

Proiect la Ingineria Software

-CV Web App-

Studenta: Vasilache Maria

Grupa: 30238

Profesor coordinator:

Luminita Marghescu

-2022/2023-

1.Despre Aplicatie:

Aplicatia este destinata utilizatorilor care isi doresc sa isi genereze un CV pentru urmatorul loc de munca.

E o aplicatie interactive si atractiva, punand in valoare noile tendinte in tehnologie.

Accentul este pus pe inovare, de aceea se foloseste pentru a atrage mai multi utilizatori, cod QR pentru navigare usoare intre pagini, butoane cu denumiri intuitive.

Feedback-ul este important, de aceea avem un formular, pentru utilizator, in care poate adauga atasamente cu CV generat, putand lasa comentarii.

2.Functionalitate

- a. Pagina pentru a crea un nou CV, vizualizarea acestuia si generarea CV-ului in format PDF
- b. Vizualizarea CV-ului create in format pdf
- c. Scanare cod QR si redirectarea catre Europass.ro
- d. Trimitere mail cu feedback si atasament in format PDF
- e. Download CV-ului existent
- f. Redirectarea catre pagina de start

3. DESIGN PATTERNS: Observer

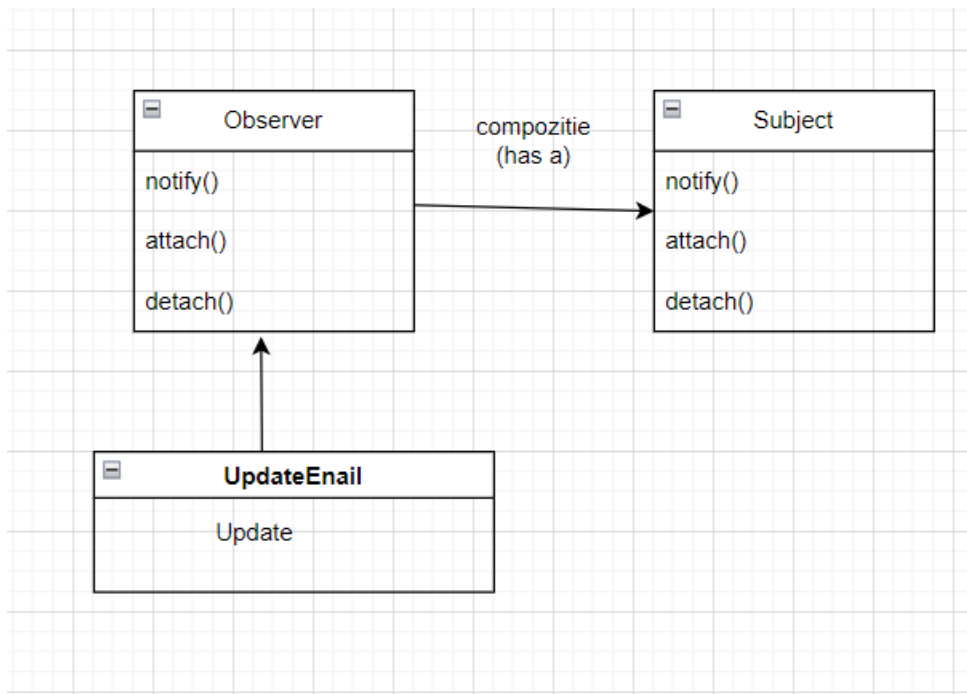
Observatorul este un model de design comportamental care vă permite să definiți un mecanism de abonament pentru a notifica mai multe obiecte despre orice evenimente care se întâmplă cu obiectul pe care îl observă.

