UNIVERSITATEA BABEŞ-BOLYAI CLUJ-NAPOCA
FACULTATEA DE MATEMATICĂ ŞI INFORMATICĂ
SPECIALIZAREA INFORMATICĂ ROMÂNĂ

LUCRARE DE LICENŢĂ

AI Suport pentru design vestimentar

Conducător ştiinţific
Lect. dr. Mihoc Tudor

Absolvent
Gherasim Georgian

2022

ABSTRACT

The first chapter entitled 'Machine learning' aims to present the model used in the development of the application, more precisely the virtual test of a clothing item, thus making a complex analysis of the algorithm used. Is presented in as much detail as possible global appearance flow based try-on model, the techniques used but also the results obtained in comparison with the other existing models in the literature.

In the second chapter are presented the technologies used in the development process both on the backend and on the frontend.

The third chapter represents the application development process. Exactly like the previous chapter, we structured in two parts: a part for the client in which the elements of the graphical interface are illustrated by images and the one of the server in which the communication between the server and the http requests are presented.

Cuprins

1 Introducere **1**

2 Machine learning **3**

2.1 Definirea problemei 3

2.2 Preantrenarea unui model bazat pe analizator 6

2.3 Extragerea caracteristicilor 11

2.4 Estimarea fluxului de aspect bazat pe stil 11

2.5 Obiective de învățare 13

2.6 Experimente si rezultate 14

3 Tehnologii folosite **19**

3.1 Backend 19

3.1.1 Python,MongoDB,Flask 19

3.2 FrontEnd 22

3.2.1 React,Typescript,MaterialUI 22

4 Dezvoltarea aplicației **24**

4.1 Dezvoltarea aplicației client 24

4.1.1 Interfața autentificare si înregistrare 25

4.1.2 Interfața home page si vizualizare produse 27

4.1.3 Interfața virtual dressing 29

4.2 Dezvoltarea aplicatiei server 30

4.2.1 Cererile HTTP 31

4.2.2 Baza de date 32

5 Concluzii **35**

Bibliografie **36**

Capitolul 1

Introducere

Primul capitol intitulat „Machine learning” își propune să prezinte modelul utilizat în dezvoltarea aplicației, mai exact testarea virtuală a unui articol vestimentar, făcând astfel o analiză complexă a algoritmului utilizat.

În al doilea capitol sunt prezentate tehnologiile utilizate în dezvoltarea backend-ului, cât și a frontend-ului.

Al treilea capitol reprezintă procesul de dezvoltare a aplicației. Exact ca în capitolul anterior, am structurat acest capitol în două părți: o parte pentru client în care elementele interfeței grafice sunt ilustrate prin imagini și cea a serverului în care sunt prezentate comunicarea dintre server și baza de date, cererile http și secvențe de cod.

Formularea problemei

AI Suport pentru design vestimentar este o aplicație web care poate fi utilizată cu scopul de a proba produse vestimentare cât mai sigur și mai rapid posibil. Aplicația are ca scop probarea îmbrăcămintei în format digital si alegerea acesteia în funcție de stil/preferințe fără a necesita deplasarea inutilă la magazin cât reducerea cumpărării unei cantități nenecesare de haine. Utilizatorul poate selecta de pe site îmbrăcămintea dorită apoi încarcă o imagine cu acesta, urmând apoi procesarea de către server a celor două imagini și rezultatul aferent cu cea mai bună portivire a îmbrăcămintei pe corpul persoanei selectate.

Motivație

În ziua de astăzi ce porți te reprezintă. Hainele poartă un rol foarte important în viața noastră de zi cu zi, în fiecare zi ne îmbracam, unii dintre noi pierdem foarte mult timp doar gândindu-ne ce ne-ar sta bine și m-am gandit să vin cu o aplicație care să vină în ajutorul tuturor, un suport pentru design vestimentar folo-

sind inteligență artificială.

