**Ministerul Educaţiei al Republicii Moldova**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică şi Microelectronică**

**Raport**

Lucrare de laborator **№4**

Disciplina: Programarea in Retea

A îndeplinit: st.gr.FI-161 Panfil Nicolae

A controlat: Gavrilita Mihai

Chișinău 2019

**Lucrare de Laborator Nr.4**

**Scop:**

**1. Ințelegerea protocolului SMTP de transfer mesaje electronice și metodele de bază;**

**2. Inițializarea proprietăților pentru protocolul SMTP;**

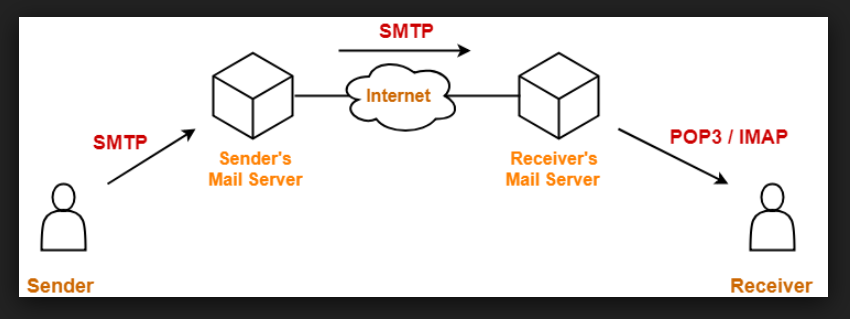
**3. Utilizarea contului de mail real / serverului fals pentru testarea funcțiilor de trimitere mail;**

**4. Concepte de bază ale protocolului SMTP / POP3.**

**Sarcina:**

Implementarea unui client de Mail la trimiterea și preluarea mesajelor din contul de e-mail. Clientul ar trebui să utilizeze protocoalele SMTP / POP3.

În această lucrare de laborator a trebuit să creez Mail Client, care va trimite și primi mesaje e-mail. În funcție de sarcina dată, ar trebui să folosesc protocoalele SMTP și POP3. **SMTP** - serverele de poștă electronică și alți agenți de transfer de e-mail utilizează SMTP pentru a trimite și a primi mesaje de poștă electronică pe portul TCP 25. Aplicațiile poștale client la nivel de utilizator folosesc în mod obișnuit numai SMTP pentru trimiterea mesajelor la un server de mail pentru retransmitere. Pentru aceasta, clienții de poștă electronică trimit în mod obișnuit e-mailurile de ieșire către un server de mail de pe portul 587 sau 465.

**POP3** - pentru recuperarea mesajelor IMAP și POP3 sunt standarde, dar serverele frecvent utilizează propriul sistem de recuperare.Deși majoritatea clienților POP3 au opțiunea de a părăsi poșta pe server după descărcare, în general se conectează, recuperează toate mesajele, le stochează pe client sistem și le șterge de pe server. Pe de altă parte, Internet Message Access Protocol (IMAP), în mod normal, lasă toate mesajele de pe server. [](https://user-images.githubusercontent.com/43058513/56412195-13e18100-628c-11e9-8eae-9f50b91b38b3.PNG)

Pentru realizarea trimiterii mesajelor trebuie să deschidem o sesiune și să obținem instanța proprietăților (configurațiilor) de trimitere. Apoi, am creat transportul variabil de tip SMTPTransport pentru conectarea la postul Gmail și trimiterea către receptor a acreditărilor și mesajului cu toate părțile (header, message body, attachments).

public static void main(String[] args) {

Session session = Session.getInstance(GetProperties(), null);

try {

Message msg = MessageInitialize(session);

SMTPTransport transport = (SMTPTransport)session.getTransport("smtps");

transport.connect("smtp.gmail.com", RetrieveEmails.USERNAME, RetrieveEmails.PASSWORD);

transport.sendMessage(msg, msg.getAllRecipients());

System.out.println("Response: " + transport.getLastServerResponse());

transport.close();

} catch (MessagingException e) {

e.printStackTrace();

}

}

În funcția MessageInitiliaze am efectuat mesajul MIME (Multi-Purpose Internet Mail Extensions) și inițializat părțile mesajului, unde am adăugat un fișier .txt PR-File.txt și o imagine utm.png. O singură întrebare poate să apară: în cazul în care toate părțile mesajului sunt stocate și trimise? Răspunsul e evident - avem Multipart inițializat in beggining , care este un container care deține mai multe părți din body.

