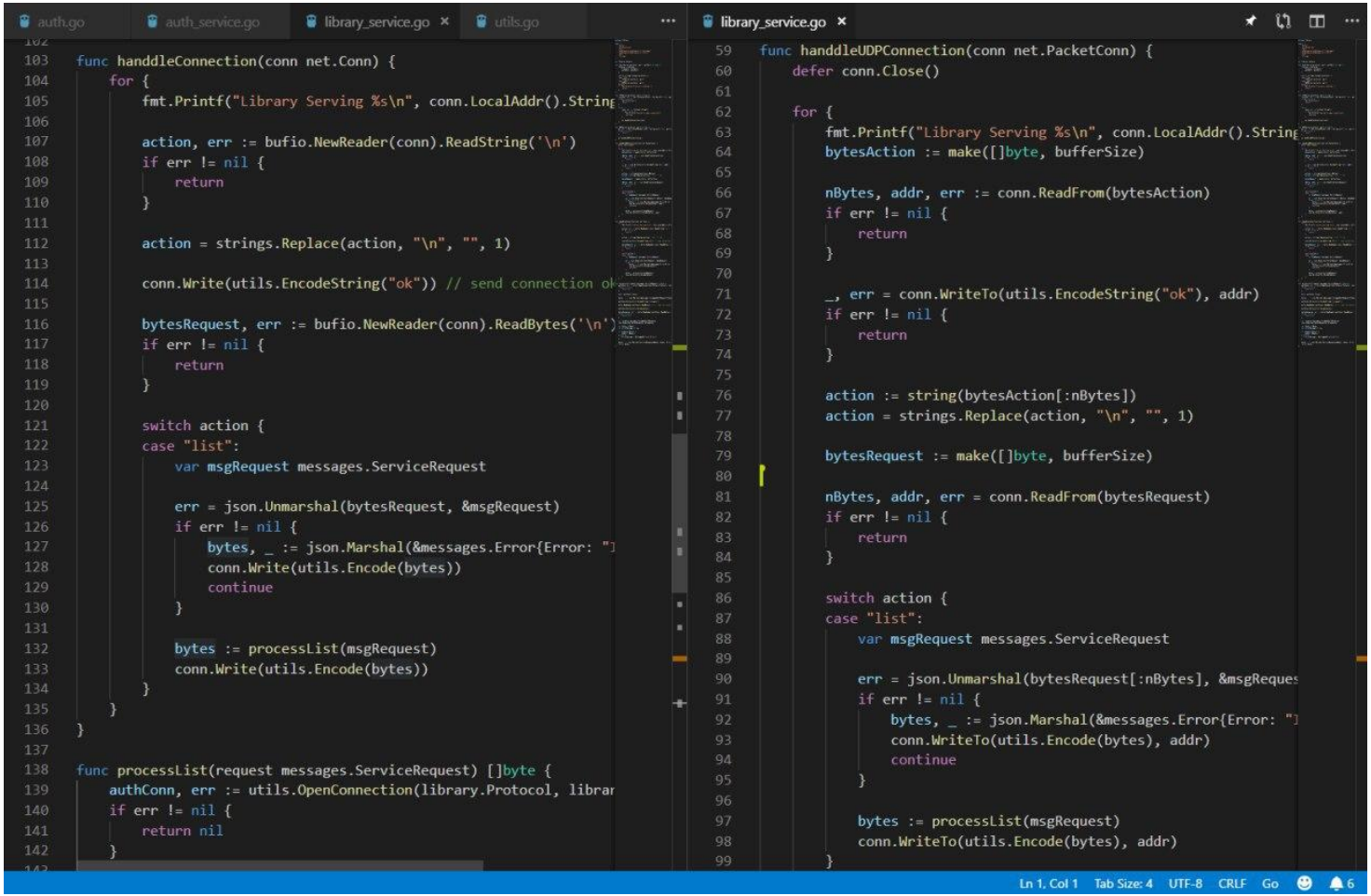
- Servidor 2 (biblioteca): retorna uma lista de livros.

