

СЕКЦИЯ III. Информатика и системы управления

отформатировано: Шрифт: Times New Roman, 20 пт, полужирный

Автоматическая запись к врачу с получением уведомлений (Quick appointment with a doctor) Полнофункциональный мобильный фоторедактор Роло с элементами социальной сети

отформатировано: Шрифт: 20 пт, полужирный
Отформатировано: По центру, междустрочный, 1.5 строки

отформатировано: русский
отформатировано: русский
отформатировано: русский
отформатировано: русский
отформатировано: русский
отформатировано: Шрифт: 20 пт, полужирный

Чудновский Новокшанов Даниэль Евгений Евгеньевич Андреевич
г. Москва, ГБОУ Школа Физматшколы №138 №2007, 11 класс
Научный руководитель: Степанов Павел Валерьевич, преподаватель
кафедры ИУ-6, МГТУ им. Н.Э. Баумана

отформатировано: Шрифт: Times New Roman, 14 пт, курсив
отформатировано: Шрифт: курсив
отформатировано: Шрифт: Times New Roman, 14 пт, курсив
отформатировано: Шрифт: курсив
отформатировано: Шрифт: Times New Roman, 14 пт, курсив
отформатировано: Шрифт: курсив
отформатировано: Шрифт: Times New Roman, 14 пт, курсив
отформатировано: Шрифт: курсив
отформатировано: Шрифт: Times New Roman, 14 пт, курсив
отформатировано: Шрифт: курсив
отформатировано: Шрифт: 14 пт, курсив

QAWD - это приложение, которое сможет заметно улучшить вашу жизнь в сфере записи к врачу. Если вы стоите в онлайн очередях или очередях поликлиник, то с QAWD вы сможете освободить свое время и посвятить его любимым занятиям и хобби, потому что мое веб-приложение будет выполнять запись к врачу без непосредственного наблюдения пользователя за его работой. Вам остается только нажать на кнопку и получить уведомление на почту о успешной записи.

Цель работы:

Написание программы для автоматической записи к врачу с получением уведомлений.

Задачи:

- 1) решить проблему потери времени людей, которые стоят в онлайн-очередях для записи на прием к врачу;
- 2) облегчить процесс записи к врачу на определенное время и дату.

Методы и инструменты:

- 1) программа написана на языке Python 3.7;
- 2) использована библиотека Selenium WebDriver (библиотека для управления браузерами);

3) работа с веб-сервисами.

Описание:

После введения определенных параметров записи (интервалы даты и времени, специализация врача, Ф.И.О. врача) выполняется автоматический поиск и отслеживание доступного для записи к врачу даты и времени. После того, как программа проведет запись, она отправит уведомление с указанием даты и времени приема у врача.

Содержание:

1) Введение

2) Используемые методы

3) Интерфейс

4) Связующая программа

5) Основная программа

6) Отправка сообщений

7) Заключение

8) Приложение

Введение

Люди давно уже не стоят часами в очередях, чтобы записаться к врачу. Существует множество сервисов, через которые можно выполнить запись, не выходя из дома, однако и часто требует много времени. Люди, имеющие онкологические и сердечно-сосудистые заболевания, не могут записаться на прием, т.к. все время занято, и они вынуждены постоянно следить, не освободилось ли место в нужное время.

Я решил разобраться с этой проблемой и написать собственное веб-приложение, которое сможет автоматически искать свободное время для записи. Это освободило бы много времени у людей, которые в этом нуждаются.

Основание моего приложения состоит из следующих идей:

8.1. Быстрая запись

9.2. Если нет подходящего времени, то программа продолжает поиск

3. Обеспечение скорости записи

Обеспечение скорости записи

Обеспечение скорости записи

10.4. Понятный и простой интерфейс, которым будет удобно пользоваться не только людям среднего возраста , но и людям преклонного возраста.

Я считаю, что именно веб-приложение является наиболее удобным в использовании, т.к. оно не занимает памяти на устройстве, и пользоваться им можно где угодно и когда угодно.

В чем же заключаются отличия моего приложения от аналогов в Google Play, App Store и сайтов гос. услуг? Оно заключается в том, что пользователю не нужно будет заходить непосредственно на сайт в установленное приложение, если необходимо будет лишь открыть вкладку в браузере по ссылке и вписать свои данные, необходимые при записи. Далее программа выполнит всю работу сама и отправит вам сообщение о успешной записи на почту.

