

**Project – Zalando Price Tracker**

Sabrina Cappellari

16-606-303

sabrina.cappellari@student.unisg.ch

Nicolas Greiner

19-611-193

nicolas.greiner@student.unisg.ch

Silvan Moser

16-565-228

silvan.moser@student.unisg.ch

Norielys De Leon Cabral

17-607-235

norielys.deleoncabral@student.unisg.ch

Nina Ollagnon

18-616-235

nina.ollagnon@student.unisg.ch

University of St. Gallen

Fall Semester

3,793,1.00 Skills: Programming Introduction Level /

7,789,1.00 Skills: Programming with

Advanced Computer Languages

Dr. Mario Silic

**Overview**

This project describes a Python program which is able to track prices from Zalando (or any other web scrapping capable website) and informs the user by e-mail as soon as the price fell below a given price level. Additionally, the code contains multiple functions to retrieve the title, brand and price rating of three different Zalando products.

**Description**

Online shopping is on the rise. We as students don't have too much budget, so a price tracker comes in handy. In our project we are creating a program that informs the user by e-mail as soon as one of the prices of the three Zalando items has dropped below the maximum willingness to pay of the users of the program. We used web scrapping to extract information from the website.

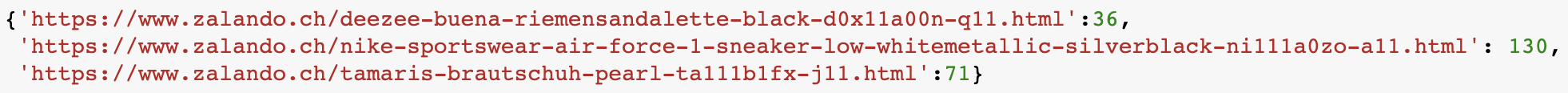
**Steps of the program**

First, it is necessary to import different libraries:

A picture containing shape

Description automatically generated

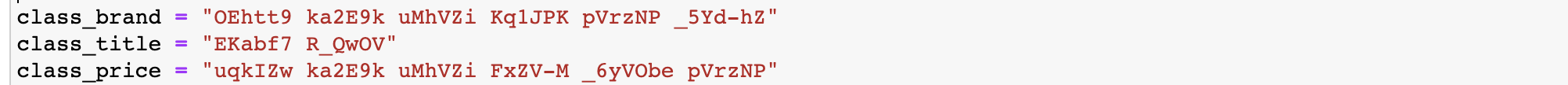
* *Requests:* HTTP library for Python, to send in a very simple way HTTP requests. The request returns a response object (content, headers, status)
* *BeautifulSoup:* library to scrape information from web pages (on HTML or XLM parser). Creation of a parse ‘tree’ for pages that have been parsed to extract data.
* *Smtplib:* library to send email to SMTP servers
* *Html5lib:* library for parsing HTML
* *Os:* provides different functions for interacting with the operating system
* *Boto3:* to be able to use AWS services in Python, it is the SDK of AWS
* *Botocore.exceptions:* to find, handle exceptions from Boto3

Then, we created a dictionary with the URL of the articles as keys and the maximum we are willing to pay for those as values.

A picture containing graphical user interface

Description automatically generated

Then, the class attributes of the HTML page elements are defined. Those can be found with the “inspect” option of an HTML page. They are used to access specific elements. We then retrieved the class attributes for the brand of the articles, the title of the HTML page and the different prices of the products.



Here, we specify information about our browser so that the servers can identify the application, operating system of the requesting user agent. This information is necessary to then use a HTTP request. 

The send\_mail() function sends an e-mail to the defined recipient if certain conditions are met, i.e.: if the prices of the articles go below the price limit defined by the user agent. The Amazon “Simple Email Service” (SES) is used for this task. The sender and recipient’s addresses have to be verified in order for the service to work. To use this service in Python, boto3 has to be installed. A SES resource, specifies the service we are using, SES, and from where the client is using it.

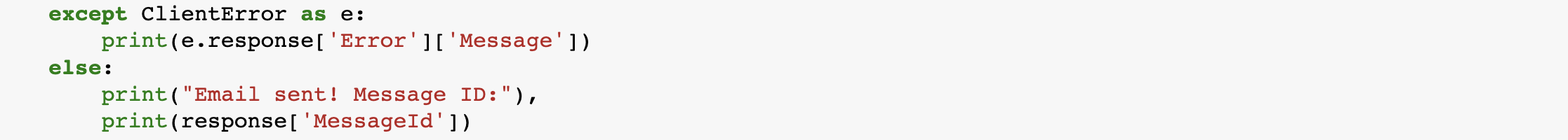
The send\_email() method composes an email message and prepares it for sending. A certain number of conditions have to be met. If those conditions are not met, a ClientError exception is raised. The specific conditions are specified here:

<https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/ses.html#SES.Client.send_email>

If all the conditions are met, the function returns the Message ID of the email.

Graphical user interface, text, application, email

Description automatically generated



The update\_price() function is there to retrieve the brand, title and prices of the articles and return it. The find() method finds the class attributes defined previously and returns the string associated with it.

Text

Description automatically generated with medium confidence

The get\_rating() function returns the rating of the article. The find() method is used again for returning the string values (specific rating of the article) of the class attribute, inside the tag 'span'.

A picture containing text

Description automatically generated

The last part of the code links the different functions. We iterate through the dictionary values (the different articles) and return the elements defined in the update\_price() function.

Graphical user interface, text

Description automatically generated

Then, the send\_mail() function is called if the limit prices defined in the dictionary as value are not reached. The get\_rating function is used for the Nike Sneakers, as an example.

**Notes**

1. We set up an email to check whether the program works or not:   
   trackingprices7@gmail.com password: PricesDown8
2. You can access the email address to see whether the email was sent. <https://accounts.google.com/AccountChooser/identifier?service=mail&continue=https%3A%2F%2Fmail.google.com%2Fmail%2F&flowName=GlifWebSignIn&flowEntry=AccountChooser>