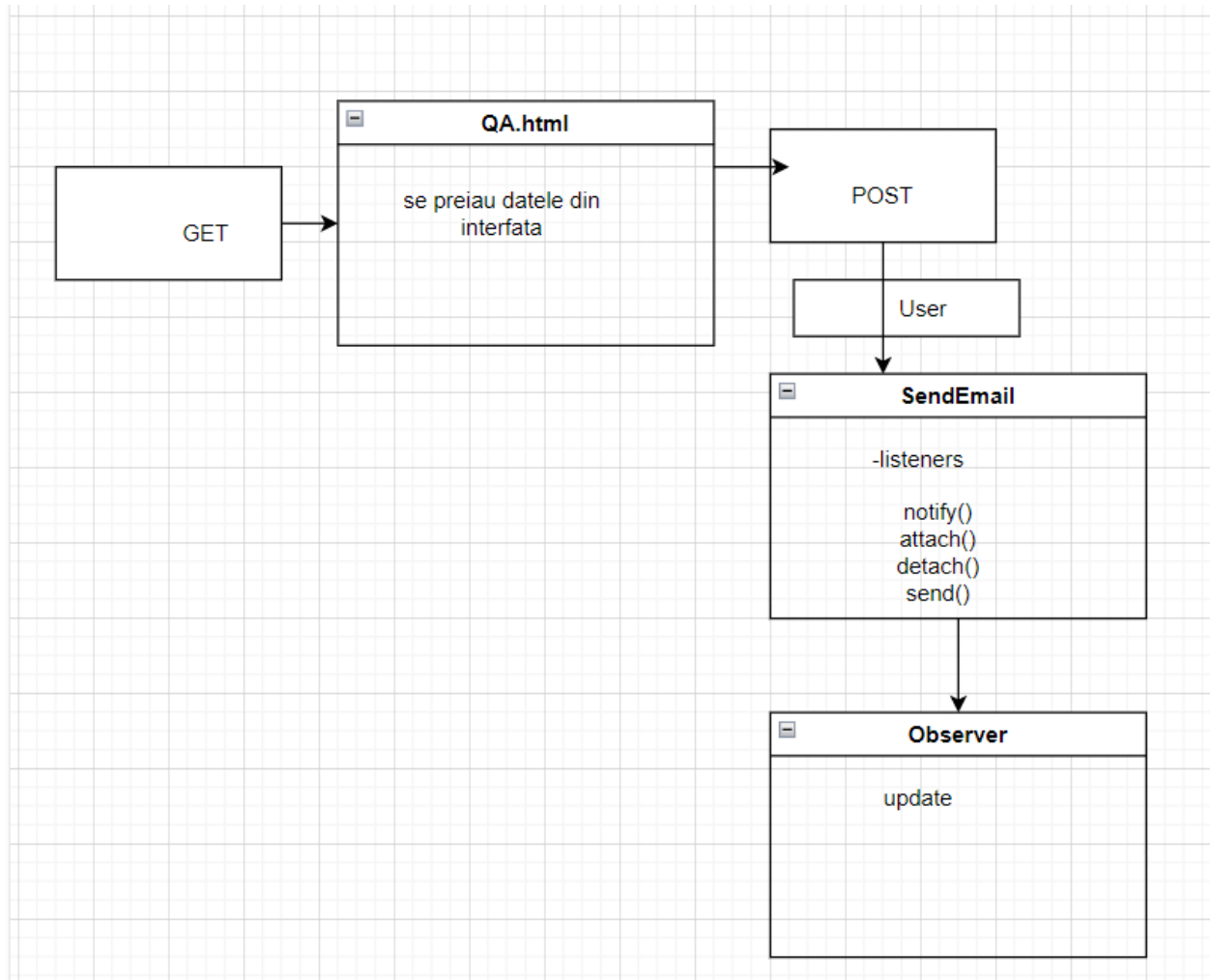
În proiectul meu, am folosit acest pattern pentru a trimite e-mail-uri atât user-ului care trimite e-mail-ul, cât și adminului.

Astfel, la schimbarea clasei Email, prin modificarea atributului trimis, se va trimite o notificare paginii principale pentru a anunța trimiterea cu succes a e-mail-ului.

Atunci când se detectează mesajul de trimis, atunci se va trimite o notificare, și toate instanțele vor trimite mai departe conținutul, generând mesajul propriu-zis.

Implementarea propriu-zisă se găsește în **observer.py**, iar apelarea acesteia are loc în metoda **observer**, din **qaForm.py**.





4. Limbaje de programare folosite

Pentru implementarea Frontend-ului, am folosit **JS**, **CSS**, si framework-ul **Bootstrap**.

Pentru Backend, am folosit limbajul de programare **Python**.

5. Baza de date

Pentru coexiunea aplicatiei la baza de date, am folosit XAMPP, phpMyAdmin.

Aici am create o baza de date numita **cv**, in care sunt tabele pentru preluarea datelor si punerea acestora in pagina pentru prezentarea unui CV by default. De asemenea, printre celelalte tabele care sunt necesare pentru functionarea aplicatiei, am create un tabel(**Account_emailaddress**), in care se mentin toti userii exxistenti, impreuna cu rolul acestora

Tabelul **account**: (detalii CV -by default)

Showing rows 0 - 0 (1 total, Query took 0.0000 seconds)

`SELECT * FROM `account``







Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all Number of rows: 25 Filter rows: Search this table

Extra options

yourName	yourProfession	yourBio	yourCountry	socialContact	yourContact	yourBirthday	yourSkills	yourHobbies	yourCerts	yourEdu	eduYear	yourWork	workYear	yourProject	projectLink
Vasilache Maria	Studenta	Studenta la Universitatea Tehnica Din Cluj-Napoca	Cluj-Napoca	fab fa-github, fab fa-linkedin, fab fa-facebook	https://github.com/vasilachemaria20 https://linkedin.com/vasilachemaria20 https://facebook.com/vasilachemaria20	1/6/2002	HTML/CSS, JavaScript, PHP, MySQL, C, Python	Voluntariat, Limbaje de programare	Technical University Certificate	Technical University Certificate	14/12/2016	11Digits - Web Developer	11/11/2022	Java	https://github.com/vasilachemaria20 /internship

Account_emailaddress (conturile existente)

				id	email	verified	primary	user_id
<input type="checkbox"/>	 Edit	 Copy	 Delete	1	vasilachemaria20@yahoo.com	0	1	1
<input type="checkbox"/>	 Edit	 Copy	 Delete	2	dranga@yahoo.com	0	2	2

6. Implementare

1. Clasa **CV**

- are implementata metoda SendMail

-foloseste clasa EmailForm

2. Fisierul **Files** – contine toate fisierele pe care le vom updata, noi

3. **Pages**- clasele necesare pentru a prelua date din html(templates)