Моя программа ничего не весит, поэтому у нее не занимает место на устройстве, а также предоставляет больший функционал, чем аналоги.

1 Используемые методы

Приложение было написано на языке Python 3. Также для создания веб-интерфейса использовался язык HTML5.

Я выбрал язык Python, потому что он прост в использовании и для него написано множество библиотек, которые упрощают работу.

Приложение состоит из нескольких функциональных блоков:

переходе по ссылке на веб-приложение, вы увидите название веб-приложения и строки, необходимые для заполнения:

- 1) Имя
- 2) Электронная почта
- 3) Номер телефона
- 4) Направление (например, Офтальмолог)
- 5) Врач
- 6) Интервал времени, в которое вам удобно посетить врача
- 7) Интервал дней, в которые вам необходимо посетить врача

После того, как вы заполните все выше перечисленное, вам необходимо нажать на кнопку «Записаться»

Автоматическая запись к врачу

Имя

Эл.почта

Номер телефона

Направление

Офтальмолог

Врач

Иванов И.И.

Выберите интервал:

-- 1 --

-- 1 --

Выберите интервал:

ДД - ММ - ГГГГ

ДД - ММ - ГГГГ

Записаться

Для того, что бы это сделать, мне необходимо было изучить работу HTML5.

После нажатия на эту кнопку, данные пользователя передаются в связующую программу.

Элемент страницы с тегом style отвечает за раскраску кнопки «Записаться».

При выборе направления и врача открывается окно с дня выбора направления и доктора соответственно.

Автоматическая запись к врачу

Имя

Эл.почта

Номер телефона

Направление

✓ Офтальмолог

Дерматология

Лучевая диагностика

Женское здоровье

Ортопедия

Аптечная служба

Экстренная помощь

Внутренние болезни

Лечение ран

Выберите интервал:

ДД . ММ . ГГГГ

ДД . ММ . ГГГГ

Записаться



Автоматическая запись к врачу

Имя

Эл.почта

Номер телефона

Направление

Офтальмолог

Врач

✓ Иванов И.И.

Петров П.П.

Выберите интервал:

-- : --

-- : --

Выберите интервал:

ДД • ММ • ГГГГ

ДД • ММ • ГГГГ

Записаться

А вот и код страницы:

3 Связующая программа

Связующая программа, несмотря на свое название выполняет одну из самых важных функций.

Эта программа считывает данные, которые пользователь вводил на странице. После эти данные отправляются в основную программу. Именно в этой программе осуществляется постоянный в основную программу. Именно в этой программе осуществляется постоянный в основную программу. Именно в этой программе осуществляется постоянный

4 Основная программа:

В этой программе осуществляется запись к врачу по данным, которые были переданы Связующей программой.

Все действия выполняются благодаря библиотеке Selenium. Она позволяет работать с веб-интерфейсом.

Программа делится на несколько функций:

- 1) Функция выбора направления
- 2) Функция выбора доктора
- 3) Функция выбора подходящего времени
- 4) Функция выбора подходящей даты
- 5) Функции, которые отвечают за регистрацию
- 6) Функции, отвечающие за активацию вышеперечисленных

Все функции реализованы путем кликов

Поэтому путем ссылок на определенные «кнопки» мы их нажимаем.

Функция `pecialty_selection(a)` отвечает за выбор направления. После перехода на соответствующую страницу сайта(<https://zhenyanovokshanov1.wixsite.com/klinika/blank-5>) функция начнет проверку «кнопок» на соответствие с данным направлением.

Функция `doctor_choice(target_name)` ищет на сайте необходимого доктора с помощью метода ХРАТН по тексту “doctor_name”

Функции data_choice и time_chose работают вместе:

Функция data_choice выбирает дату и проверяет, есть ли свободное время в этот день путем вызова функции time_chose. Именно поэтому в эту функцию передается 4 переменных:

-интервал дат data_start, data_finish

-интервал времени time_start, time_finish

Так же это обеспечивает «гибкость» программы, поэтому ее можно будет изменить для других сайтов. Я писал программу для сайта, который был специально для этого создан, что бы не нарушать работу реальных сайтов и клиник.

Самую главную роль в этом блоке выполняет последняя функция, которая обеспечивает связь между Связующей программой и функциями, которые отвечают за запись. Так же эта функция отправляет данные в программу отправки сообщения.

