Ministry of Education of Republic of Moldova

Technical University of Moldova

Faculty of Computers, Informatics and Microelectronics

Computer Science

Network Programming

Laboratory work #4

E-mail client

Authors:
Irina Ungureanu

Supervisor:
Alexandr Gavrișco

Chișinău - 2018

1 Work purpose:

Use application-level protocols (POP3/IMAP and SMTP) to implement an email client with basic functionality.

2 Task implementation

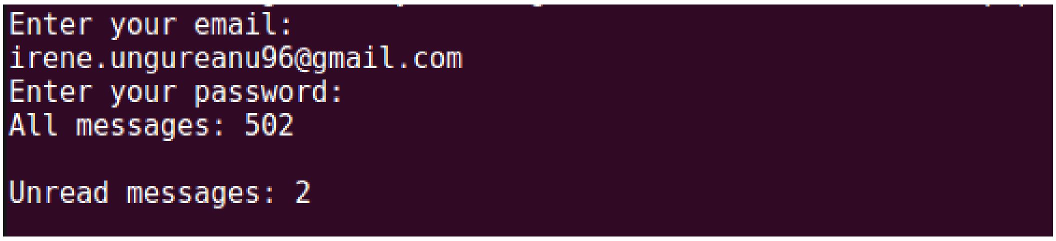
2.1 Log in

In order to log in I used the help of the IMAP ruby native gem. As we can see in the code below, I asked the user for his username and password and then just created a new IMAP object and used the login method. For the password I used STDIN.noecho in order for the password to not be seen in the console.

```ruby
puts "Enter your email: "
username = gets.chomp

puts "Enter your password: "
password = STDIN.noecho(&:gets).chomp!

imap = Net::IMAP.new('imap.gmail.com', ssl: true)
imap.login(username, password)
```



```
Enter your email:
irene.ungureanu96@gmail.com
Enter your password:
All messages: 502

Unread messages: 2
```

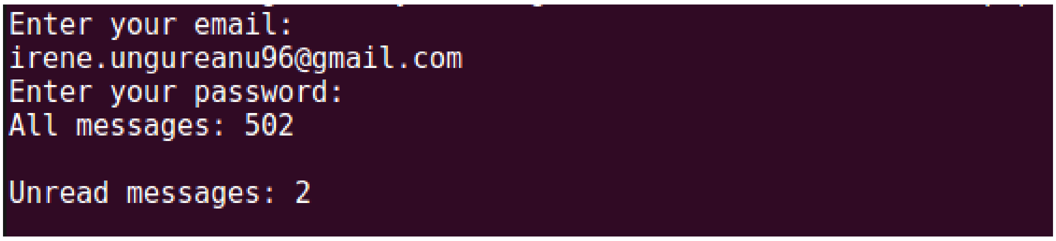
Figure 2.1 – Log in

2.2 Get number of unread messages

In order to get the number of unread messages I used the code below:

```ruby
imap.examine('INBOX')
all_emails = imap.search(["ALL"]).size
puts "All messages: #{all_emails}\n\n"
puts "Unread messages: #{imap.search(["NOT", "SEEN"]).size}\n\n"
```

The first line, sends an EXAMINE command to select a mailbox so that messages in the mailbox can be accessed. Then, I used the search method with the ALL param so I get all the emails and NOT SEEN params which gave me all the unseen emails.



```
Enter your email:
irene.ungureanu96@gmail.com
Enter your password:
All messages: 502

Unread messages: 2
```