- O servidor 2 se comunica com o 1 para validar o *token* do cliente para ele poder visualizar os livros.

auth


auth.go
auth_service.go
client.go
library.go
requests.go
unc handdleUDPConnection(conn net.PacketConn) {
defer conn.Close()
for {
fmt.Printf("Auth Serving %s\n", conn.LocalAddr().String())
bytesAction := make([]byte, bufferSize)
nBytes, addr, err := conn.ReadFrom(bytesAction)
if err != nil {
fmt.Println(err)
return
}
action := string(bytesAction[:nBytes])
action = utils.FormatString(action)
conn.WriteTo(utils.EncodeString("ok"), addr) // send ok cor
bytesRequest := make([]byte, bufferSize)
nBytes, addr, err = conn.ReadFrom(bytesRequest)
if err != nil {
return
}
switch action {
case "login":
var msgRequest messages.UserAuthRequest
err = json.Unmarshal(bytesRequest[:nBytes], &msgRequest
if err != nil {
bytes, _ := json.Marshal(&messages.UserAuthResponse
conn.WriteTo(utils.Encode(bytes), addr)
continue
}
bytes := processLogin(msgRequest)
conn.WriteTo(utils.Encode(bytes), addr)
case "isLogged":
var msgRequest messages.IsLoggedAuthRequest
err = json.Unmarshal(bytesRequest[:nBytes], &msgRequest
auth_service.go
func handdleConnection(conn net.Conn) {
for {
fmt.Printf("Auth Serving %s\n", conn.LocalAddr().String())
action, err := bufio.NewReader(conn).ReadString('\n')
if err != nil {
return
}
action = strings.Replace(action, "\n", "", 1)
conn.Write(utils.EncodeString("ok")) // send ok connectior
bytesRequest, err := bufio.NewReader(conn).ReadBytes('\n')
if err != nil {
return
}
switch action {
case "login":
var msgRequest messages.UserAuthRequest
err = json.Unmarshal(bytesRequest, &msgRequest)
if err != nil {
bytes, _ := json.Marshal(&messages.UserAuthRespons
conn.Write(utils.Encode(bytes))
continue
}
bytes := processLogin(msgRequest)
conn.Write(utils.Encode(bytes))
case "isLogged":
var msgRequest messages.IsLoggedAuthRequest
err = json.Unmarshal(bytesRequest, &msgRequest)
if err != nil {
// adicionar erro ao messages.IsLoggedAuthRespons
bytes, _ := json.Marshal(messages.IsLoggedAuthResp
conn.Write(utils.Encode(bytes))
continue
}
Ln 174, Col 1
Tab Size: 4
UTF-8
CRLF
Go

library


auth.go
auth_service.go
library_service.go
utils.go
func handleConnection(conn net.Conn) {
for {
fmt.Printf("Library Serving %s\n", conn.LocalAddr().String
action, err := bufio.NewReader(conn).ReadString('\n')
if err != nil {
return
}
action = strings.Replace(action, "\n", "", 1)
conn.Write(utils.EncodeString("ok")) // send connection ok
bytesRequest, err := bufio.NewReader(conn).ReadBytes('\n')
if err != nil {
return
}
switch action {
case "list":
var msgRequest messages.ServiceRequest
err = json.Unmarshal(bytesRequest, &msgRequest)
if err != nil {
bytes, _ := json.Marshal(&messages.Error{Error:
conn.Write(utils.Encode(bytes))
continue
}
bytes := processList(msgRequest)
conn.Write(utils.Encode(bytes))
}
}
}
func processList(request messages.ServiceRequest) []byte {
authConn, err := utils.OpenConnection(library.Protocol, librar
if err != nil {
return nil
}
library_service.go
func handdleUDPConnection(conn net.PacketConn) {
defer conn.Close()
for {
fmt.Printf("Library Serving %s\n", conn.LocalAddr().String
bytesAction := make([]byte, bufferSize)
nBytes, addr, err := conn.ReadFrom(bytesAction)
if err != nil {
return
}
_, err = conn.WriteTo(utils.EncodeString("ok"), addr)
if err != nil {
return
}
action := string(bytesAction[:nBytes])
action = strings.Replace(action, "\n", "", 1)
bytesRequest := make([]byte, bufferSize)
nBytes, addr, err = conn.ReadFrom(bytesRequest)
if err != nil {
return
}
switch action {
case "list":
var msgRequest messages.ServiceRequest
err = json.Unmarshal(bytesRequest[:nBytes], &msgReques
if err != nil {
bytes, _ := json.Marshal(&messages.Error{Error:
conn.WriteTo(utils.Encode(bytes), addr)
continue
}
bytes := processList(msgRequest)
conn.WriteTo(utils.Encode(bytes), addr)
}
Ln 1, Col 1
Tab Size: 4
UTF-8
CRLF
Go
6

