Netcentric lab 3
Nguyen Manh viet Khoi
ITCSIU21081

Client.go

```
package main

import (
    "fmt"
    "net"
    "os"
)

const (
    HOST = "localhost"
    PORT = "8080"
    TYPE = "tcp"
)

func main() {

    login := false
    // Resolve TCP address
    tcpServer, err := net.ResolveTCPAddr(TYPE,
HOST+":"+PORT)
    //
    if err != nil {
        println("ResolveTCPAddr failed:", err.Error())
        os.Exit(1)
    }

    conn, err := net.DialTCP(TYPE, nil, tcpServer)
```

```
    if err != nil {
        println("Dial failed:", err.Error())
        os.Exit(1)
    }

    defer conn.Close()

    for {
        if login {
            // buffer to get data
            received := make([]byte, 1024)
            str := string(received)
            _, err = conn.Read(received)

            // error
            if err != nil && err.Error() != "EOF" {
                println("Read data failed:", err.Error())
                os.Exit(1)
            }

            // print message
            if str != "" && err.Error() != "EOF" {
                fmt.Print("Received message: " +
string(received) + "\n")
            }
            if str == "start" {
                for {
                    // start the geussing game
                    fmt.Print("Guess the number: ")
                    var guess int
                    fmt.Scan(&guess)
```

```
                    _, err =
conn.Write([]byte(fmt.Sprintf("%d", guess)))
                    if err != nil {
                        println("Write data failed:",
err.Error())
                        os.Exit(1)
                    }
                    // buffer to get data
                    received := make([]byte, 1024)
                    _, err = conn.Read(received)
                    if err != nil {
                        println("Read data failed:",
err.Error())
                        os.Exit(1)
                    }
                    // print message
                    fmt.Print("Received message: " +
string(received) + "\n")
                    if string(received) == "correct" {
                        break
                    }
                }
            }
            // clear buffer
            str = ""
            received = make([]byte, 1024)

        } else {
            fmt.Print("1. Login\n2. Register\n3. Exit\n")
            var choice int
            fmt.Scan(&choice)
            switch choice {
```

```
            case 1:
                fmt.Print("Username: ")
                var username string
                fmt.Scan(&username)
                fmt.Print("Password: ")
                var password string
                fmt.Scan(&password)
                _, err = conn.Write([]byte("login:" +
username + ":" + password))
                if err != nil {
                    println("Write data failed:",
err.Error())
                    fmt.Printf("here\n")
                    os.Exit(1)
                }
                login = true
            case 2:
                fmt.Print("Username: ")
                var username string
                fmt.Scan(&username)
                fmt.Print("Password: ")
                var password string
                fmt.Scan(&password)
                _, err = conn.Write([]byte("register:" +
username + ":" + password))
                if err != nil {
                    println("Write data failed:",
err.Error())
                    os.Exit(1)
                }
            case 3:
                os.Exit(1)
```

```
            }

        }
    }

}
```

Server.go

```
package main

import (
    "encoding/json"
    "fmt"
    "io"
    "log"
    "math/rand"
    "net"
    "os"
    "strings"
    "time"
)

const (
    HOST = "localhost"
    PORT = "8080"
    TYPE = "tcp"
)

type User struct {
    Id       int      `json:"id"`
```

```
    Username string   `json:"username"`
    Password string   `json:"password"`
    Fullname string   `json:"fullname"`
    Email    []string `json:"email"`
    Address  []string `json:"address"`
}

func main() {

    listen, err := net.Listen(TYPE, HOST+":"+PORT)
    if err != nil {
        log.Fatal(err)
        os.Exit(1)
    }
    // close listener
    defer listen.Close()
    for {
        conn, err := listen.Accept()
        if err != nil {
            log.Fatal(err)
            os.Exit(1)
        }
        go handleRequest(conn)
    }
}

func handleRequest(conn net.Conn) {
    // incoming request
    buffer := make([]byte, 1024)
    _, err := conn.Read(buffer)
    if err != nil {
        log.Fatal(err)
```

```
    }

    // print message
    fmt.Println("Received message:", string(buffer))

    mess := string(buffer)

    if mess[:4] == "exit" {
        fmt.Println("exit")
        _, err = conn.Write([]byte("exit success"))
        os.Exit(1)
    } else if mess[:5] == "login" {

        // read user.json and check if user exists
        jsonFile, err := os.Open("user.json")

        // check if file exists
        if err != nil {
            fmt.Println(err)
        }
        byteValue, _ := io.ReadAll(jsonFile)
        var users []User

        json.Unmarshal(byteValue, &users)

        // extract using plsit ":"
        data := strings.Split(mess, ":")

        for _, user := range users {
            username := strings.TrimSpace(data[1])
            password := strings.TrimSpace(data[2])
            // check if username and password match
```

```
            if user.Username == username && user.Password
== password {
                fmt.Println("login success")
                _, err = conn.Write([]byte("login
success"))
                _, err = conn.Write([]byte("Welcome " +
user.Fullname + "\n"))
                    if err != nil {
                        log.Fatal(err)
                    }
                // send message
                _, err = conn.Write([]byte("Let's start
the game! "))
                if err != nil {
                    log.Fatal(err)
                }
                // send start message
                _, err = conn.Write([]byte("start"))
                if err != nil {
                    log.Fatal(err)
                }
                // random a number
                rand.Seed(time.Now().UnixNano())
                number := rand.Intn(100)
                for {
                    // send message
                    _, err = conn.Write([]byte("Enter your
guess: "))
                    if err != nil {
                        log.Fatal(err)
                    }
                    // buffer to get data
```

```
                    received := make([]byte, 1024)
                    _, err = conn.Read(received)
                    if err != nil {
                        log.Fatal(err)
                    }
                    // convert to string
                    str := string(received)
                    // convert to int
                    guess := 0
                    fmt.Sscanf(str, "%d", &guess)
                    // check if guess is correct
                    if guess == number {
                        _, err =
conn.Write([]byte("Congratulations! You got it! "))
                        if err != nil {
                            log.Fatal(err)
                        }
                        break
                    } else if guess < number {
                        _, err = conn.Write([]byte("Too
low! "))
                        if err != nil {
                            log.Fatal(err)
                        }
                    } else {
                        _, err = conn.Write([]byte("Too
high! "))
                        if err != nil {
                            log.Fatal(err)
                        }
                    }
```

```
                }
            }
        }



        if err != nil {
            log.Fatal(err)
        }
    } else if mess[:8] == "register" {
        fmt.Println("register")
        _, err = conn.Write([]byte("register success"))
        if err != nil {
            log.Fatal(err)
        }
    } else {
        fmt.Println("invalid")
        _, err = conn.Write([]byte("invalid"))
        if err != nil {
            log.Fatal(err)
        }
    }

    // close conn
    conn.Close()
}
```

User.json

```
[
    {
```

```
        "id": 1,
        "username": "admin",
        "password": "admin",
        "fullname": "Administrator",
        "email": [
            "admin@email.com",
            "ad@email.com"
        ],
        "address": [
            "Jl. Jend. Sudirman",
            "Jl. Jend. Gatot Subroto"
        ]
    },
    {
        "id": 2,
        "username": "baby",
        "password": "user",
        "fullname": "User",
        "email": [
            "mini@gmail.com",
            "max@gmail.com"
        ],
        "address": [
            "Jl. Jend. Sudirman",
            "Jl. Jend. Gatot Subroto"
        ]
    }
]
```


Output

- In this code, I use user.json to store the user info and the server will load it to the memory.
- To play the game client must login and pass the authentication section.
-
-