Функция data_choice отправляет данные в функцию time_chose, которая

успешной записи. На почту указанную пользователем отправляется сообщение в данной форме: "Уважаемый, {user}! Вы записаны на прием к {doctor} на {time} {day}"

Поэтому пользователь будет не только предупрежден о записи, но и узнает дату, время и доктора. Поиск нужной кнопки осуществляется путем поиска элементов по XPATH: Ву.XPATH, f'*/h3[contains(text(), "{a}")]'



Заключение :

За время написания программы я не только хорошо проводил время, но и изучал то, что раньше для меня было непонятным. Я впервые работал с HTML кодом страницы сайта клиники и интерфейса моего веб-приложения.

Список использованной литературы

1. 1 SurfaceView [Электронный ресурс] -

<http://developer.alexanderklimov.ru/android/views/surfaceview.php>
р

2. 2 Список с множественным выбором и собственным адаптером ArrayAdapter [Электронный ресурс] -

http://developer.alexanderklimov.ru/android/listview/multiselect_customarrayadapter.php

3. 3 Диалоговое окно AlertDialog [Электронный ресурс] -

<http://developer.alexanderklimov.ru/android/alertdialog.php>

4. 4 OkHttp [Электронный ресурс] - <http://square.github.io/okhttp/>

5. 5 Android OkHttp Example [Электронный ресурс] -

<http://www.zoftino.com/android-okhttp-example>

6. 6 Работа с JSON [Электронный ресурс] - [http://java-](http://java-help.ru/android-json/)

[help.ru/android-json/](http://java-help.ru/android-json/)

7. 7 JSON: основы использования [Электронный ресурс] - <https://ruseller.com/lessons.php?id=1212&rub=28>

8. 8 Библиотека GSON [Электронный ресурс] - <http://developer.alexanderklimov.ru/android/library/gson.php>

9. 9 Glide [Электронный ресурс] - <https://github.com/bumptech/glide>

10. 10 Android Glide Example [Электронный ресурс] -

<https://www.dev2qa.com/android-glide-example/>

11.11 Матричные фильтры обработки изображений [Электронный

ресурс] - <https://habr.com/ru/post/142818/>

12.12 Графические фильтры на основе матрицы скручивания

[Электронный ресурс] - <http://wb0.ru/articles/120.htm>

Приложение:

Интерфейс(HTML код):

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Login</title>
  <style>
    .text {
      text-align: center;
    }
    .myButton
    {
      box-shadow:inset 0px 0px 0px 0px #c4ccc0;
      background:linear-gradient(to bottom, #74ad5a 5%, #68a54b 100%);
      background-color:#74ad5a;
      border-radius:10px;
      display:inline-block;
      cursor:pointer;
      color:#ffffff;
      font-family:Times New Roman;
      font-size:23px;
      font-weight:bold;
      padding:10px 40px;
      text-decoration:none;
    }
    .btn
    {
      /* по-умолчанию для <button>, но пригодится для <a> */
      display: inline-block;
      text-align: center;
      text-decoration: none;
      /* создаём маленькие отступы, если кнопки перенесутся на две строки */
      margin: 2px 0;
      /* невидимая граница (понадобится для цвета при наведении/фокусе) */
```

```
border: solid 1px transparent;
border-radius: 4px;
/* размер строится из текста и отступов (без width/height) */
padding: 0.5em 1em;
/* убедитесь, что достаточно контраста! */
color: #ffffff;
background-color: #9555af;
}
</style>
</head>
<body>
  <div class="text">
    <form action="" method="post">

      <h1>Автоматическая запись к врачу</h1>
      <p>
        <label for="username">Имя</label>
        <input type="text" name="username">
      </p>

      <p>
        <label for="email">Эл. почта</label>
        <input type="text" name="email">
      </p>

      <p>
        <label for="phone_number">Номер телефона</label>
        <input type="text" name="phone_number">
      </p>