Arhitectura si implementare

AI Suport pentru design vestimentar este o aplicație client-server destinată persoanelor dornice să-și schimbe stilul vestimentar și să reducă deplasările inutile in scopul achiziționarii de produse vestimentare. Utilizatorul trebuie să-si selecteze îmbrăcămintea dorită de pe site si încarca o poza cu corpul/silueta acestuia.

Algoritmul de inteligentă artificială adaptează ținuta pe corpul imaginii alese de către utilizator astfel încat să aibă o experientă cât mai apropiată de realitate. Utilizatorul poate decide dacă doreste să achiziționeze articolul selectat adaugandu-l in coșul de cumpărături.

La începerea proiectului am utilizat React, o librarie javascript pentru construirea de interfețe grafice pe partea de client. ReactJS ne permite sa creăm repede un frontend scalabil si ușor de utilizat pentru aplicațiile web. Pe partea de server atât pentru antrenarea ai-ului cat si pentru interacțiunea bazei de date cu clientul vom utiliza python.

Modelul client-server este o structură sau arhitectură care partajează procesarea între furnizorii de servicii, numiți servere și elementele care solicită servicii, numite clienți. Clienții și serverele comunică printr-o rețea de calculatoare, de obicei prin Internet, având suporturi hardware diferite, dar pot rula și pe același sistem fizic. Un server (fizic) rulează unul sau mai multe programe server, care partajează resursele existente cu clienții. Clientul nu partajează niciuna dintre resursele proprii, ci apelează la resursele serverului prin funcțiile server. Pentru comunicarea intre client server am folosit flask, un micro web framework scris în python pentru crearea api-urilor.

Capitolul 2

Machine learning

2.1 Definirea problemei

Proba virtuală bazată pe imagini urmărește să încadreze o îmbrăcăminte dintr-un magazin într-o imagine a unei persoane îmbrăcate. Pentru a realiza acest lucru, un pas cheie este deformarea îmbrăcămintei care alinează spațial îmbrăcămintea țintă cu partile corespunzatoare ale corpului din imaginea persoanei. Metodele anterioare folosite de alte persoane adopta de obicei un model local de estimare al fluxului de aspect. Pot exista probleme la pozitii dificile ale corpului și dezechilibre mari între imaginile cu persoana si hainele.

Pentru a depăsi această limitare, algortimul propus foloseste un model global de estimare a fluxului de aspect. Pentru inceput pentru estimarea fluxului de aspect este adoptată o arhitectura bazată pe StyleGAN. Acest lucru ne va permite să profităm de vectorul global de stil pentru a codifica întreaga imagine și a face fată provocarilor de mai sus.

Pentru a ghida generatorul de flux StyleGAN, pentru a acorda mai multă atenție deformarii locale a îmbracamintei, este introdus un modul de rafinare a fluxului pentru a adauga contextul local. Rezultatele experimentelor acestui model pe un benchmark virtual tryon arata ca metoda aceasta realizeaza o nouă performantă de ultimă generație.

Tranziția de la comerțul offline în magazin la comertul electronic a fost accelerată de blocarea cauzată recent de pandemie. In 2020, vanzarile de comerț electronic la nivel global s-au ridicat la 4.28 trilioane de dolari si se estimează ca va creste la 5.4 trilioane de dolari in 2022. Cu toate acestea cand vine vorba de moda spre deosebire de offline, cumparatori online nu traiesc experienta probarii unui articol vestimentar

