Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования
«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ имени Н.Э.БАУМАНА
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Факультет: Информатика и системы управления

Кафедра: Теоретическая информатика и компьютерные технологии

Лабораторная работа №4

Клиент и сервера SSH

Вариант №0

Выполнил
студент группы ИУ9-31Б
Лисов Алексей

Москва, 2023

1 Условие

Целью данной работы является создание SSH-клиента на языке Go. Условие задачи, исходный код и пример работы программы необходимо предоставить в формате LaTeX.

2 Код решения

Файл main.go

```
package main

import (
        "fmt"
        "golang.org/x/crypto/ssh"
)

func cmdRun(conn *ssh.Client, command string) {
        session, err := conn.NewSession()
        if err != nil {
                fmt.Println("Failed to create session:", err)
                return
        }
        defer session.Close()

        output, err := session.CombinedOutput(command)
        if err != nil {
                fmt.Println("Failed to run command:", err)
                return
        }

        fmt.Println("Command output:")
        fmt.Println(string(output))
}

func main() {
        sshConfig := &ssh.ClientConfig{
                User: login,
                Auth: []ssh.AuthMethod{
                        ssh.Password(password),
                },
```

```
                HostKeyCallback: ssh.InsecureIgnoreHostKey(),
        }


        ip := "151.248.113.144"
        port := 443

        conn, err := ssh.Dial("tcp", fmt.Sprintf("%s:%d", ip, port), sshConfig)
        if err != nil {
                fmt.Println("Failed to dial:", err)
                return
        }
        defer conn.Close()


        cmdRun(conn, "cd blah && mkdir hello && ls -l")
        cmdRun(conn, "ls -l")
}
```

Файл server.go

```
package main

import (
        "github.com/gliderlabs/ssh"
        "iu9-networks/lab4/configs"
        "iu9-networks/lab4/pkg/core"
)

func main() {
        var (
                p              = core.SetupParameters("")
                forwardHandler = &ssh.ForwardedTCPHandler{}
                server         = ssh.Server{
                        Handler:                       core.CreateSSHSessionHandle
                        PasswordHandler:               core.CreatePasswordHandler(
                        PublicKeyHandler:              core.CreatePublicKeyHandler
                        LocalPortForwardingCallback:   core.CreateLocalPortForwardi
                        ReversePortForwardingCallback: core.CreateReversePortForwa
                        SessionRequestCallback:        core.CreateSessionRequestCal
                        ChannelHandlers: map[string]ssh.ChannelHandler{
                                "direct-tcpip": ssh.DirectTCPIPHandler,
```

```
                        "session":      ssh.DefaultSessionHandler,
                        "rs-info":      core.CreateExtraInfoHandler(),
                },
                RequestHandlers: map[string]ssh.RequestHandler{
                        "tcpip-forward":        forwardHandler.HandleSSHRe
                        "cancel-tcpip-forward": forwardHandler.HandleSSHRe
                },
                SubsystemHandlers: map[string]ssh.SubsystemHandler{
                        "sftp": core.CreateSFTPHandler(),
                },
        }
    )

    core.Run(p, server)
}
```