→data.py = avem datele pe care le vom pune in cv(pentru CV-ul Demo)

```
from mysqlP import *

mycursor = connector()

titleCV = "CV"

# type here your name and surname
yourName = mycursor.execute("SELECT yourName FROM cv")

# type here current profession
yourProfession = mycursor.execute("SELECT yourProfession FROM cv")

# type here your bio
yourBio = mycursor.execute("SELECT yourBio FROM cv")

# type here your current location
yourCountry = mycursor.execute("SELECT yourCountry FROM cv")

# if you want add another social media account, add here icon code of it
socialContact = mycursor.execute("SELECT socialContact FROM cv").split(",")

# type here some contact
yourContact = mycursor.execute("SELECT yourContact FROM cv").split(",")

# type here your birthday and (your age)
yourBirthday = mycursor.execute("SELECT yourBirthday FROM cv")

# type here your skills
yourSkills = mycursor.execute("SELECT yourSkills FROM cv").split(",")
```

→forms.py = preia datele din formularul de la QA

```
pages > form.py > ...
1  from django import forms
2
3  class EmailForm(forms.Form):
4      recipient = forms.EmailField()
5      message = forms.CharField(widget=forms.Textarea)
```

→urls.py = redirectare spre pagini, cu functii pentru post

```

pages > urls.py > ...
1  # pages/urls.py
2  from django.urls import path
3  from .views import newCV
4  from .views import myCv
5  from .views import qr
6  from .views import observer
7
8  from .views import HomePageView
9
10 urlpatterns = [
11     path('', HomePageView.as_view(), name='home'),
12     path('cv/', newCV),
13     path('myCV/', myCv),
14     path('qr/', qr),
15     path('qaa/', observer),
16
17 ]
18

```

→ qrPage = clasa pentru redirectare caree pagina cu Europass, se genereaza codul QR

```

from django.shortcuts import render
import qrcode
import qrcode.image.svg
from io import BytesIO

def qr(request):
    context = {}
    if request.method == "POST":
        factory = qrcode.image.svg.SvgImage
        img = qrcode.make(request.POST.get("qr_text", ""), image_factory=factory, box_size=20)
        stream = BytesIO()
        img.save(stream)
        context["svg"] = stream.getvalue().decode()

    return render(request, "QR/Qrcode.html", context=context)

```

→ qaForm.py = clasa care prei date din pagina QA Form si trimite mesaj. In aceasta metoda preluandu-se si atasamentul care va fi trimis prin email.

```

pages > qaForm.py > ...
9 from observer import Observer, UpdateEmail
10
11 def observer(request):
12     obj1 = Observer('Email 1')
13     view1 = UpdateEmail()
14     #atasez observator
15     obj1.attach(view1)
16     #modific email (s-a completat formularul)
17     obj1.data = newValue() #noul email, prin notify(), apelez functia de send mail, pentru ce s-a generat
18
19 #preiau datele din interfata
20 def newValue(request):
21     if request.method == 'POST':
22
23         form = EmailForm(request.POST)
24         # check if data from the form is clean
25         if form.is_valid():
26             cd = form.cleaned_data
27             message = cd['message']
28             emm=cd['recipient']
29             message= message+' '+ 'maria.vasilache02@gmail.com'+ ' '+ emm
30             #trimitere mail
31
32             server=smtplib.SMTP_SSL("smtp.gmail.com", 465)
33             server.login("maria.vasilache02@gmail.com", "qgrqmaoreadfcspz")
34             server.send_message(message)
35             messageSent = True

```

→ getDetaliinewCV.py: clasa in care se vor prelua toate datele primite de la utilizator 😊

-avem functii pentru preluarea datelor din baza de date:

```

def data():
def getSocials(shema, link):
def getSkills(yourSkills):
def getListWithYear(shema, year):
def getListWithLink(shema, link):
def getSocial(shema, link):

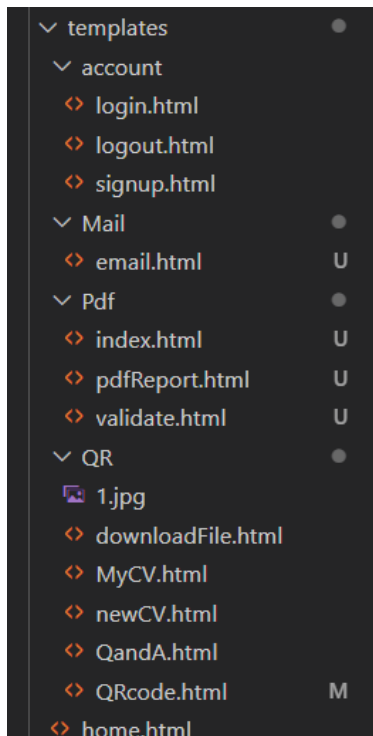
```

→ newCV.py: redirectarea catre pagin in care avem template-ul pentru CV

→ downloadCV.py: se va descarca cv-ul model

→ pdf.py: genereaza un pdf, in urma preluarii datelor din interfata.