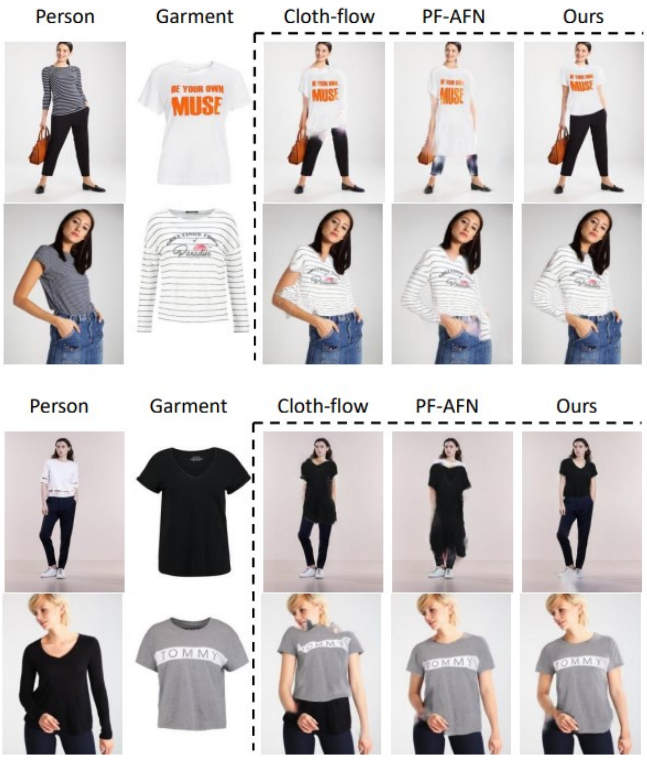

Figura 2.1: Diferentele dintre modelul bazat pe flux de aspect fata de metodele SOTA existente [HSX22]

in vestiar

Pentru a reduce costul de returnare pentru comerciantii online si pentru a oferi cumparatorilor aceeasi experienta offline si online, proba virtuala bazata pe imagini (VTON) a fost studiata intens recent.[YGL21]

Un model VTON își propune sa încadreze o îmbracaminte din magazin in imaginea unei persoane. Un obiectiv cheie al unui model VTON este alinierea imbracamintei din magazin cu partile corespunzatoare ale corpului în imaginea persoanei. Acest lucru se datorează faptului ca îmbracamintea din magazin de obicei nu este aliniată spatial cu imaginea persoanei.2.1

Fară alinierea spatială, aplicarea directă a imaginii avansate de păstrare a detaliilor la translatarea modelelor imaginii[PIE17] pentru a fuziona textura in persoana si imaginea imbracamintei va avea ca rezultat un efect nerealist în imaginea generată in special in regiunile ocluzate si nealiniate.

Metodele anterioare folosite în rezolvarea acestei probleme abordează aceasta problema de aliniere prin deformarea îmbracamintei, adica mai întai deformează îmbracamintea din magazin, care este apoi concatenată cu imaginea persoanei si

introdus intr-un model de traducere imagine in imagine pentru generarea finală a imaginii de încercare.

Mulți dintre ei [YGL21],[XHD18] adoptă un Thin Plate Spline(TPS)[Duc77] bazat pe metoda warping, exploatand corelatia dintre trasaturile extrase din persoana si imaginile vestimentare. Cu toate acestea, analizand lucrarile anterioare, TSP are limitari in manipularea deformarii complexe, de exemplu atunci cand regiuni diferite de îmbracaminte necesită diferite deformari.

Ca urmare, metodele SOTA recente sunt estimarea fluxului de aspect dens pentru a deforma haina. Aceasta implica antrenarea unei rețele pentru a prezice campul de flux dens pe care îl reprezinta deformarea necesara pentru a alinia imbracamintea cu partile corespunzatoare ale corpului.

