Webscraping,

-automation and

-crawling mit Python

Agenda

1) Data Analytics Center @TeamBank

2) Webscraping

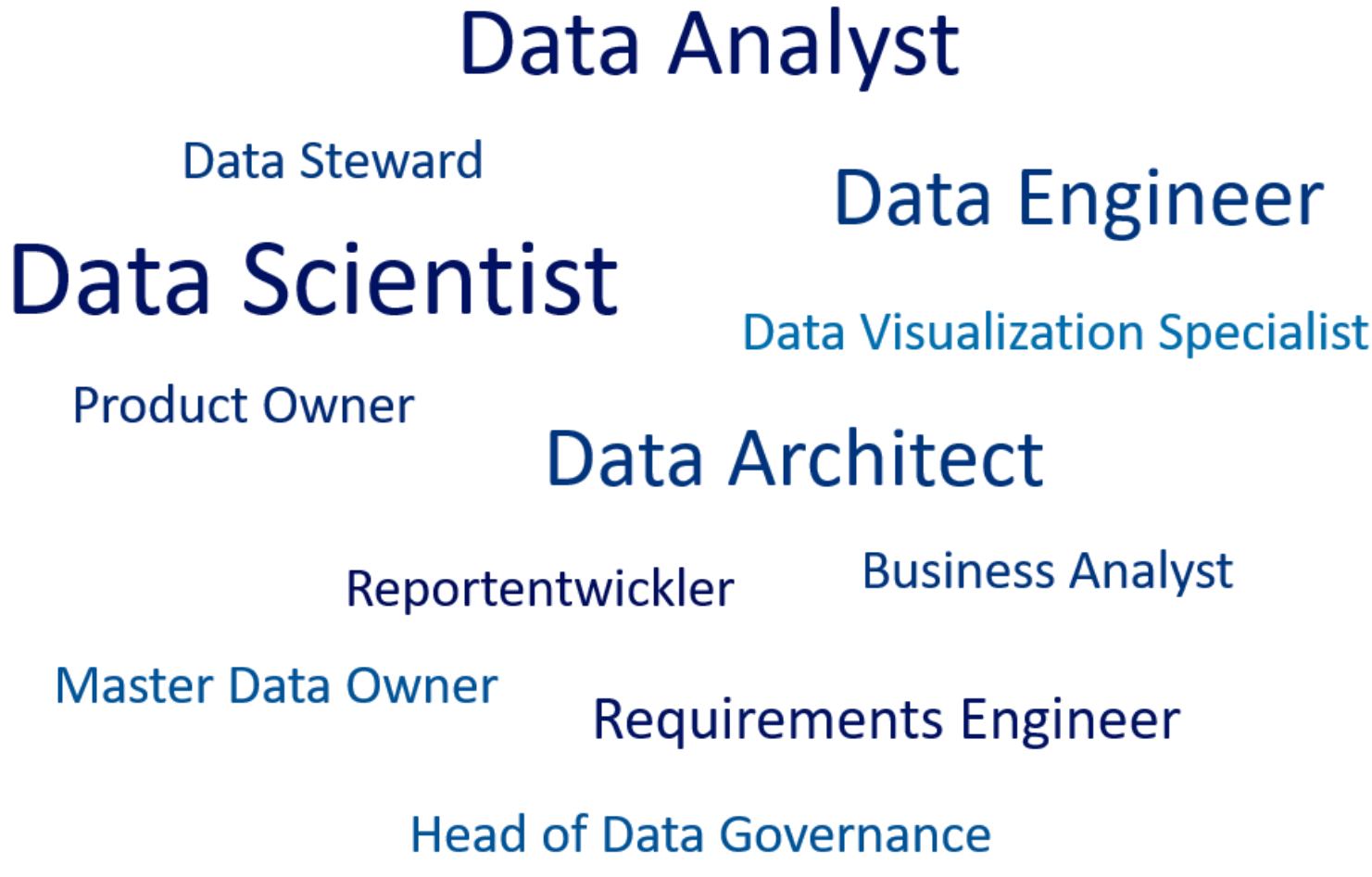
3) Webautomation

4) Webcrawling

5) Use Case @TeamBank

1) Data Analytics Center @TeamBank

Data Analytics Center



Data Science Tool Box

IDE

- Jupyter Notebook
- Visual Studio Code
- PyCharm

Packages

- matplotlib, seaborn
- scikit-learn, Keras, TensorFlow, CatBoost, XGBoost
- RASA NLU, spaCy
- Selenium, Scrapy, BeautifulSoup, urlib