      <p>
        <label for="direction">Направление</label>
        <select name="direction">
          <option value="direction_1">Офтальмолог</option>
          <option value="direction_2">Дерматология</option>
          <option value="direction_3">Лучевая диагностика</option>
          <option value="direction_4">Женское здоровье</option>
          <option value="direction_5">Ортопедия</option>
        </select>
      </p>
    </div>
  </form>
</body>
</html>
```

`<option value="direction_6">Аптечная служба</option>`

`<option value="direction_7">Экстренная помощь</option>`

`<option value="direction_8">Внутренние болезни</option>`

`<option value="direction_9">Лечение ран</option>`

`</select>`

`<p>`

`<label for="doctor">Врач</label>`

`</p>`

`<select name="doctor">`

`<option value="doctor_1">Иванов И.И.</option>`

`<option value="doctor_2">Петров П.П.</option>`

`</select>`

`<p>`

`Выберите интервал:`

`</p>`

`<p>`

`<input type="time" name="time1">`

`<input type="time" name="time2">`

`</p>`

`<p>`

`Выберите интервал:`

`</p>`

`<p>`

`<input type="date" name="calendar1">`

`<input type="date" name="calendar2">`

`</p>`

`<button class="myButton" type="submit">Записаться</button>`

`</form>`

`</div>`

`</body>`

`</html>`

Связующая программа:

`from flask import Flask, render_template, request`

`from Klinika_way import take_zapis`

`from threading import Thread`


```

from selenium import webdriver
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.by import By
from webdriver_manager.chrome import ChromeDriverManager
import time

app = Flask(__name__)

a = 'Офтальмолог'
target_name = 'Иванов И.И.'
direction_elem = ['Офтальмолог', 'Дерматология', 'Лучевая диагностика', 'Женское здоровье',
'Ортопедия', 'Аптечная служба', 'Экстренная помощь', 'Внутренние болезни', 'Лечение ран']
doctor_elem = ['Иванов И.И.', 'Петров П.П.']
# ...
@app.route('/login/', methods=['post', 'get'])
def login():
    if request.method == 'POST':
        user = request.form.get('username')
        email = request.form.get('email')
        phone_number = request.form.get('phone_number')

        direction = request.form.get('direction')

        direction = direction_elem[int(direction)-1] - 1
        # запрос к данным формы
        doctor = request.form.get('doctor')

        doctor = doctor_elem[int(doctor)-1] - 1

        time1 = request.form.get('time1')
        time2 = request.form.get('time2')

        data1 = request.form.get('calendar1')
        data2 = request.form.get('calendar2')
        """
        user = 'kkk'
        email = 'aa@aa.ru'
        phone_number = '+7'
        direction = 'Офтальмолог'
        doctor = 'Иванов И.И.'
        data1 = '2020-02-12'
        data2 = '2020-02-15'
        time1 = '10:00'
        time2 = '17:00'
        """
        print(user, email, phone_number, direction, doctor, data1, data2, time1, time2)
        Thread(target=take_zapis, args=(user, email, phone_number, direction, doctor, data1, data2, time1,
time2)).start()

    return render_template('login.html')

# ...

```

```
if __name__ == "__main__":  
    app.run(host='0.0.0.0', port=4567)
```

Основная программа:

```
from selenium import webdriver  
from selenium.webdriver.support.ui import WebDriverWait  
from selenium.webdriver.support import expected_conditions as EC  
from selenium.webdriver.common.by import By  
from selenium.webdriver.chrome.options import Options  
from selenium.webdriver.common.keys import Keys  
from selenium.webdriver.common.action_chains import ActionChains  
from webdriver_manager.chrome import ChromeDriverManager  
import time  
import traceback  
from email_send import send_massege  
import datetime  
WINDOW_SIZE = "1920,1080"  
chrome_options = Options()  
chrome_options.add_argument("--headless")  
chrome_options.add_argument("--window-size=%s" % WINDOW_SIZE)  
driver = webdriver.Chrome(ChromeDriverManager().install(), chrome_options=chrome_options)  
# driver = webdriver.Chrome(ChromeDriverManager().install())  
  
driver.implicitly_wait(20)  
  
# -----  
def specialty_selection(a):  
    # print(driver.find_elements_by_tag_name("h3"))  
    way_elem = WebDriverWait(driver, 30).until(EC.visibility_of_element_located(  
        By.XPATH, f"//h3[contains(text(), '{a}')]"))  
    way_elem.click()  
    driver.switch_to.frame(driver.find_element_by_xpath("./iframe"))  
    # print(driver.find_elements_by_tag_name("button"))  
    time.sleep(2)  
    tmp = driver.find_element_by_tag_name("button")  
    tmp.click()  
    # driver.switch_to.default_content()  
  