4. Folder-ul **templates**, toate paginile html pe care le vom folosi:

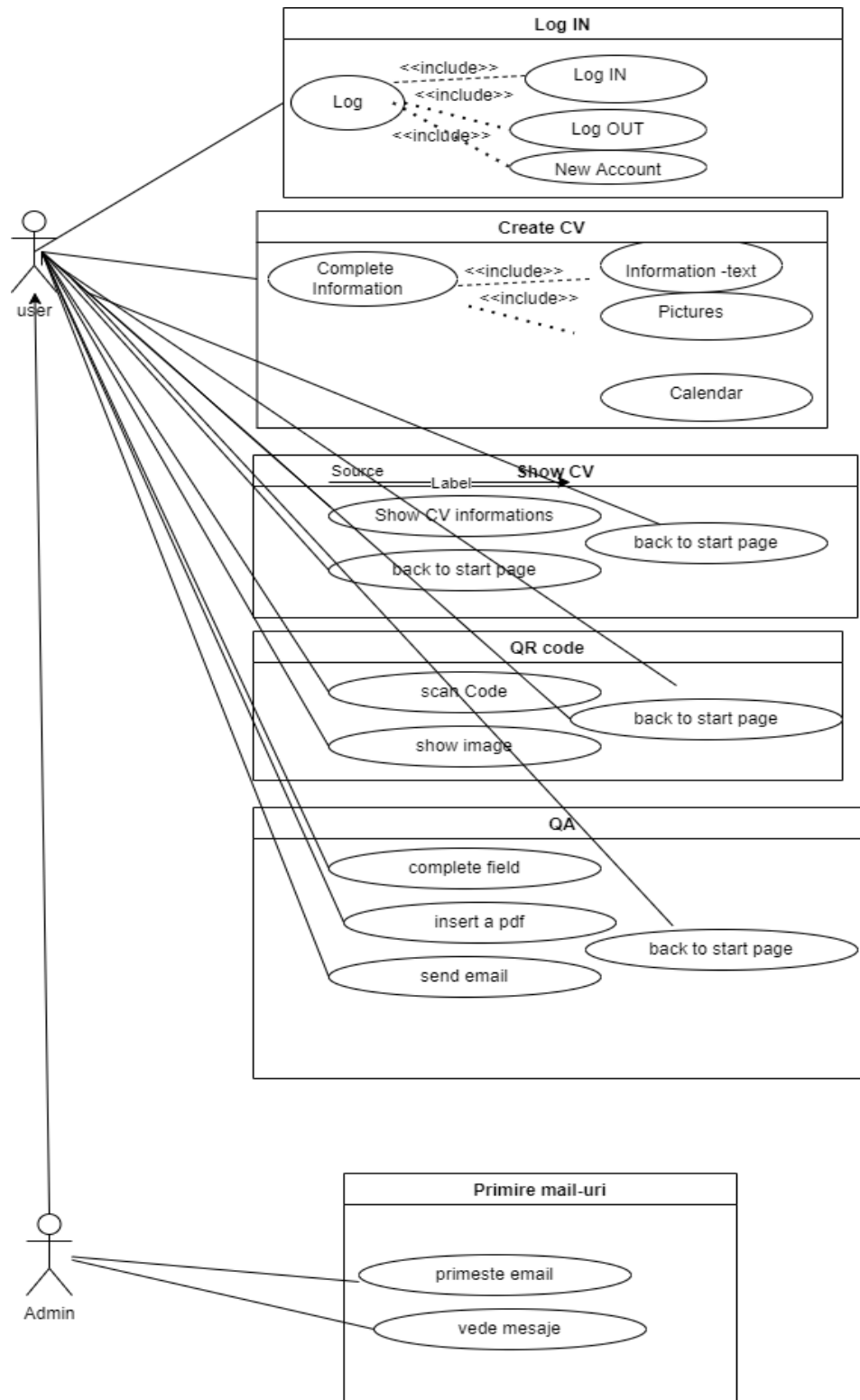


5. Folder-ul Files/page/static = contine css, js si imaginile pe care le vom pune in cv

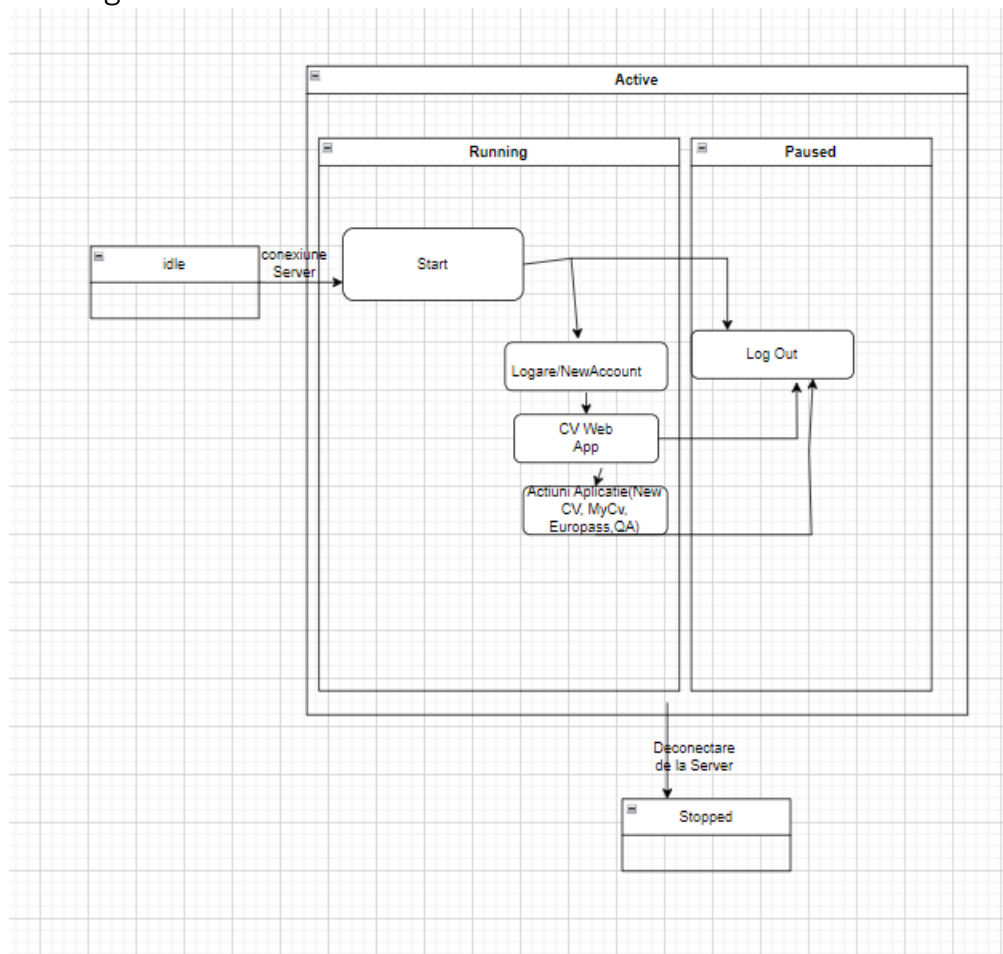
6. Folder config-> aici avem toate link-urile, in **urls.py**