2) Webscraping

HTML-Seite untersuchen

www.teambank.de (https://www.teambank.de/)

HTTP Request starten

In [2]:
```
# load relevant packages
from urllib.request import urlopen, Request
```

In [3]:
```
# define url
url = "https://www.teambank.de/nuedigital/"
url
```

Out[3]:
```
'https://www.teambank.de/nuedigital/'
```

In [4]:
```
# send http request
response = urlopen(Request(url))
response
```

Out[4]:
```
<http.client.HTTPResponse at 0x239e8c1c7f0>
```

HTTP Response verarbeiten

In [5]:
```
# get http status code
print(f"HTTP Status code: {response.status}")
```

```
HTTP Status code: 200
```

HTTP Response verarbeiten

In [6]:
```
# extract html code from response (<meta charset="utf-8">)
html = response.read().decode("utf-8")
```

In [7]:
```
print(f"Type of html doc: {type(html)}")
```

```
Type of html doc: <class 'str'>
```

In [9]:
```
print(html[:1000])
```

```
<!doctype html>
<!-- TODO :: set correct blog language as html lang code -->
<html lang="en">

        <head>

                <!-- Google Tag Manager -->
                <script>(function(w,d,s,l,i){w[l]=w[l]||[];w[l].push({'gtm.start':
                new Date().getTime(),event:'gtm.js'});var f=d.getElementsByTagName(s)
[0],
                j=d.createElement(s),dl=l!='dataLayer'?'&l='+l:'';j.async=true;j.src=
                'https://www.googletagmanager.com/gtm.js?id='+i+dl;f.parentNode.inser
tBefore(j,f);
                })(window,document,'script','dataLayer','GTM-MWR3PG7');</script>
                <!-- End Google Tag Manager -->
                <!-- Facebook Pixel Code -->
                <script>
                  !function(f,b,e,v,n,t,s)
                  {if(f.fbq)return;n=f.fbq=function(){n.callMethod?
```

BeautifulSoup zum Extrahieren von Informationen

- Transformation von HTML (und XML) Dokumenten in durchsuchbare Objekte
- Suchen nach HTML-Tag-Namen
- Navigieren durch Baumstruktur ohne konkreten Tag-Namen

HTML Seite parsen

In [8]:
```
from bs4 import BeautifulSoup
```

In [9]:
```
# parse html
soup = BeautifulSoup(html, 'html.parser')
```

In [10]:
```
print(f"Type of html doc: {type(soup)}")
```

```
Type of html doc: <class 'bs4.BeautifulSoup'>
```

Spezifischen Inhalt extrahieren

In [12]:
```
# find info by html tag
soup.title.get_text()
```

Out[12]:
```
'#nuedigital'
```

In [13]:
```
# get all links
for link in soup.find_all("a")[:10]:
    url = link.get("href")
    if url.startswith("https"):
        print(url)
```

```
https://www.teambank.de/unternehmen/
https://www.teambank.de/unternehmen/
https://www.teambank.de/unternehmen/die-teambank/
https://www.teambank.de/unternehmen/die-teambank/#Portrait
https://www.teambank.de/geschaeftsbericht/
https://www.teambank.de/unternehmen/die-teambank/#Standort
https://www.teambank.de/unternehmen/organe-und-gremien/
https://www.teambank.de/unternehmen/organe-und-gremien/#Vorstand
https://www.teambank.de/unternehmen/organe-und-gremien/#Aufsichtsrat
```

Gesamten Inhalt (Body) extrahieren

In [14]:
```
import re
```

In [15]:
```
# extract text from html page
text = soup.get_text()
```

In [16]:
```
# remove unnecessary content
text = re.sub("[\n\t\r ]+", " ", text)
```

