Documentul de Proiectare a Soluţiei Aplicaţiei Software (Software Design Document)

08 Ianuarie, 2020

Aplicaţie pentru obținerea unei caricaturi pornind de la o poză

Caricatur App

Facultatea de Automatică şi Calculatoare
Universitatea “Politehnica” Bucureşti

Roluri în cadrul echipei

Apostol ALEXANDRA-SIMONA
Grupa: 342C5
Rol: Project Manager
Email: apostol.alexandra.simona@gmail.com

Crîșmaru VALENTIN
Grupa: 342C5
Rol: Team Leader
Email: valentin.crismaru@gmail.com

Irimia TUDOR
Grupa: 341C5
Rol: Developer
Email: tudor_irimia@outlook.com

Smaranda ALEXANDRU
Grupa: 342C5
Rol: Developer
Email: alex.sma280797@gmail.com

Mazilu ANA-MARIA
Grupa: 342C5
Rol: Tester
Email: mazilu.ana01@gmail.com

Cuprins

1. Scopul proiectului

Scopul acestui proiect este dezvoltarea unui program software care să primească o poză (un selfie) și să obțină pe baza acesteia o caricatură .

2. Conţinutul documentului

Documentul de faţă este alcătuit din patru secţiuni principale:

1. Modelul datelor - prezintă principalele structuri de date folosite
2. Modelul arhitectural şi modelul componentelor - prezintă şabloanele arhitecturale folosite, arhitectura sistemului şi descrie componentele arhitecturii
3. Modelul interfeţei cu utilizatorul – prezintă interfaţa cu utilizatorul şi succesiunea ferestrelor acesteia
4. Elemente de testare – prezintă componentele critice şi alternative de proiectare a acestora

3. Modelul datelor

3.1. Structuri de date

Date vizibile în cadrul întregii aplicaţii. Modificările aduse oricărei dintre aceste date vor fi vizibile pentru toate celelalte componente.

Aici avem următoarele:
butoane - fiecare având un event listener corespunzător

- show image: folosind acest buton putem vedea imaginea iniţială (cea înainte de caricatură).
- cartoon: buton care aplică şi afişează filtrul de cartoon pe imaginea de dinainte. Apăsând în acest moment butonul menţionat anterior, show image, se revine la imaginea iniţială.
- canny filter pentru edge detection.
- dog şi flowerpower sunt butoane ce aplică filtre de pe Snapchat
- oldfilter: buton ce aplică un filtru de imagine, nu de faţă, şi face ca imaginea să arate veche (anii 80, spre exemplu).
- face detection: buton care creează un dreptunghi în jurul feţei detectate.

3.2. Structuri de date temporare

Nu se utilizează structuri de date temporare cu rol important sau presupunând un consum semnificativ de resurse de memorie.

3.3. Formatul fişierelor utilizate

Aplicaţia foloseşte fişiere de tip **.java** şi **.xml**.

Fişierele folosite sunt **activity_main.xml** în care am creat un welcome text şi butoanele utilizate în cadrul aplicaţiei şi **mainactivity.java** care preia date din activity_main.xml.