7. Diagrame

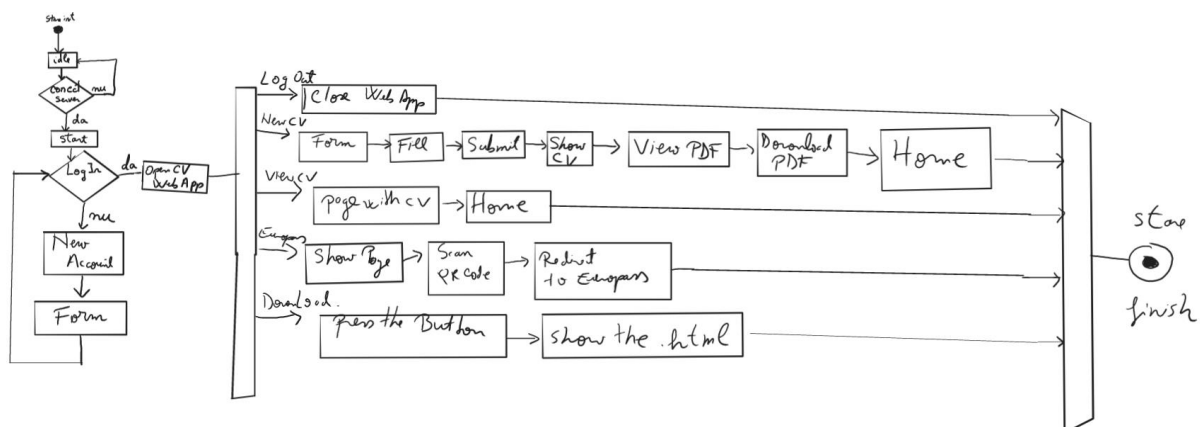
7.1 Use-Case



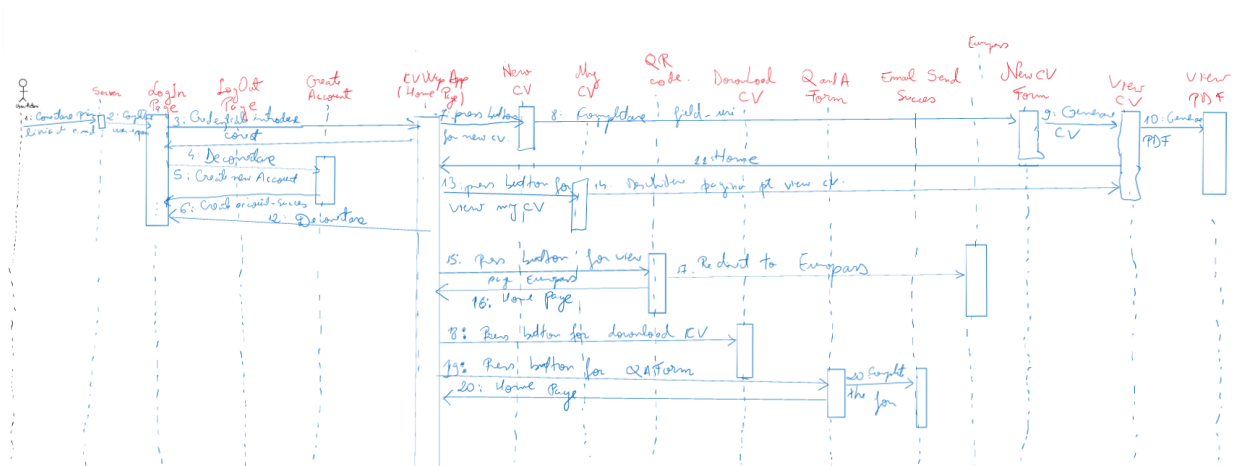
7.2 Diagrama de Stare



7.3 Diagrama de activitate



7.4. Diagrama de secventa



8. Functionare

8.1. Pagina de start

Se porneste aplicatia din linia de comanda cu urmatoarele comenzi:

→python -m pipenv shell

→python manage.py runserver

→In browser: <http://127.0.0.1:8000/>

Actiuni pagina start:

- (1) Numele utilizatorului conectat
- (2) Buton de Log OUT
- (3) Butoane: NewCV(**creez propriul CV**), MyCV(**un exemplu de CV**), Europass QR(**cv-ul Europas, accesay cu codul QR**), Download(**descarcarea CV-ului**)
- (4) Butoane pentru miscarea imaginilor de prezentare

1. NewCV

(1) [Home](#)

Create your first CV

Personal details

Fill in the fields with your personal data

First Name
eg: Maria

Last Name
eg: Vasilache

Your Profession
eg: Student

Your current job
eg: UTCN

Profile photo

Please attach your picture

Alege fisierul | Nu ai ales niciun fisier

Your Country

Choose country and city

Birth date

Date

11/08/2022

Education

Fill in the fields with
University Name

UTCN

Date of graduation

11/08/2022

Submit

- (1) Butonul pentru revenire la pagina de start
- (2) Campuri de completat -Field
- (3) Attachment
- (4) Drop Down
- (5) Selectare data(dintr-un calendar)
- (6) Buton pentru a trimite formularul

2.1 Dupa apasarea butonului Submit

The screenshot shows a CV web application interface. At the top, there is a navigation bar with a 'Home' button (1) and a date '11/10/2022'. The main profile section displays the user's name 'Emi Dranga', profession 'Driver', and current job 'The Current Job: Heritage'. Below this, there is a section for 'Country: Romania/Iasi' and social media links (2). The CV is divided into several sections: 'Skills' (Java, VHDL), 'Hobbies' (driver, swing), 'Education' (UTCN, 11/01/2022), 'Work' (Bosch, 11/17/2022), and 'Projects' (Java, 1). At the bottom, there is a 'Create PDF' button (4).

(1) Home

Name: Emi Dranga
Profession: Driver
The Current Job: Heritage

Country: Romania/Iasi

11/10/2022

Skills

- 1 Java
- 2 VHDL

Hobbies

- 1 driver
- 2 swing

Education

- 1 UTCN 11/01/2022

Work

- 1 Bosch 11/17/2022

Projects

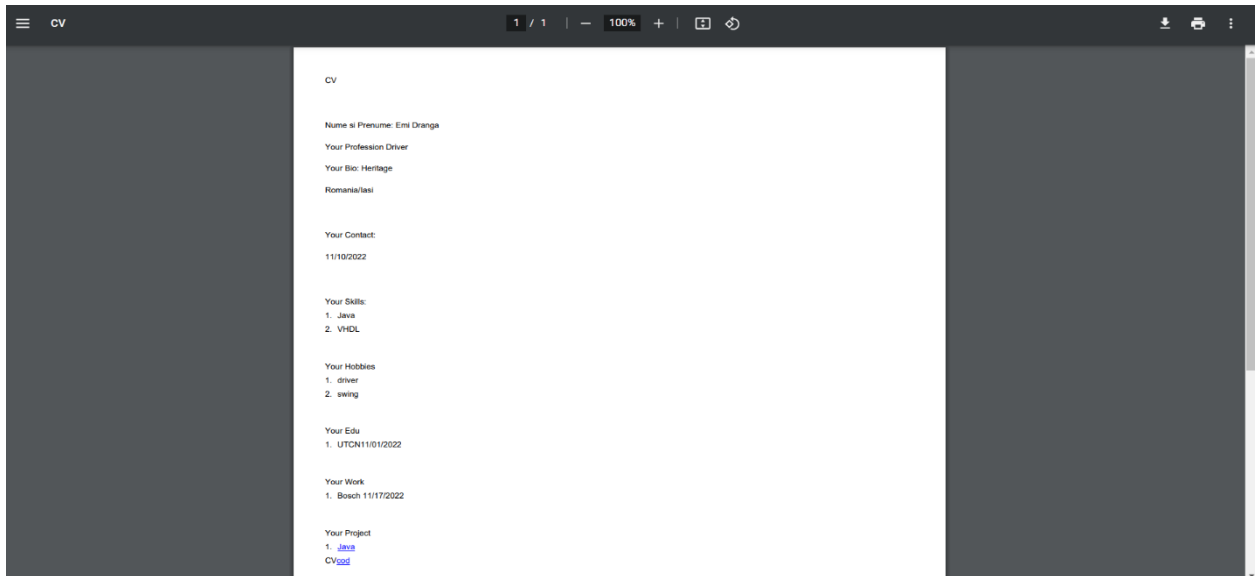
- 1 Java

Create PDF

- (1) Buton pentru revenire la pagina principala
- (2) Redirectionare catre paginile sociale
- (3) Redirectionarea catre GitHub- proiect personal
- (4) Generare de pdf, cu datele preluate din formular