In [17]:
```
text[6318:8100]
```

Out[17]:
```
'Auch bei uns im Haus liegt der Fokus auf den Einsen und Nullen und was sie für unser
Leben bedeuten. Vom Webscraping über Digital Learning bis hin zur Vorstellung unserer
eigenen TeamBank-Arbeitswelt ist für jeden was dabei. Kommet zuhauf! .module.accordio
n .block { width: 100%; } DevBBQ@TeamBank Wir zeigen in drei Sessions Webscraping mit
Python, KI-Modelle in Kubernetes und Camunda BPM. Im Anschluss gibt es Gelegenheit zu
m Austausch bei einem BBQ auf der Dachterrasse. Camunda: Eine Workflow-Engine, die BP
MN (Business Process Model and Notation) ausführbar macht. Prozesse werden transparen
t und können sogar Spaß machen. Anhand eines einfachen Beispiels werden wir einen Pro
zess entwerfen, diesen mit Leben füllen und in OpenShift bereitstellen. \xa0 Webscrap
ing und -automation mit Python: Automatisiert Informationen aus öffentlich zugänglich
en Internet-Seiten gewinnen. Wie die TeamBank Webscraping nutzt, um sich das Web als
Datenquelle zu erschließen. Lerne den Aufbau und die Nutzung deiner eigenen Scraper.
\xa0 Open Source-basierte KI-Microservices in Kubernetes: KI-Anwendungen helfen unter
anderem dabei Prozesse zu automatisieren – Chatbots, Textklassifizierung, Sentiment-A
nalysen und Betrugserkennung. Wir zeigen euch, wie man leichtgewichtige KI-REST-APIs
in einem Kubernetes Cluster deployt und welche Fallstricke es zu beachten gilt. Hier
anmelden Digital Learning Wie viel Digitalisierung verträgt das berufliche Lernen im
Dienstleistungssektor? Vom Träumen und Aufwachen zur neuen Lernkultur. Ein Praxiseinb
lick mit Diskussion beim Snack auf der Dachterrasse. Digital Learning ist mehr als We
b Based Training. Wir geben einen Einblick in die Herausforderungen, die uns bei der
Entwicklung gegenüberstanden. Die Fragestellungen bei der Einführung einer digitalen
'
```

3) Webautomation

Selenium zum Automatisieren von User-Aktivitäten im Web

- Formulare ausfüllen
- Buttons klicken
- Dropdown-Listen auswählen
- Ansicht des Browser-Fensters verändern
- Screenshots erstellen

Vorraussetzung:

- Download eines Web Drivers und Platzieren in Ordner, in dem python.exe liegt
- Bsp.: Chrome https://sites.google.com/a/chromium.org/chromedriver/downloads (https://sites.google.com/a/chromium.org/chromedriver/downloads)
- Alternativen: Firefox, Android, Safari

Driver und Link öffnen

In [18]:
```
# load relevant packages
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
```

In [19]:
```
# open browser with Chrome Webdriver
browser = webdriver.Chrome()
```

In [20]:
```
# define url
url = "https://www.easycredit.de/"
browser.get(url)
```

Java Script ausführen

- JavaScript in HTML body wird ausgeführt

In [ ]:
```
# execute java script
script = browser.execute_script("return document.body.innerHTML")
```

In [ ]:
```
script
```

Scrollen

In [21]:
```
# scroll down
browser.find_element_by_tag_name('body').send_keys(Keys.END)
```

In [22]:
```
# scroll up
browser.find_element_by_tag_name('body').send_keys(Keys.HOME)
```

www.youtube.de (https://www.youtube.de/)

Screenshots erstellen

In [23]:
```
browser.save_screenshot("screenshot.png")
```

Out[23]:
```
True
```

In [26]:

```
import matplotlib.pyplot as plt
import matplotlib.image as mpimg

plt.rcParams["figure.figsize"] = (10,10)

img=mpimg.imread('screenshot.png')

plt.imshow(img)
plt.axis("off")
plt.show()
```

Textfelder befüllen und Buttons klicken

In [27]:
```
# get field "PLZ oder Ort eingeben"
loc = browser.find_element_by_name("location")
```

In [28]:
```
# send Ort
loc.send_keys("Nürnberg")
```

In [29]:
```
# get button "Filiale finden"
next_page = browser.find_element_by_css_selector("#mapForm > div > button")
```

In [30]:
```
# click on "Filiale finden" Button
next_page.click()
```

4) Webcrawling

Scrapy zum schnellen Entwickeln von Crawlern

- Durchsuchen von Webseiten (nicht einzelner Urls)
- Parallelisierung und Drosselung konfigurierbar
- Regeln zum Folgen von Links definieren
- Detailliertes Logging

Entwicklung und Konfiguration eines Crawlers

DEMO

Ergebnisse

In [10]:
```
import pandas as pd
```

In [11]:
```
# load data
data = pd.read_json(r"/NUEDigital/items_backup.json", encoding="utf-8")
```