Următoarea funcție în procesul de trimitere a e-mailului este: Attachments (), unde această funcție inițializează o parte abstractă care conține content-ul.

private static void Attachments(String path, String contentId) throws MessagingException {

BodyPart messageBodyPart = new MimeBodyPart();

DataSource fds = new FileDataSource(path);

messageBodyPart.setDataHandler(new DataHandler(fds));

messageBodyPart.setHeader("Content-ID", contentId);

messageBodyPart.setFileName(path);

multipart.addBodyPart(messageBodyPart);

}

Funcția BodyParts() este creată pentru adăugarea content-ului la message.

private static void BodyParts(String contentText, String contentType) throws MessagingException {

BodyPart messageBodyPart = new MimeBodyPart();

messageBodyPart.setContent(contentText, contentType);

multipart.addBodyPart(messageBodyPart);

}

Ultima funcție este pentru proprietăți, care sunt pentru configurație: host , auth...

private static Properties GetProperties(){

Properties props = System.getProperties();

props.put("mail.smtps.host","smtp.gmail.com");

props.put("mail.smtps.auth","true");

props.put("mail.smtp.EnableSSL.enable","true");

return props;

}

Pentru extragerea mesajelor am un alt package, care are clasa RetrieveEmiles, care preia mesajele din Inbox Folder din post. Aici, am Store provider, care se conectează la store prin POP3. Apoi, am primit Inbox Folder și am citit mesajele primite. Message array este creat pentru a stoca mesajele primite și pentru a le parsa în structura corespunzătoare. In continuare am 2 funcții care sunt pentru parsarea informațiilor din e-mail.

1. **getTextFromMessage** - getContent() din Message part.
2. **getTextFromMimeMultipart** - parsarea HTML și obținerea textului din body.

// get the text of the message in plain mode in cosole.

private static String getTextFromMessage(Message message) throws Exception {

String result = "";

if (message.isMimeType("text/plain")) {

result = message.getContent().toString();

} else if (message.isMimeType("multipart/\*")) {

MimeMultipart mimeMultipart = (MimeMultipart) message.getContent();

result = getTextFromMimeMultipart(mimeMultipart);

}

return result;

}

// parsing the parts of the message (the Bodyparts)

private static String getTextFromMimeMultipart(MimeMultipart mimeMultipart) throws Exception {

String result = "\n";

int count = mimeMultipart.getCount();

for (int i = 0; i < count; i++) {

BodyPart bodyPart = mimeMultipart.getBodyPart(i);

if (bodyPart.isMimeType("text/plain")) {

result += bodyPart.getContent();

break;

} else if (bodyPart.isMimeType("text/html")) {

String html = (String) bodyPart.getContent();

result += Jsoup.parse(html).text();

} else if (bodyPart.getContent() instanceof MimeMultipart){

result += getTextFromMimeMultipart((MimeMultipart)bodyPart.getContent());

