DerbyHospital

Gustavo Mazzanti (0000914975)
Tommaso Brini (0000933814)

23 luglio 2021

Indice

Capitolo 1

Introduzione

Il progetto consiste nel realizzare un Web Server in Python per una azienda ospedaliera. Abbiamo deciso di creare il Web Server per l'ospedale San Raffaele di Milano.

Capitolo 2

Descrizione

Il programma, implementato interamente nel modulo Derby-Hospital.py, consiste in un server che apre una connessione tramite socket TCP nell'interfaccia dell'utente e carica i file HTML. L'applicazione contiene i seguenti metodi:

- **getLink()** : ispeziona l'HTML del sito del SanRaffaele cercando i primi nove servizi e salvandone nome e link in un dizionario.

```
39  def getLink():
40      url = link_hospital
41      count = 0
42      c = 0
43      dizionario={}
44      while count<9:
45          if c<10:
46              tag = "00" + str(c)
47          elif c<100:
48              tag = "0" + str(c)
49          else:
50              tag = str(count)
51
52          only_links = SoupStrainer("a", href=re.compile(tag))
53          soup = BeautifulSoup(urlopen(url), parse_only=only_links,features="lxml")
54          urls = [urljoin(url,a["href"]) for a in soup(only_links)]
55          link = "n".join(urls)
56          c+=1
57          try :
58              dizionario[count]= str(soup.a.string), str(link)
59              count+=1
60          except AttributeError:
61              c=c
62      return dizionario
63
```

Figura 2.1: Metodo getLink()

- **getIP()** : restituisce l'IP della macchina su cui il programma è stato avviato.

```
76
77  def getIp():
78      s = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
79      try:
80          s.connect(("10.255.255.255", 1))
81          ip = s.getsockname()[0]
82
83      except:
84          ip = '127.0.0.1'
85      finally:
86          s.close()
87      print("L'ip per il sito è:" ,ip)
88      return ip
89
90  ip = getIp();
91
```

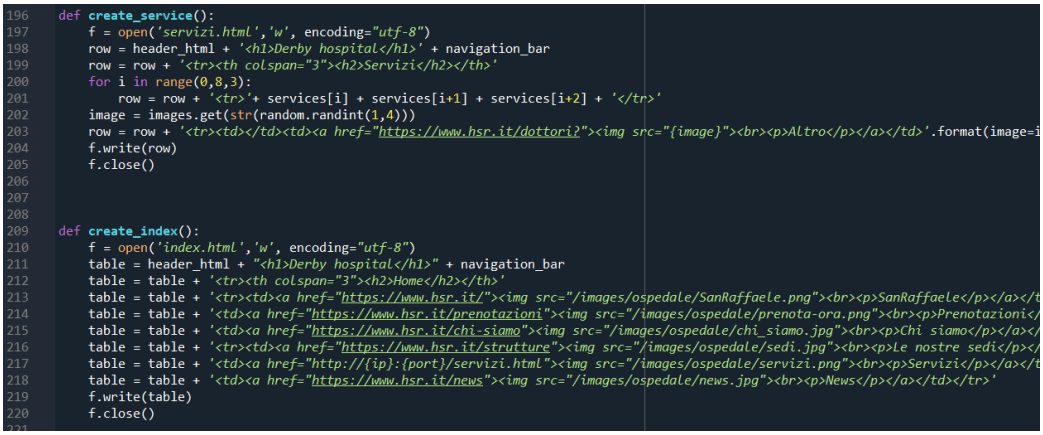
Figura 2.2: Metodo getIp()

- **refresh-contents()** : richiama i vari metodi per aggiornare i contenuti.
- **load-services()** e **add-service()** : prendono tutti i servizi contenuti nel dizionario e li inseriscono in una apposita struttura dati.

```
182 def load_services():
183     dizionario=getLink()
184     c=0
185     while c<9:
186         name,link=dizionario.get(c)
187         add_service(link, name)
188         c+=1
189
190
191 def add_service(link, name):
192     image = images.get(str(random.randint(1,4)))
193     service = str('<td><a href="{link}"><img src="{image}"><br><p>{name}</p></a></td>'.format(link=link,image=image,name=name))
194     services.append(service)
```