main

auth.go | auth_service.go | library_service.go | main.go

```
14
15  var client c.Client
16  var user library.User
17
18  func main() {
19      protocol := os.Args[1]
20      authPort := os.Args[2]
21      servicePort := os.Args[3]
22
23      go auth.StartServer(protocol, authPort)
24      go library.StartServer(protocol, servicePort, authPort)
25
26      client = c.Client{
27          Protocol:    protocol,
28          AuthPort:    authPort,
29          ServicePort: servicePort,
30      }
31
32      reader := bufio.NewReader(os.Stdin)
33
34      for {
35          fmt.Println("Escolha uma das opções abaixo: ")
36          fmt.Println("1 - Login")
37          fmt.Println("2 - Listar livros")
38
39          option, _ := reader.ReadString('\n')
40          option = utils.FormatString(option)
41
42          switch option {
43          case "1":
44              login()
45          case "2":
46              books()
47          default:
48              fmt.Printf("%s é uma opção inválida.\n", option)
49          }
50      }
51  }
52
53  func login() {···
75  }
```

ok

```
PS C:\Users\1555 MX7\go\src\middleware2\middleware\src> go run .\main.go tcp 1234 2345
Escolha uma das opções abaixo:
1 - Login
2 - Listar livros
1
Login: edjan
Password: michiles
Auth Serving 127.0.0.1:1234
Auth Serving 127.0.0.1:1234
User logged in successfuly
Escolha uma das opções abaixo:
1 - Login
2 - Listar livros
2
Library Serving 127.0.0.1:2345
Auth Serving 127.0.0.1:1234
Auth Serving 127.0.0.1:1234
Library Serving 127.0.0.1:2345
[Name: Medicina Interna de Harrison - 2 Volumes
Description: Apresentando os extraordinários avanços ocorridos em todas as áreas da medicina, esta nova edição do Harrison foi amplamente revisada para oferecer uma
atualização completa sobre a patogênese das doenças, ensaios clínicos, técnicas de diagnóstico, diretrizes clínicas baseadas em evidências, tratamentos já estabeleci
dos e métodos recentemente aprovados
PublishDate: 21 nov 2016
Author:  Dennis L. Kasper, Stephen L. Hauser, J. Larry Jameson, Anthony S. Fauci, Dan L. Longo, Joseph Loscalzo
Categories: [Medicina Especialidades]

 Name: Netter Atlas de Anatomia Humana 7ª edição
Description: É um dos nomes mais fortes mundialmente na área de Anatomia, reconhecido pela didática e clareza de suas ilustrações. Figuras modernas, que, em um volum
e, apresenta todo o corpo humano em descrições detalhadas e clinicamente relevantes
PublishDate: 8 dez 2018
Author: Frank H. Netter
Categories: [Medicina Anatomia]

 Name: Cultura Inglesa. Go Beyond - Caixa com Worbook: Student's Pack With Worbook
Description: Go Beyond is an exciting 6-level American English course for teenagers learning English. The course covers CEFR levels A1+ through to B2, + all levels b
eing based on mapping of the requirements of the CEFR and international exams.
PublishDate: 2 out 2018
Author: Rebbeca Robb Benne
Categories: [Inglês e outras línguas Educação Didáticos]

]
```

user not found

```
PS C:\Users\1555 MX7\go\src\middleware2\middleware\src> go run .\main.go tcp 1234 2345
Escolha uma das opções abaixo:
1 - Login
2 - Listar livros
1
Login: xablau
Password: xablau
Auth Serving 127.0.0.1:1234
Auth Serving 127.0.0.1:1234
User not found
Escolha uma das opções abaixo:
1 - Login
2 - Listar livros
```

user not logged in