# -----  
def doctor_choice(doctor_name):  
    # driver.switch_to.frame(driver.find_element_by_xpath("./iframe"))  
    dropdown_menu = WebDriverWait(driver, 30).until(EC.visibility_of_element_located(  
        By.XPATH, './span[@data-hook="dropdown-select-label"]'  
    ))  
    dropdown_menu.click()  
    doctor_elem = WebDriverWait(driver, 30).until(EC.visibility_of_element_located(  
        By.XPATH, f"//li[contains(text(), '{doctor_name}')]'  
    ))
```

```

doctor_elem.click()

#options = webdriver.ChromeOptions()
#options.add_argument('--headless')

# -----
def time_choice(time_start, time_finish):
    try:
        elem = driver.find_elements_by_xpath('//*[div//span[@class="ng-binding" and not(@data-hook)]]')
    except:
        return 0
    print(elem)
    if len(elem) > 0:
        for i in elem:
            if time_start <= i.text <= time_finish:
                i.click()
        return i
    return 0
else:
    return 0

# -----
def data_choice(data_start, data_finish, time_start, time_finish):
    month = str(datetime.date.today())
    elem = [month]
    time = 0
    while elem[-1] <= data_finish:

        elem = driver.find_elements_by_xpath('//*[td//td[@data-date]]')
        # print(elem[0].get_attribute('data-date'))

        for i in range(len(elem)):
            elem[i] = elem[i].get_attribute('data-date')
        elem = sorted(list(set(elem)))

        for time_i in elem:
            print(time_i)
            if data_start <= time_i <= data_finish:
                dropdown_menu = WebDriverWait(driver, 30).until(EC.visibility_of_element_located((
                    By.XPATH,
                    f'//*[td[@data-date="{time_i}"]'
                )))
                dropdown_menu.click()

            time = time_choice(time_start, time_finish)
            if time:
                break
            target_elem = WebDriverWait(driver, 30).until(EC.visibility_of_element_located((
                By.XPATH, f'//*[div[contains(text(), "Показать месяц")]']
            )))
            target_elem.click()

```

```

else:
    driver.find_element_by_xpath("//*[@aria-label = "next month"]').click()

if time:
    break
return i, time.text

# -----
def go_to_registration():
    dropdown_menu = WebDriverWait(driver, 30).until(EC.visibility_of_element_located((
        By.XPATH,
        '/html/body/main/div/bks-app/div/div[1]/boost-calendar-page/div[2]/boost-visitor-
        sidebar/section/div[1]/div/button/boost-next-step-label/span'
    )))
    dropdown_menu.click()

# -----
def reception_registration(user, email, phone_number):
    elem_2 = driver.find_elements_by_tag_name("input")

    elem_2[0].send_keys(user)
    elem_2[1].send_keys(email)
    elem_2[2].send_keys(phone_number)

    button = driver.find_element_by_xpath('//*[div[@class="sidebar"]//span[@data-hook="next-step-label"]')
    button.click()

# -----
def take_zapis(user, email, phone_number, direction, doctor, data_start, data_finish, time_start,
time_finish):
    while True:
        try:
            driver.get("https://zhenyanovokshanov1.wixsite.com/klinika/blank-5")
            specialty_selection(direction)
            doctor_choice(doctor)
            a, b = data_choice(data_start, data_finish, time_start, time_finish)

            go_to_registration()
            reception_registration(user, email, phone_number)

        except Exception as e:
            print("Ошибка:\n", traceback.format_exc())
            print("Время занято")
        else:
            print("OK")
            send_massege(email, b, a, user, doctor)
            break
    time.sleep(10)

```

Программа отправки сообщений:

```
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.image import MIMEImage
def send_massege(addr_to, time, day, user, doctor):
    addr_from = "*****"

    password = "*****"

    msg = MIMEMultipart()
    msg['From'] = addr_from
    msg['To'] = addr_to
    msg['Subject'] = 'Запись на прием'

    body = f"Уважаемый, {user}! Вы записаны на прием к {doctor} на {time}-{day}"
    msg.attach(MIMEText(body, 'plain'))

    smtpObj = smtplib.SMTP_SSL('smtp.mail.ru', 465)
    smtpObj.login(addr_from, password)
    smtpObj.send_message(msg)
```

Почта и пароль были скрыты по понятным причинам!