Figura 2.3: Metodi load-services() e add-services()

- **create-service()** e **create-index()** : creano i file HTML rispettivamente per la pagina dei servizi e per la pagina iniziale.



```
def create_service():
    f = open('servizi.html','w', encoding="utf-8")
    row = header_html + '<h1>Derby hospital</h1>' + navigation_bar
    row = row + '<tr><th colspan="3"><h2>Servizi</h2></th>'
    for i in range(0,8,3):
        row = row + '<tr>'+ services[i] + services[i+1] + services[i+2] + '</tr>'
    image = images.get(str(random.randint(1,4)))
    row = row + '<tr><td></td><td><a href="https://www.hsr.it/dottori?"><img src="{image}"><br><p>Altro</p></a></td>'.format(image=i
    f.write(row)
    f.close()



def create_index():
    f = open('index.html','w', encoding="utf-8")
    table = header_html + "<h1>Derby hospital</h1>" + navigation_bar
    table = table + '<tr><th colspan="3"><h2>Home</h2></th>'
    table = table + '<tr><td><a href="https://www.hsr.it/"><img src="/images/ospedale/SanRaffaele.png"><br><p>SanRaffaele</p></a></t
    table = table + '<td><a href="https://www.hsr.it/prenotazioni"><img src="/images/ospedale/prenota-ora.png"><br><p>Prenotazioni</
    table = table + '<td><a href="https://www.hsr.it/chi-siamo"><img src="/images/ospedale/chi_siamo.jpg"><br><p>Chi siamo</p></a></
    table = table + '<tr><td><a href="https://www.hsr.it/strutture"><img src="/images/ospedale/sedi.jpg"><br><p>Le nostre sedi</p></
    table = table + '<td><a href="http://{ip}:{port}/servizi.html"><img src="/images/ospedale/servizi.png"><br><p>Servizi</p></a></t
    table = table + '<td><a href="https://www.hsr.it/news"><img src="/images/ospedale/news.jpg"><br><p>News</p></a></td></tr>'
    f.write(table)
    f.close()
```

Figura 2.4: Metodi create-services() e create-index()

- **signal-handler()** : gestisce l'interruzione del programma da console.

```
223
224  def signal_handler(signal, frame):
225      print('Exiting (Ctrl+C pressed)')
226      try:
227        if(server):
228          server.server_close()
229      finally:
230        sys.exit(0)
231
```

Figura 2.5: Metodo signal-handler()

Capitolo 3

Funzionamento

Innanzitutto, bisogna lanciare il modulo `DerbyHospital.py` che si occuperà di creare la connessione e che caricherà il Web Server. Dopodichè, da un qualsiasi motore di ricerca sarà visibile il sito scrivendo nella barra di ricerca la propria interaccia seguita dalla porta 8080.

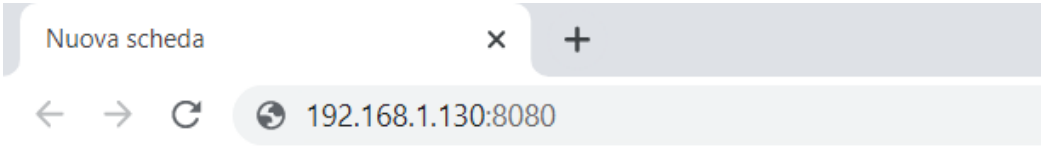


Figura 3.1: Esempio di come cercare il Web Server da un qualsiasi motore di ricerca

Il sito si aprirà alla pagina index.html, cioè la schermata iniziale. Selezionando le opzioni nella tabella centrale (e analogamente nella barra orizzontale situata in alto) si verrà ridirezionati alla pagina selezionata.