Cu toate acestea, metodele existente de estimare a fluxului de aspect sunt limitate în deformarea precisa a articolelor de imbracaminte din cauza lipsei de context global. Mai precis toate metodele existente sunt pe baza corespondentei caracteristicii locale, de exemplu caracteristica locala concatenare sau colelare, dezvoltat pentru estimarea fluxului optic.[ADB]

Pentru a estima fluxul de aspect, ei fac presupunerea nerealisă că regiunile corespunzatoare din imaginea persoanei si articolele de îmbracaminte sunt localizate in acelasi câmp receptiv local al extractorului de caracteristici. Cand exista o nealiniere mare intre imbracaminte si partile corespunzatoare corpului (Figura 2.1 Sus) metodele bazate pe fluxul aspectului actual se vor deteriora drastic si vor genera rezultate necorespunzatoare. Lipsa unui context global face, de asemenea, metodele VTON existente bazate pe flux vulnerabile la pozitii dificile/ocluzii(Figura 2.1 jos) atunci cand corespondentele trebuie cautate dincolo de un cartier local.

Acest lucru limiteaza sever utilizarea acestor metode “inthe-wild”, prin care un utilizator poate avea o imagine completa a corpului insusi ca imaginea persoanei pentru a incerca mai multe articole vestimentare. Pentru a depasi aceasta limitare, o noua aparitie globala, modelul de estimare al fluxului este utilizat în rezolvarea problemei noastre.

Mai exact, pentru prima data, o arhitectura StyleGAN este folosita pentru estimarea fluxului de aspect dens.[TKA19] Aceasta difera fundamental de metodele existente care utilizeaza o arhitectura U-Net [ORB15] pentru a pastra contextul spatial local. Folosind un vector de stil global extras extras din intreaga referinta si articole de imbracaminte faciliteaza modelul nostru pentru a surprinde contexul

global.

Cu toate acestea, ridica si o intrebare importanta: poate fi crucial contextul spatial local pentru aliniamentele locale? La urma urmei un singur vector de stil aparent a pierdut contextul spatial local. Pentru a raspunde la aceasta intrebare, observam mai întai că StyleGAN a fost cu succes aplicat sarcinilor locale de manipulare a imaginii fetei, in care vectori de stil diferiti pot genera aceeasi fată din diferite puncte de vedere si diferite forme.[SZ21]

Aceasta sugereaza ca un vector de stil global are contextul spatial local codificat. Cu toate acestea, observam, de asemenea , ca arhitectura StyleGAN este mult mai robusta împotriva alinierilor gresite și ipostaze/ocluzii dificile în comparatie cu U-Net, care este mai slab cand vine vorba de modelarea deformarii locale. Prin urmare introducem un modul de rafinare a fluxului local in generatorul StyleGAN existent.

Concret, modulul nostru de warping bazat pe StyleGAN (W in Fig 2.3) constă din blocuri de deformare stivuite care iau ca intrări un vector de stil global, caracteristici de îmbrăcăminte și caracteristici ale persoanei. Vectorul de stil global este calculat de la cel mai mic plan de rezolutie ale imaginii persoanei si imbracamintei din magazin pentru modelarea contextului global. În fiecare bloc de warping în generator, vectorul de stil global este utilizat pentru a modula canalele caracteristice care preia harta corespunzatoare caracteristicilor de imbracaminte pentru a estima fluxul de aspect. [HSX22]

Pentru a permite estimatorului nostru de flux sa modeleze fluxul aspectului local cu granulatie fina, de exemplu, regiunile bratului si mainii (din Figura 2.6) în fiecare bloc de deformare introducem un strat de rafinare. Acest strat de rafinare deformeaza mai întai harta caracteristică a articolelor de îmbracaminte, care este ulterior concatenată cu harta caracteristicilor persoanei la aceeasi rezolutie si apoi folosit pentru a prezice fluxul de aspect local detaliat.

2.2 Preantrenarea unui model bazat pe analizator

VTON bazat pe imagine 2D poate fi categorizat in metode bazate pe analizator si metode fara analizator. Metodele bazate pe analizator aplica o harta de segmentare umana pentru a masca regiunea de imbracaminte in imaginea persoanei de intrare pentru estimarea parametrilor de intrare.

