



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 № 6.1

по курсу «Компьютерные сети»

«Разработка SMTP-клиента и приложения почтовой рассылки»

Студент группы ИУ9-32Б Волохов А. В.

Преподаватель Посевин Д. П.

Москва 2023

1 Задание

Рассматривается задача разработки smtp-клиента на языке Golang.

Используя пакеты <https://pkg.go.dev/net/smtp>, <https://pkg.go.dev/crypto/tls>, а также в зависимости от реализации возможно понадобится <https://pkg.go.dev/strings>. Необходимо реализовать задачи приведенные ниже.

Задача 1: SMTP-клиент на Golang. Необходимо реализовать программу отправки проверочного SMTP сообщения, которое необходимо производить на ящик danila@bmstu.posevin.ru с одного из электронных ящиков, доступ к которым приведен ниже. При этом работоспособность приложения необходимо продемонстрировать очно. В этом приложении должны быть реализованы следующие функции: ввод значения поля To из командной строки; ввод значения поля Subject из командной строки; ввод сообщения в поле Message body из командной строки. При отправке проверочного сообщения необходимо в теме сообщения обязательно указать фамилию, имя и группу студента выполнившего задание.

Задача 2: Приложение почтовых рассылок.

1. В базе данных MySQL создать таблицу рассылки, в которой должны быть следующие поля: имя пользователя, адрес электронной почты пользователя, сообщение пользователю.

2. Реализовать рассылку по таблице рассылки, при этом в тексте письма должно быть реализовано персональное обращение к пользователю, текст письма оформлен в HTML формате, при этом как минимум приветствие должно быть выделено жирным шрифтом, текст письма курсивом и фон письма отличаться от белого, рекомендуется прочитать статьи приведенные ниже о верстке электронных писем для рассылок.

3. Необходимо реализовать логирование ответов от smtp сервера в момент сеанса связи в таблицу MySQL.

4. Работоспособность приложения необходимо продемонстрировать очно, поместив в список рассылки ящики danila@posevin.com, posevin@mail.ru, posevin@bmstu.ru. Исходный код программы представлен в листингах 1– 2– 4.

Листинг 1 — client.go

```

1 package main
2 import (
3     "bufio"
4     "crypto/tls"
5     "fmt"
6     "net/smtp"
7     "os"
8     "strings"
9 )
10 func main() {
11     fmt.Print("Enter address (To): ")
12     to := readInput()
13     fmt.Print("Enter Subject: ")
14     subject := readInput()
15     fmt.Print("Enter Message body: ")
16     messageBody := readInput()
17     username := "dts21@dactyl.su"
18     password := "12345678990DactylSUDTS"
19     smtpServer := "mail.nic.ru"
20     smtpPort := 465
21     useSSL := true
22     messageSubject := fmt.Sprintf("%s", subject)
23     message := fmt.Sprintf("To: %s\r\nSubject: %s\r\n\r\n%s", to,
24         messageSubject, messageBody)
25     tlsConfig := &tls.Config{InsecureSkipVerify: !useSSL, ServerName:
26         smtpServer,}
27     conn, err := tls.Dial("tcp", fmt.Sprintf("%s:%d", smtpServer, smtpPort),
28         tlsConfig)
29     if err != nil { fmt.Println("Error", err) return }
30     defer conn.Close()
31     auth := smtp.PlainAuth("", username, password, smtpServer)
32     client, err := smtp.NewClient(conn, smtpServer)
33     if err != nil {fmt.Println("Error", err) return}
34     defer client.Quit()
35     if err := client.Auth(auth); err != nil {fmt.Println("Error", err)
36         return}
37     if err := client.Mail(username); err != nil {fmt.Println("Error", err)
38         return}
39     if err := client.Rcpt(to); err != nil {fmt.Println("Error", err)
40         return}
41     writer, err := client.Data()
42     if err != nil {
43         fmt.Println("Error", err)
44         return
45     }
46     defer writer.Close()
47     _, err = writer.Write([]byte(message))
48     if err != nil {
49         fmt.Println("Error", err)
50         return
51     }
52     fmt.Println("Success.")
53 }
54 func readInput() string {
55     reader := bufio.NewReader(os.Stdin)
56     input, _ := reader.ReadString('\n')
57     return strings.TrimSpace(input)
58 }