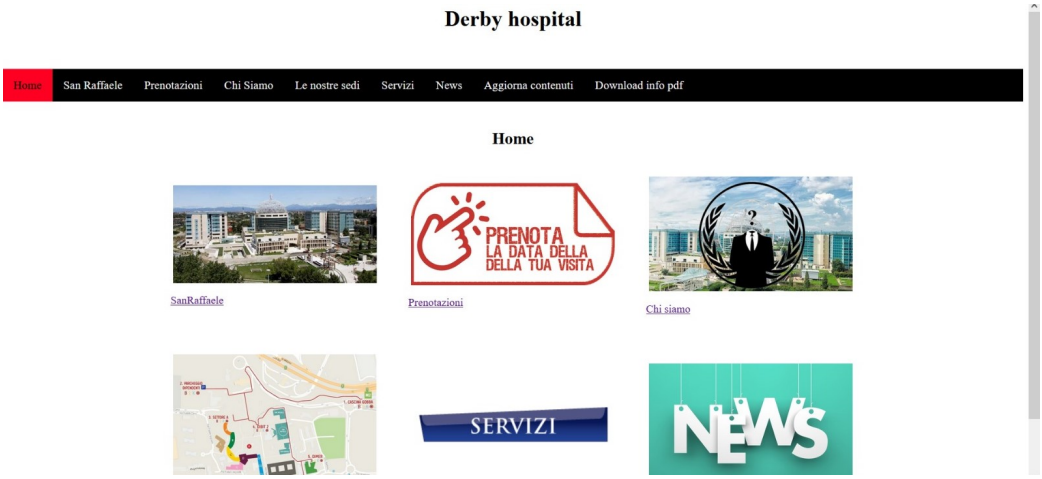


Figura 3.2: Schermata iniziale

Se si sceglierà la pagina dei servizi, verrà aperta la pagina HTML contenente i link dei vari servizi che l'ospedale offre.

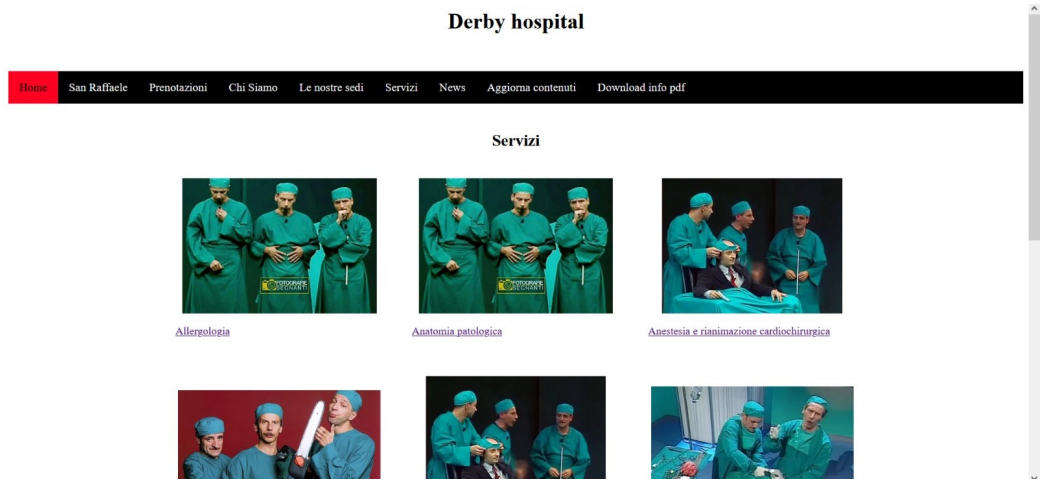


Figura 3.3: Schermata dei servizi

Capitolo 4

Librerie utilizzate

- socket
- bs4.BeautifulSoup
- random
- sys
- signal
- hhtp.server