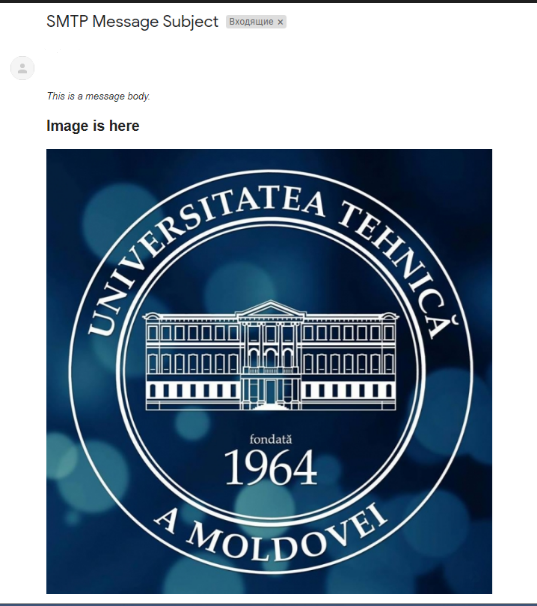
}

}

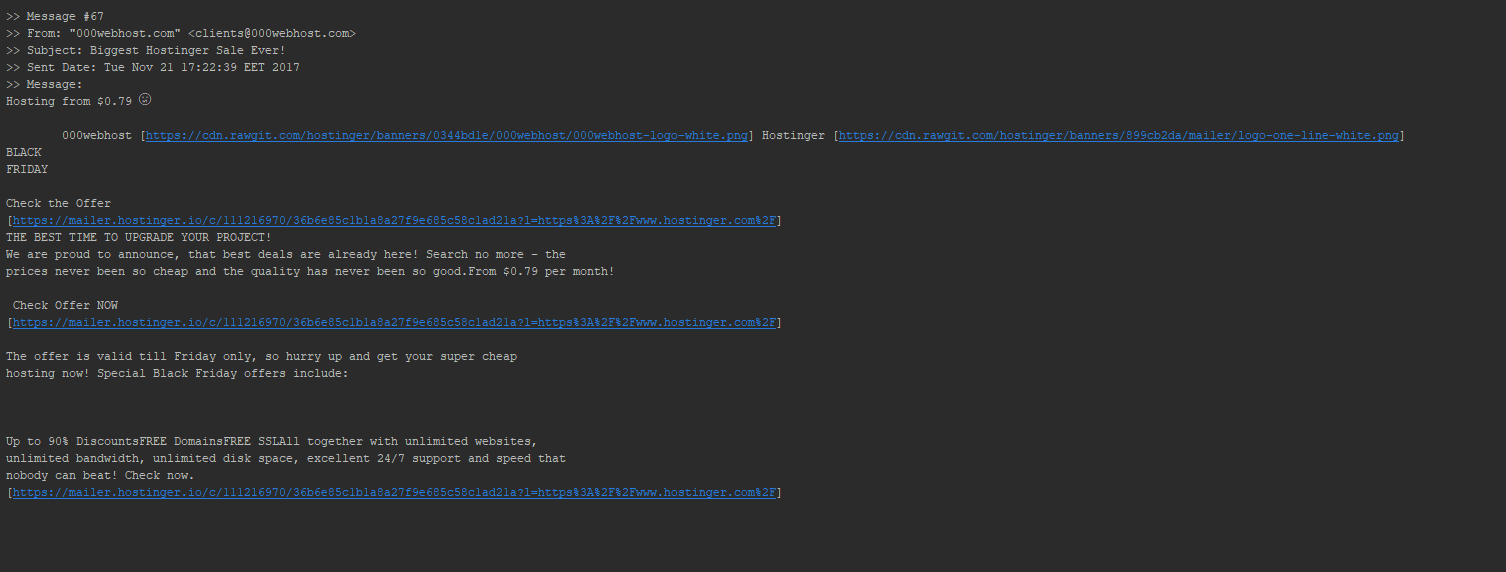
return result;

}

**Output:**

[](https://user-images.githubusercontent.com/43058513/56415377-3e383c00-6296-11e9-985b-a59495e5a305.png)

[](https://user-images.githubusercontent.com/43058513/56415417-5c05a100-6296-11e9-8a27-a74ce7f24087.png)

[](https://user-images.githubusercontent.com/43058513/56415469-763f7f00-6296-11e9-9f39-52ac47c824bd.png)

**Concluzii:**

În această lucrare de laborator am obținut abilitățile de a efectua Mail Client utilizând protocoalele de trimitere a mesajelor: SMTP și POP3, înțelegînd procesul de trimitere a mesajelor.

În timpul sarcinilor am observat câteva fapte despre SMTP, acest protocol este nesigur, poate fi ușor „hacked", pe de altă parte se poate utiliza sistemul SMTP: permite sincronizarea unui server Smarthost. De exemplu, dacă aveți 200 de calculatoare în birou care utilizează CRM și doriți să vă sincronizați cu serverul SMTP; nu este necesar să configurați SMTP pe fiecare aparat. Trebuie doar să sincronizați un server Smarthost care să gestioneze toate conturile de e-mail.