```

Листинг 2 — app.go

```

1 package main
2
3 import (
4     "crypto/tls"
5     "database/sql"
6     "fmt"
7     _ "github.com/go-sql-driver/mysql"
8     "log"
9     "net/smtp"
10 )
11
12 type SMTPSettings struct {
13     Server    string
14     Port      string
15     Username  string
16     Password  string
17 }
18
19 type DBSettings struct {
20     Host      string
21     Database  string
22     Username  string
23     Password  string
24 }
25
26 type Mail struct {
27     To        string
28     Subject   string
29     Body      string
30 }
31
32 type Log struct {
33     To        string
34     Status    string
35     Description string
36 }
37
38 func main() {
39     dbSettings := DBSettings{
40         Host:      "students.yss.su",
41         Database:  "iu9networkslabs",
42         Username:  "iu9networkslabs",
43         Password:  "Je2dTYr6",
44     }
45
46     smtpSettings := SMTPSettings{
47         Server:    "mail.nic.ru",
48         Port:      "465",
49         Username:  "dts21@dactyl.su",
50         Password:  "12345678990DactylSUDTS",
51     }
52
53     db, err := sql.Open("mysql", fmt.Sprintf("%s:%s@tcp(%s)/%s",
54         dbSettings.Username, dbSettings.Password, dbSettings.Host,
55         dbSettings.Database))
56     if err != nil {
57         log.Fatal(err)
58     }
59     defer db.Close()

```

Листинг 3 — app.go - продолжение

```

1 query := "SELECT username, email, message FROM Volokhov_smtp"
2 rows, err := db.Query(query)
3 if err != nil {
4     log.Fatal(err)
5 }
6 defer rows.Close()
7
8 for rows.Next() {
9     var username, email, message string
10    if err := rows.Scan(&username, &email, &message); err != nil {
11        log.Println(err)
12        continue
13    }
14
15    htmlBody := fmt.Sprintf(<html><body><p><strong>%s</strong></p><p><em>%s</em></p><p>%s</p></body></html>', username, message, message)
16
17    err := sendMail(smtpSettings, Mail{
18        To:      email,
19        Subject: "Volokhov Aleksandr IU9-32B",
20        Body:    htmlBody,
21    }, db)
22    if err != nil {
23        log.Println(err)
24        continue
25    }
26
27    log.Printf("Mail sent to %s\n", email)
28 }
29 }
30
31 func sendMail(smtpSettings SMTPSettings, mail Mail, db *sql.DB) error {
32     message := fmt.Sprintf("To: %s\r\nSubject: %s\r\nContent-Type: text/html\r\n\r\n%s", mail.To, mail.Subject, mail.Body)
33
34     auth := smtp.PlainAuth("", smtpSettings.Username, smtpSettings.Password, smtpSettings.Server)
35
36     tlsConfig := &tls.Config{
37         ServerName: smtpSettings.Server,
38     }
39
40     conn, err := tls.Dial("tcp", fmt.Sprintf("%s:%s", smtpSettings.Server, smtpSettings.Port), tlsConfig)
41     if err != nil {
42         return err
43     }
44     defer conn.Close()
45
46     client, err := smtp.NewClient(conn, smtpSettings.Server)
47     if err != nil {
48         return err
49     }
50     defer client.Close()
51
52     if err := client.Auth(auth); err != nil {
53         return err
54     }