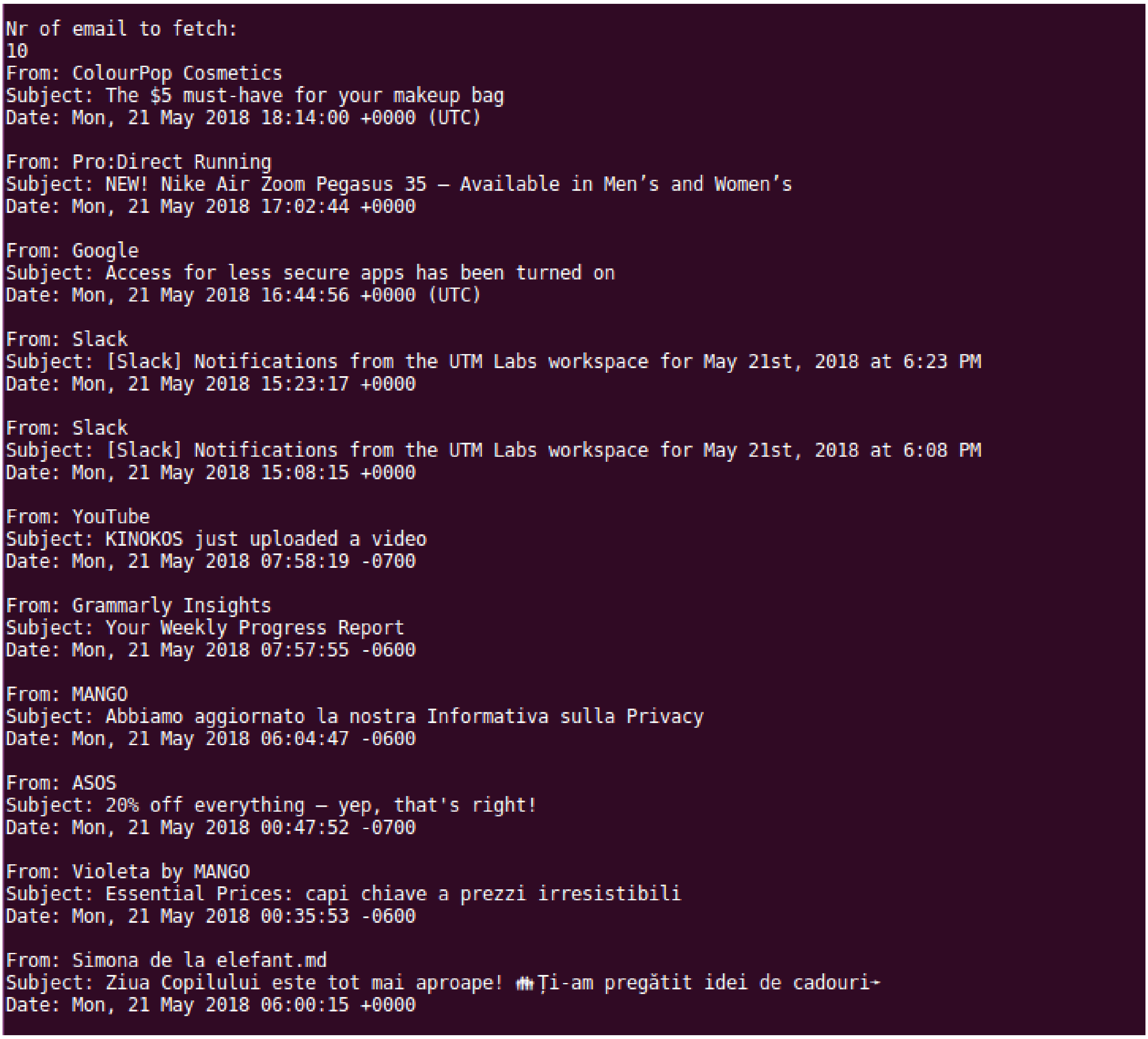
Figure 2.2 – Number of all messages and unread messages

2.3 Get last N received messages (display subject, date, sender), ordered by date

```ruby
puts "Nr of email to fetch: "
fetch_nr = gets.chomp

all_emails.downto(all_emails - fetch_nr.to_i) do |message_id|
  envelope = imap.fetch(message_id, "ENVELOPE")[0].attr["ENVELOPE"]
  subject = Mail::Encodings.unquote_and_convert_to( envelope.subject,
     'utf-8' )
  puts "From: #{envelope.from[0].name}\n"
  puts "Subject: #{subject}\n"
  puts "Date: #{envelope.date}\n\n"
end
```

Firstly, I get how many emails should I fetch. Then, for these emails I call the fetch() method and get the sender, subject and date. I also unquote and convert to utf-8 the subject of the email. The emails are already sorted by date because the last email is the newest one.



```
Nr of email to fetch:
10
From: ColourPop Cosmetics
Subject: The $5 must-have for your makeup bag
Date: Mon, 21 May 2018 18:14:00 +0000 (UTC)

From: Pro:Direct Running
Subject: NEW! Nike Air Zoom Pegasus 35 – Available in Men’s and Women’s
Date: Mon, 21 May 2018 17:02:44 +0000

From: Google
Subject: Access for less secure apps has been turned on
Date: Mon, 21 May 2018 16:44:56 +0000 (UTC)

From: Slack
Subject: [Slack] Notifications from the UTM Labs workspace for May 21st, 2018 at 6:23 PM
Date: Mon, 21 May 2018 15:23:17 +0000

From: Slack
Subject: [Slack] Notifications from the UTM Labs workspace for May 21st, 2018 at 6:08 PM
Date: Mon, 21 May 2018 15:08:15 +0000

From: YouTube
Subject: KINOKOS just uploaded a video
Date: Mon, 21 May 2018 07:58:19 -0700

From: Grammarly Insights
Subject: Your Weekly Progress Report
Date: Mon, 21 May 2018 07:57:55 -0600

From: MANGO
Subject: Abbiamo aggiornato la nostra Informativa sulla Privacy
Date: Mon, 21 May 2018 06:04:47 -0600

From: ASOS
Subject: 20% off everything – yep, that's right!
Date: Mon, 21 May 2018 00:47:52 -0700

From: Violeta by MANGO
Subject: Essential Prices: capi chiave a prezzi irresistibili
Date: Mon, 21 May 2018 00:35:53 -0600

From: Simona de la elefant.md
Subject: Ziua Copilului este tot mai aproape! 👪Ţi-am pregătit idei de cadouri→
Date: Mon, 21 May 2018 06:00:15 +0000
```