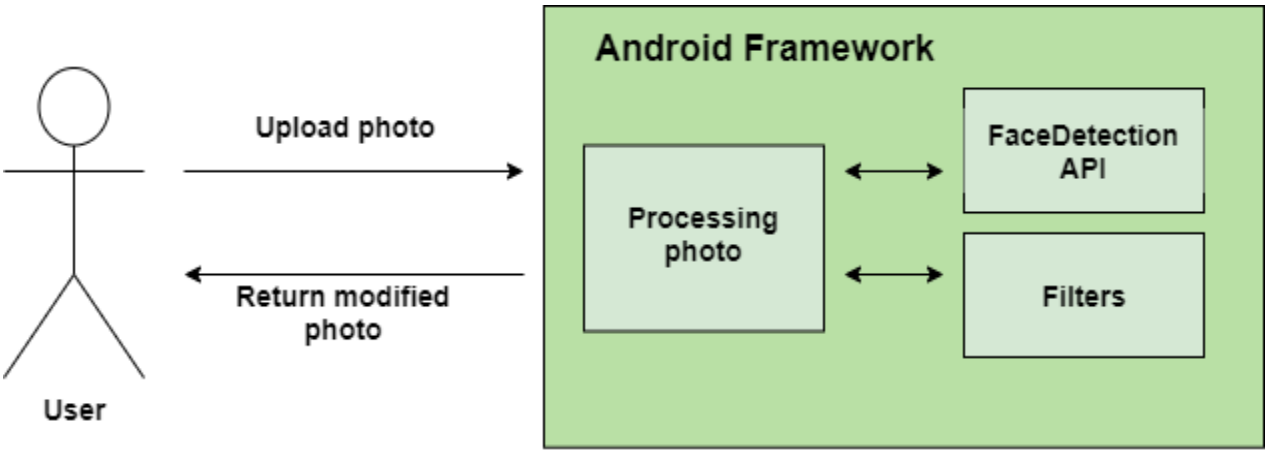
3.4. Structuri de date de legătură

Fişierul MainActivity.java mai comunică cu urmatoarele fişiere: CartoonFilter.java (acesta mai comunică şi cu Pixel.java), JCanny.java (acesta comunică cu Gaussian.java, Sobel.java şi ImageUtils.java). Prin aceste “comunicări” sunt accesate API-urile de filtre folosite în MainActivity.java.

4. Modelul arhitectural şi modelul componentelor

4.1. Diagrama de arhitectură

Diagrama de arhitectură de mai jos descrie componentele aplicaţiei şi legăturile dintre acestea.



Filters: Cartoon Filter
Gaussian Filter
jCanny Filter
Pixel Filter
ImageUtils Filter
Sobel Filter

4.2. Descrierea componentelor

Aplicaţia constă din următoarele module interconectate:

- **Modulul Processing Photo**
  Este responsabil cu primirea input-ului de la utilizator, prelucrarea acestuia (prin alegerea diferitelor opțiuni disponibile) și returnarea imaginii inițiale modificată în funcție de cerințele user-ului.
- **Modulul Face Detection API**
  Este responsabil cu detectarea feței în cadrul fotografiei inițiale, nealterată pentru a putea adăuga ulterior filtrele folosite pentru generarea caricaturii. Reprezintă pasul intermediar în proces și este folosit doar în cazul în care nu se aplică filtre de fundal.
- **Modulul Filters**
  Este responsabil cu primirea requesturilor de la modului **Processing Photo**, prelucrarea lor în funcție de filtrul ales și oferirea de imagini modificate pentru requesturile venite de la utilizator.

4.3. Restricții de implementare

Modulele aplicaţiei trebuie să acopere următoarele restricţii de implementare:

- Imaginea folosită în cadrul aplicaţiei trebuie să fie cât mai clară şi să conţină o faţă detectabilă pentru ca filtrele să funcţioneze corect.
- Imaginea trebuie să fie de o anumită dimensiune pentru a nu apărea în spatele butoanelor.
- Android-ul trebuie să fie minim versiunea 5.0.
- Pozele folosite trebuie să existe într-un folder “drawable” pentru a putea fi folosite de aplicaţie.

4.4. Interacțiunea dintre componente

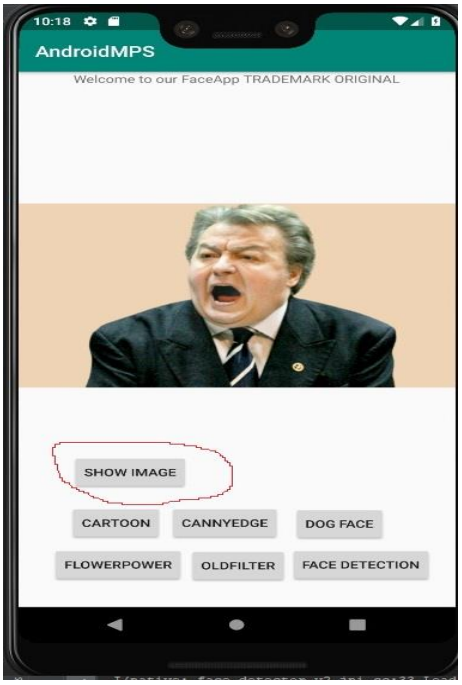
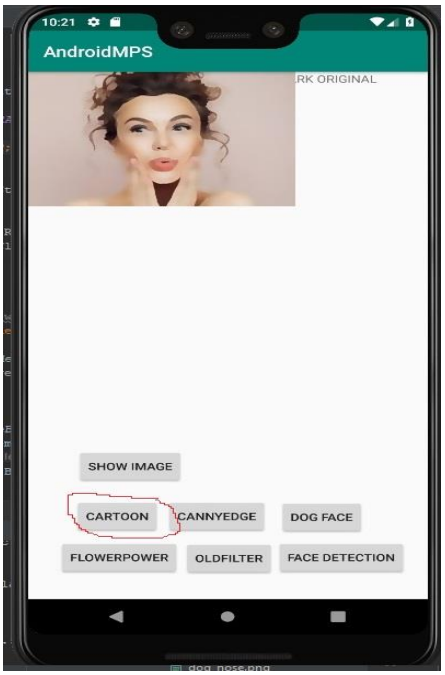
Fişierul MainActivity.java preia date din activity_main.xml :