```

Листинг 4 — app.go - продолжение

```
1 if err := client.Mail(smtpSettings.Username); err != nil {
2     return err
3 }
4
5 if err := client.Rcpt(mail.To); err != nil {
6     return err
7 }
8
9 w, err := client.Data()
10 if err != nil {
11     return err
12 }
13 defer w.Close()
14
15 _, err = w.Write([]byte(message))
16 if err != nil {
17     return err
18 }
19
20 log := Log{
21     To:      mail.To,
22     Status:   "success",
23     Description: "Email sent successfully",
24 }
25 if err := insertLog(db, log); err != nil {
26     fmt.Println("Failed to insert log into the database:", err)
27 }
28
29 return nil
30 }
31
32 func insertLog(db *sql.DB, log Log) error {
33     _, err := db.Exec("INSERT INTO Volokhov_logs (to_email, status,
34         description) VALUES (?, ?, ?)", log.To, log.Status, log.Description)
35     return err
36 }
```



Рис. 1 — Клиент и письмо

Volokhov Aleksandr IU9-32B



dts21@dactyl.su dts21@dactyl.su 20 декабря в 15:19

Я >

Папка: Входящие

Язык письма — авто. Перевести на русский?

user4,

Hello, user4!

Hello, user4!

Рис. 2 — Пример письма из рассылки

MySQL » Сервер » iu9networkslabs » Выбрать: Volokhov_smtp

Выбрать: Volokhov_smtp

Выбрать

Показать структуру

Изменить таблицу

Новая запись

Выбрать

Поиск

Сортировать

Лимит
50

Длина текста
100

Действие
Выбрать

SELECT * FROM `Volokhov_smtp` LIMIT 50 (0.000 s) Редактировать

| <input type="checkbox"/> Изменить | username | email | message |
|--|----------|---------------------|------------------------------------|
| <input type="checkbox"/> редактировать | user1 | danila@posevin.com | Hello, user1! It's a test message. |
| <input type="checkbox"/> редактировать | user2 | posevin@mail.ru | Hello, user2! It's a test message. |
| <input type="checkbox"/> редактировать | user3 | posevin@bmstu.ru | Hello, user3! It's a test message. |
| <input type="checkbox"/> редактировать | user4 | avolohoff@yandex.ru | Hello, user4! |

Рис. 3 — База данных рассылки

MySQL » Сервер » iu9networkslabs » Выбрать: Volokhov_logs

Выбрать: Volokhov_logs

Выбрать Показать структуру Изменить таблицу Новая запись

| | | | | | |
|--------------------------------------|------------------------------------|--|------------------------------------|---|--|
| <input type="text" value="Выбрать"/> | <input type="text" value="Поиск"/> | <input type="text" value="Сортировать"/> | <input type="text" value="Лимит"/> | <input type="text" value="Длина текста"/> | <input type="text" value="Действие"/> |
| | | | <input type="text" value="50"/> | <input type="text" value="100"/> | <input type="button" value="Выбрать"/> |

SELECT * FROM `Volokhov_logs` LIMIT 50 (0.000 s) [Редактировать](#)

| <input type="checkbox"/> Изменить | to_email | status | description |
|--|---------------------|---------|-------------------------|
| <input type="checkbox"/> редактировать | danila@posevin.com | error | EOF |
| <input type="checkbox"/> редактировать | danila@posevin.com | success | Email sent successfully |
| <input type="checkbox"/> редактировать | posevin@mail.ru | success | Email sent successfully |
| <input type="checkbox"/> редактировать | posevin@bmstu.ru | success | Email sent successfully |
| <input type="checkbox"/> редактировать | danila@posevin.com | success | Email sent successfully |
| <input type="checkbox"/> редактировать | posevin@mail.ru | success | Email sent successfully |
| <input type="checkbox"/> редактировать | posevin@bmstu.ru | success | Email sent successfully |
| <input type="checkbox"/> редактировать | avolohoff@yandex.ru | success | Email sent successfully |

Рис. 4 — База данных логов