2.2 Create PDF

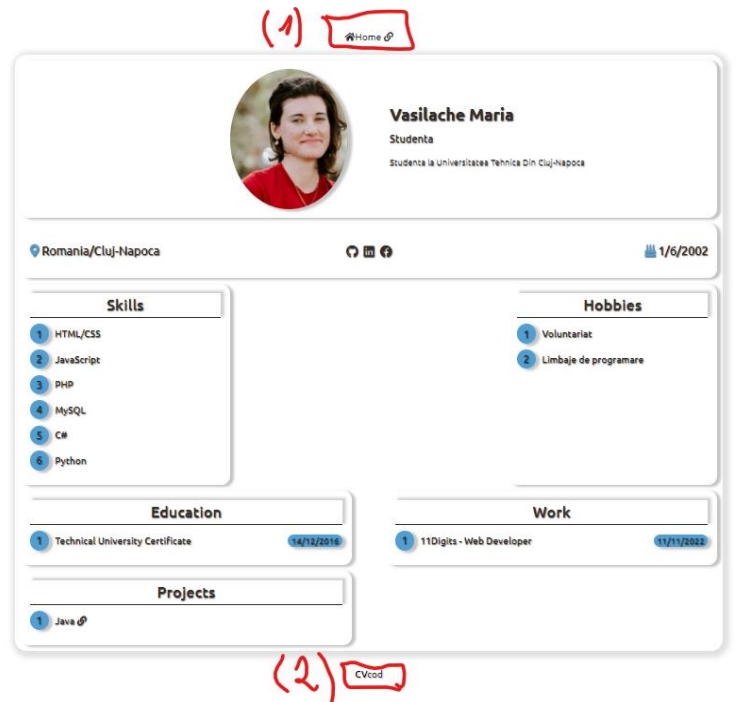
Dupa apasarea butonului de Create PDF



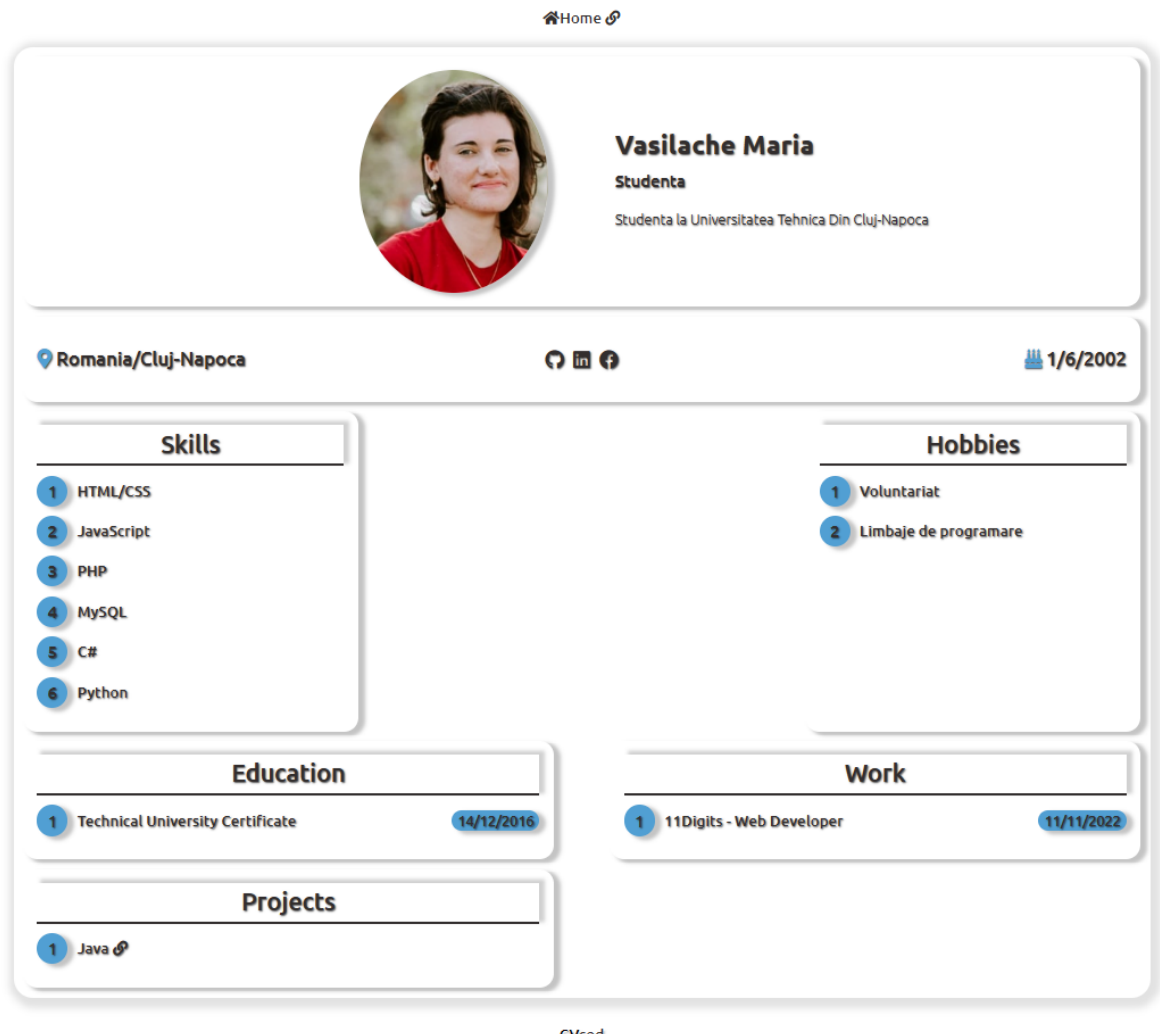
2. NewCV

(1) Revenire la pagina de start

(2) Redirectionarea la pagina de Git,
cu codul sursa



3. My CV



Pentru generarea acestei pagini, s-au folosit datele existente in baza de date.