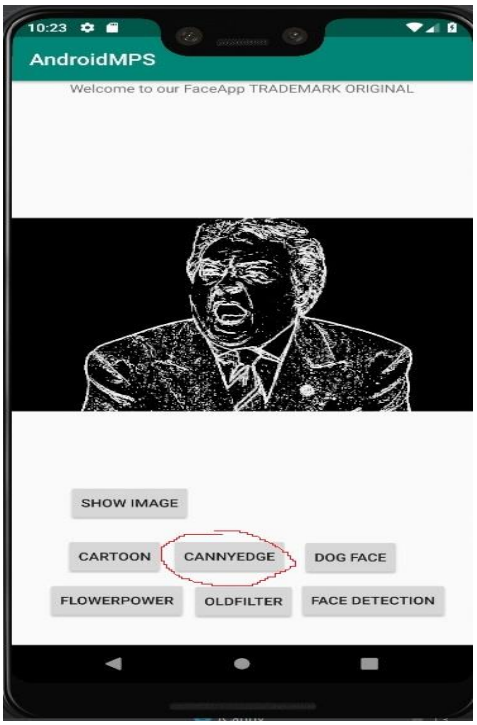
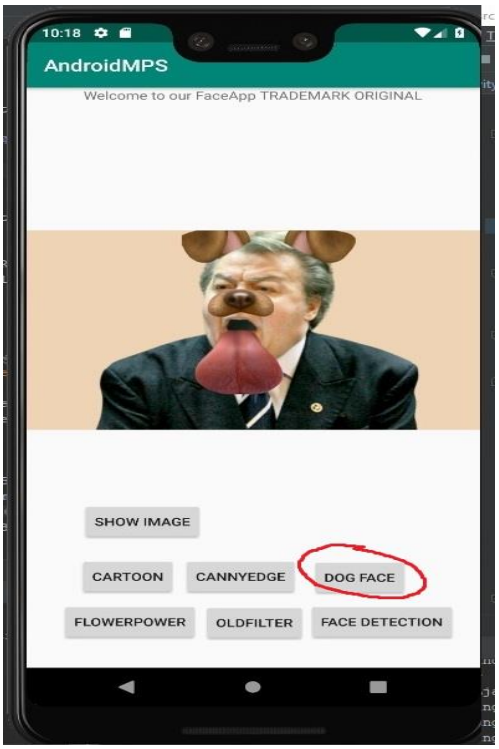
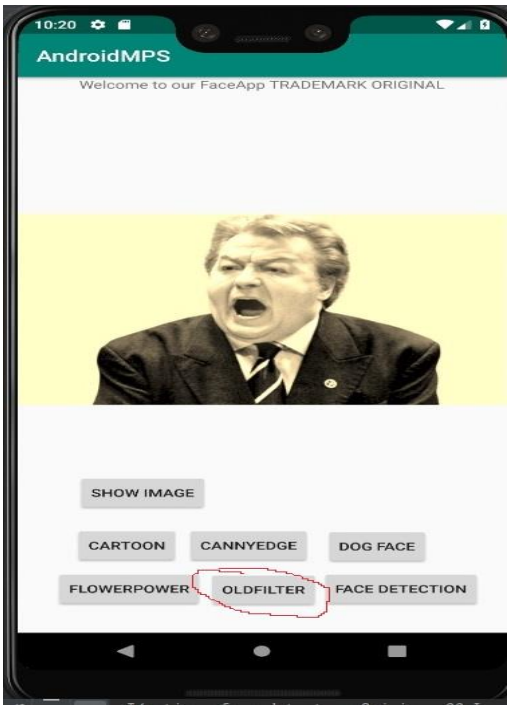
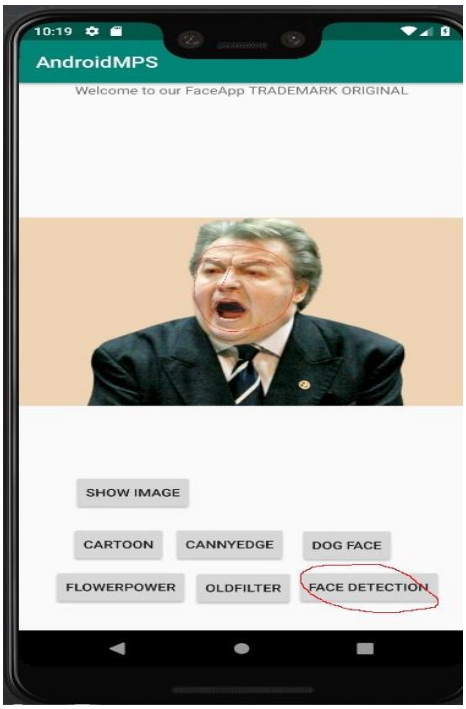
Butoanele create in activity_main.xml conţin mai multe informaţii, printre care un ID, după care butonul este “deschis” în MainACtivity.java. Ulterior, aici există o metodă “setOnClickListener” care apelează metode în funcţie de butonul apăsat.