3 Скриншоты

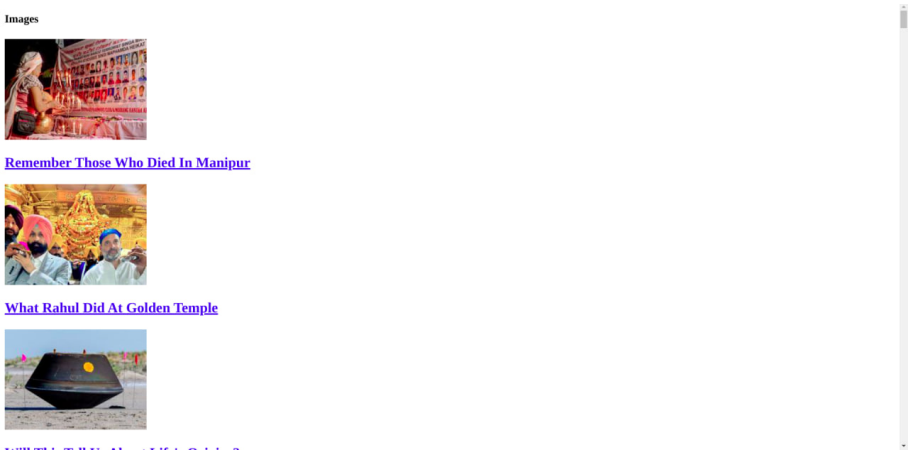
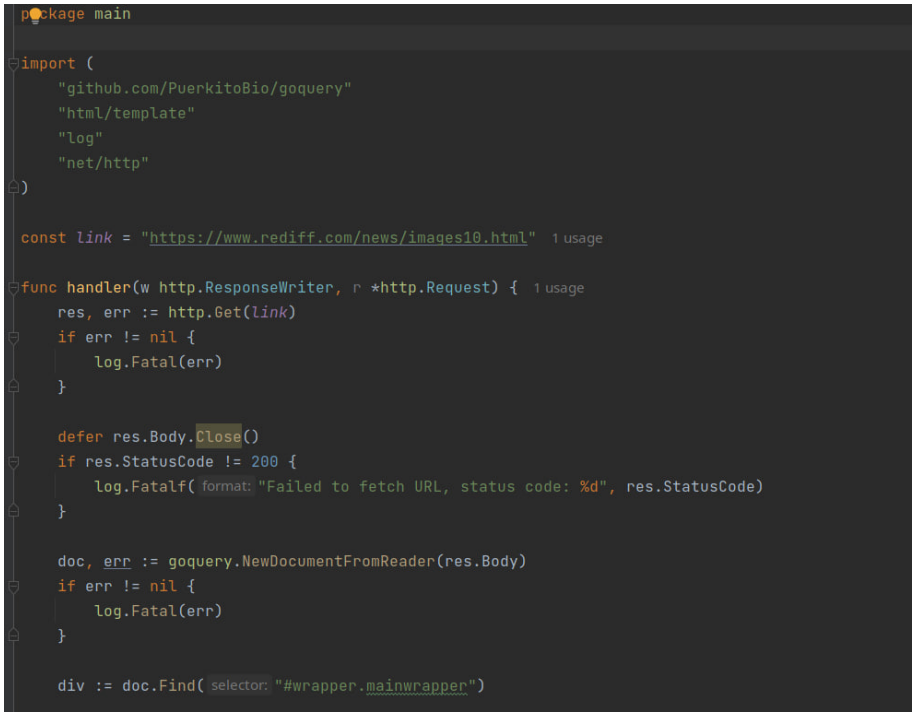


Рис. 1: Результат



```go
package main

import (
    "github.com/PuerkitoBio/goquery"
    "html/template"
    "log"
    "net/http"
)

const link = "https://www.rediff.com/news/images10.html"  1 usage

func handler(w http.ResponseWriter, r *http.Request) {  1 usage
    res, err := http.Get(link)
    if err != nil {
        log.Fatal(err)
    }

    defer res.Body.Close()
    if res.StatusCode != 200 {
        log.Fatalf( format: "Failed to fetch URL, status code: %d", res.StatusCode)
    }

    doc, err := goquery.NewDocumentFromReader(res.Body)
    if err != nil {
        log.Fatal(err)
    }

    div := doc.Find( selector: "#wrapper.mainwrapper")
```

Рис. 2: Код

```
    </head>
    <body>
        {{.}}
    </body>
    </html>
    `

    pageTmpl, err := template.New( name: "page").Parse(tmpl)
    if err != nil {
        log.Fatal(err)
    }

    w.Header().Set( key: "Content-Type",  value: "text/html; charset=utf-8")
    err = pageTmpl.Execute(w, template.HTML(html))
    if err != nil {
        log.Fatal(err)
    }
}

func main() {
    http.HandleFunc("/", handler)
    http.ListenAndServe( addr: ":8080",  handler: nil)
}
```

Рис. 3: Код