In [QR/MyCV.html](#): avem interfata care va afisa

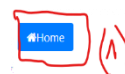
In [Pages/data.py](#) : preluarea datelor din baza de date

In [Pages/mysqlP.py](#): conexiunea cu baza de date

4. Europass QR

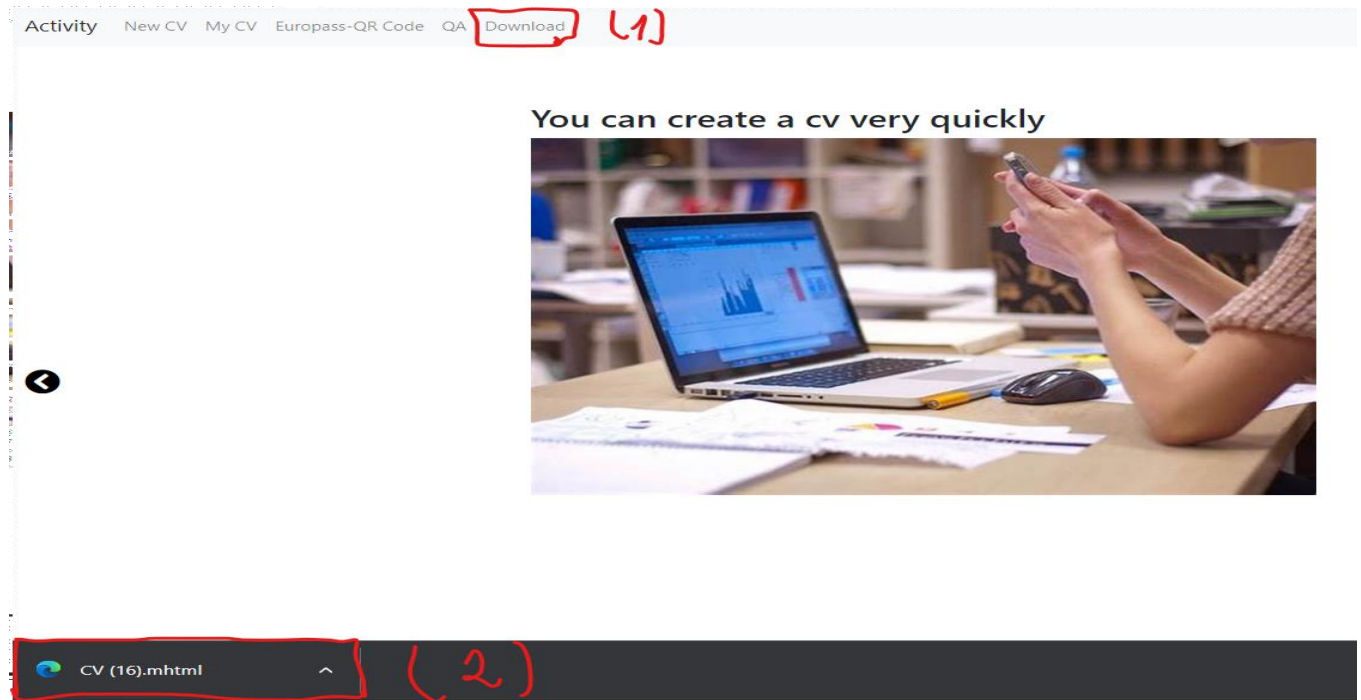
(1) Buton reveire la pagina de start

(2) Scanare cod QR



(2)

5. Download Button:



Cand se apasa butonul, se descarca CV-ul

9. Proiect-cod sursa

Se poate accesa codul incarcat pe GitHub, acesta gasindu-se la urmatorul link:

https://github.com/vmaria2002/proiect_IS

10. Observatii

Pornire aplicatie:

```
D:\An3\IS\P1>python -m pipenv shell
```

```
(P1-qv8LNru9) D:\An3\IS\P1>python manage.py runserver
```

Preluare fisiere media:

\pages\static\img – de aici se va prelua poza

\Files – de aici se va prelua cv-ul

Conturi:

user: dranga@yahoo.com -> Maria.12*ab

admin: vasilachemaria20@yahoo.com -> 12

Bibliografie

1. <https://getbootstrap.com/docs/5.0/forms/overview/>
2. <https://learndjango.com/tutorials/django-email-contact-form>
3. <https://www.geeksforgeeks.org/generate-qr-code-using-qr-code-in-python/>
4. <https://stackoverflow.com/questions/20033712/html-img-src-wont-load-my-images>
5. <https://stackoverflow.com/questions/70669746/python-how-to-call-a-function-with-django>
6. <https://www.bing.com/search?FORM=ALBN01&PC=ATAL&PTAG=ATAL00000028&q=create%20a%20pdf%20from%20html%20python>
7. <https://pythonguides.com/get-data-from-get-request-in-django/>
8. https://www.youtube.com/watch?v=AQrsjt4yyrw&ab_channel=GreatAdib
9. https://www.youtube.com/watch?v=wzZiONbtwiA&ab_channel=MaxGoodridge
10. <https://pythonguides.com/get-data-from-get-request-in-django/>
11. <https://pythonguides.com/django-get-all-data-from-post-request/>
12. <https://pythonguides.com/create-model-in-django/>
13. <https://data-flair.training/blogs/django-database/>
14. <http://jsfiddle.net/mannejkumar/cjpS2/>
15. <https://refactoring.guru/design-patterns/observer/python/example>
16. [Observer method - Python Design Patterns - GeeksforGeeks](#)