Figure 2.3 – Last 10 messages

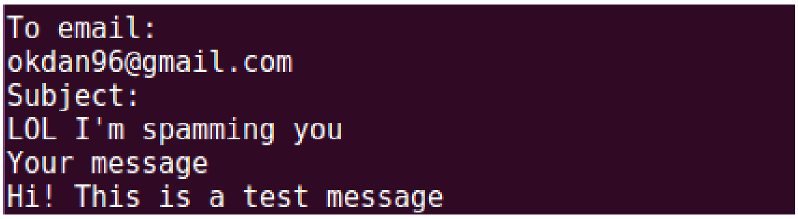
2.4 Send a message

In order to send an email I got the recipient email, subject and content. I had to enable tls in order for the SMTP to work. In line 17 I used the start() method which opens a TCP connection and starts the SMTP session. Then I send the message which I put together in lines 8 through 14.

```
1  puts "To email: "
2  to_email = gets.chomp
3  puts "Subject: "
4  subject = gets.chomp
5  puts "Your message"
6  content = gets.chomp
7
8  content = <<EOF
9  From: #{username}
10 To: #{to_email}
```

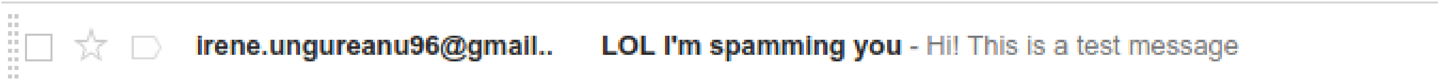
```
Subject: #{subject}
Date: #{Time.now.rfc2822}
#{content}
EOF

Net::SMTP.enable_tls(OpenSSL::SSL::VERIFY_NONE)
Net::SMTP.start('smtp.gmail.com', 587, 'gmail.com', username, password, :login) do |smtp|
  smtp.send_message content, username, to_email
end
```



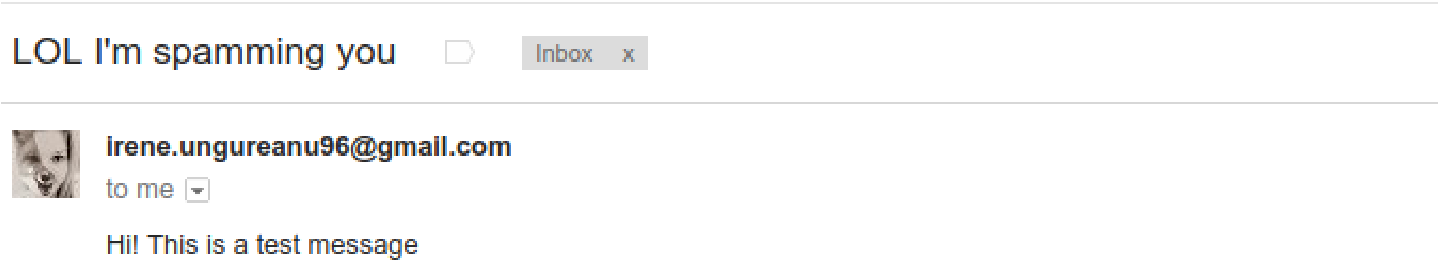
```
To email:
okdan96@gmail.com
Subject:
LOL I'm spamming you
Your message
Hi! This is a test message
```

Figure 2.4 – Message content



irene.ungureanu96@gmail.. LOL I'm spamming you - Hi! This is a test message

Figure 2.5 – Received message



LOL I'm spamming you

irene.ungureanu96@gmail.com

to me

Hi! This is a test message

Figure 2.6 – Opened received message

2.5 If you implemented previous task using POP3 protocol, migrate the app to IMAP

No need, already used IMAP :D

Conclusion

During this laboratory work, I studied the application-level protocols (POP3/IMAP and SMTP) and applied this knowledge in order to create a basic email client. For email fetching I used the IMAP protocol and for email sending the SMTP protocol.