Fisierul MainActivity.java mai comunica cu urmatoarele fisiere: CartoonFilter.java (acesta mai comunica si cu Pixel.java), JCanny.java (acesta comunica cu Gaussian.java si Sobel.java si ImageUtils.java). Prin aceste “comunicari” sunt accesate API-urile de filtre folosite in MainActivity.java.

5. Modelul interfeţei cu utilizatorul

5.1. Ferestrele aplicaţiei

<table>
<tr><td colspan="2">Aplicaţia are o singură fereastră unde se regăsesc butoanele descrise mai sus.</td></tr>
<tr><td>Pentru butonul show_image care arată în orice moment imaginea iniţială:</td><td>Butonul de cartoon care aplică şi afişează filtrul de cartoon pe imaginea anterioară:</td></tr>
<tr><td> 
</td><td> 
</td></tr>
</table>

Butonul Cannyedge	**Buton pentru filtrul de Snapchat - Dog Face**
	
Butonul pentru Old Filter	**Butonul pentru Face Detection**
	

6. Elemente de testare

Pentru testarea aplicaţiei ne-a interesat în mod special corectitudinea filtrelor folosite.

Astfel, pentru funcționarea corectă a aplicației am testat manual următoarele:

- Filtrul “Cartoon” pe imagini diferite :
    - S-a observat ca la imaginile de mici dimensiuni se procesează mai repede şi aspectul vizual este mai accentuat.
- Filtrul “Cannyedge” pe imagini diferite :
    - Efectul a fost în principiu acelaşi pe toate imaginile folosite ca test.
- Filtrul “Dog Face” pe imagini diferite :
    - S-a observat ca acest filtru depinde de dimensiunile imaginii, deoarece nu se face resize pe urechi, nas şi limba de căţel. Totuşi, elementele feţei sunt detectate bine.
- Filtrul “FlowerPower” pe imagini diferite:
    - S-a observat ca acest filtru depinde de dimensiunile imaginii, deoarece nu se face resize pe coroana de flori. Totusi, coordonatele extreme din partea de sus ale feţei sunt detectate bine.
- Filtrul “OldFilter” pe imagini diferite:
    - Efectul a fost în principiu acelaşi pe toate imaginile folosite ca test.
- Filtrul de Face Detection pe imagini diferite:
    - S-a observat că detectează feţele umane, ignorând feţe de animal.