In [12]:
```
# show sample of data
data.sample(10)
```

Out[12]:

	text	url
84	Sicherheit bei fymio hat oberste Priorität Üb...	https://www.fymio.de/sicherheit/
73	Features - fymio Über uns Team Unternehmen Üb...	https://www.fymio.de/features/
158	Daran erkennt man das optimale Fitnessstudio ...	https://www.easycredit.de/blog/fitnessstudio-c...
170	Fahrgemeinschaften ▶ Vorteile, Regeln und Rec...	https://www.easycredit.de/blog/fahrgemeinschaf...
212	#gemeinsam Teambank Übersicht Kennzahlen Gruß...	https://www.teambank.de/geschaeftsbericht/geme...
153	Kreditvertrag vom Sofa abschließen I easyCred...	https://www.easycredit.de/blog/kreditvertrag-v...
187	Werbemittel und Banner zum Ratenkauf Toggle n...	https://www.easycredit-ratenkauf.de/werbemitte...
118	Kredit aufnehmen war nie einfacher | easyCred...	https://www.easycredit.de/kredit
92	Finanzierung mit TOP Raten | easyCredit Start...	https://www.easycredit.de/kredit/finanzierung
106	Für Kunden | Ganz entspannt in Raten zahlen |...	https://www.easycredit.de/ratenkauf-by-easycredit

Ergebnisse

In [20]:
```
for idx, row in data.sample(10).iterrows():
    print(row["url"])
```

```
https://www.easycredit.de/erfahrungen
https://www.teambank.de/flaggefuervielfalt-diversity-tag-2018/
https://www.easycredit-ratenkauf.de/veranstaltungen.htm
https://www.fymio.de/ueber-uns/
https://www.easycredit.de/kredit/darlehen
https://www.fymio.de/unternehmen/
https://angebot.easycredit.de/online-abschluss/
https://www.teambank.de/karriere/kultur/
https://www.easycredit.de/blog/e-bikes-im-vergleich
https://www.fymio.de/anleitung/
```

In [21]:
```
# get number of crawled pages
print(f"Successfully crawled {data.shape[0]} pages.")
```

```
Successfully crawled 215 pages.
```

5) Use Case @TeamBank

Fragen ?

Kontakt

Magdalena Deschner | Data Scientist
T: +49 (0) 911 / 539 037 51
E: magdalena.deschner@teambank.de

www.teambank.de

typischteambank

What's next?

In [36]:
```
from datetime import datetime, timedelta
```

In [37]:
```
def round_time(tm, minutes=5):
    """
    Parameter
    ---------
    tm: datetime
        Datetime to be rounded to neares minutes
    minutes: int
        Step to be rounded to

    Examples
    --------
    >>> pause_dt = round_time(datetime.datetime(2019, 7, 8, 13, 58))
    datetime.datetime(2019, 7, 8, 14, 0)

    Returns
    -------
    datetime
        Rounded datetime
    """
    discard = timedelta(minutes=tm.minute % minutes,
                                seconds=tm.second,
                                microseconds=tm.microsecond)
    tm -= discard
    if discard >= timedelta(minutes=minutes/2):
        tm += timedelta(minutes=minutes)
    return tm
```

In [38]:

```
now = round_time(datetime.now())
pause_1 = round_time(now) + timedelta(minutes=5)
kubernetes = round_time(pause_1) + timedelta(minutes=30)
pause_2 = round_time(kubernetes) + timedelta(minutes=15)
```

In [39]:
```
print(f"{now.strftime('%H:%M')} bis {pause_1.strftime('%H:%M')} Pause/Start Hausführung
\n{pause_1.strftime('%H:%M')} bis {kubernetes.strftime('%H:%M')} Open Source-basierte KI
-Microservices in Kubernetes \n{kubernetes.strftime('%H:%M')} bis {pause_2.strftime('%
H:%M')} Pause/Start Hausführung\n{pause_2.strftime('%H:%M')} bis {'20:00'} Camunda\nab
 {'20:00'} BBQ auf der Dachterasse")
```

```
18:35 bis 18:40 Pause/Start Hausführung
18:40 bis 19:10 Open Source-basierte KI-Microservices in Kubernetes
19:10 bis 19:25 Pause/Start Hausführung
19:25 bis 20:00 Camunda
ab 20:00 BBQ auf der Dachterasse
```