Masca imaginii persoanei este concatenata cu haina deformata si apoi alimentat intr-un generator pentru generarea de imagini de incercare tinta. Majoritatea metodelor aplica un parser uman pre-antrenat pentru a analiza imaginea persoanei in mai multe regiuni semnatice predefinite de exemplu cap, trunchi si pantaloni. [XHS19]

Pentru o mai buna generare de imagini de incercare, transforma si harta de segmentare pentru a se potrivi cu imbracamintea tinta. Rezultatul transformat al analizei, impreuna cu imbracamintea deformata si imaginea persoanei mascate sunt folosite pentru generarea finala de imaginii de incercare. Dependenta de un parser face ca aceste metode sa fie sensibile la rezultate proaste ale analizei umane care duc inevitabil la rezultate inexacte de deformare si incercare. In schimb metodele fara analizator, in stadiul de inferenta, ia doar ca intrari imaginea persoanei imaginea imbracamintei. [HYL20]

Sunt concepute special pentru a elimina efectele negative induse de rezultatele proaste ale analizei. Aceste metode de obicei antreneaza mai intai un model de profesor bazat pe analizator si apoi un model student fara analizator. Thibaut Issenhuth, Jeremie Mary, and Clement Calauzenes au propus un pipeline care distileaza modelul de deformare a articolelor de imbracaminte si reteaua de generare a probelor folosind perechi de tripleti. Yuying Ge, Yibing Song, Ruimao Zhang, Chongjian Ge, Wei Liu, si Ping Luo au imbunatatit in continuare consistenta ciclului pentru o mai buna distilare.[TIC15]

Metoda folosita in aplicatie este de asemenea o metoda fara analizator. Cu toate acestea metoda folosita se concentreaza pe proiectarea partii de deformare a imbracamintei, unde se propune un nou flux de aspect global bazat pe modul de deformarea al articolelor de imbracaminte.

In comparatie cu VTON bazat pe imagini, VTON 3D ofera o experienta de incercare mai buna (de exemplu permitand sa fi vizualizat cu vederi si ipostaze arbitrare), dar este si mult mai provocator. Majoritatea lucrarilor 3D VTON se bazeaza pe modele 3D parametrice ale corpului uman si necesita scanare 3D seturilor de date pentru antrenament. Colectarea se seturi de date 3D la scara larga este costisitoare si laborioase, punand astfel o constrangere asupra scalabilitati unui model 3D VTON. Cu toate acestea exista 3D VTON care inca genereaza detalii de textura inferioare in comaparatie cu modelele 2D.

StyleGAN pentru manipularea imaginilor. StyleGAN a revolutionat cercetarile

privind manipularea imaginilor in ultima vreme. Aplicarea sa de succes pe imaginea sarcinilor de manipulare adesea datorita adecvarii sale in invatarea unui spatiu latent extrem de dezlegat.

Eforturile recente au fost concentrate pe descoperirea nesupravegheata a semanticii latente. Kathleen M Lewis, Srivatsan Varadharajan, si Ira Kemelmacher-Shlizerman au aplicat pozitia conditionata StyleGAN pentru incercarea virtuala. Cu toate acestea modelul lor nu poate pastra detaliile imbracamintei si este lent in timpul interfetei. Designul retelei de deformare a articolelor de imbracaminte folosit in aplicatia noastra este inspirat de la StyleGAN in manipularea imaginii, in special super performanta in deformarea formei.[TKA19]

In loc sa se foloseasca modularea stilului pentru a genera imbracaminte deformata, se foloseste modularea stilului pentru a prezice fluxul de aspect implicit care este apoi folosit pentru a deforma imbracamintea prin esantionare. Acest design este mult mai potrivit pentru pastrarea detaliilor articolelor de imbracaminte comparativ cu cel mentionat mai sus.

Fluxul de aspect. In contexul VTON fluxul de aspect a fost introdus pentru prima data de Xintong Han, Xiaojun Hu, Weilin Huang, si Matthew R Scott. De atunci, a castigat mai multa atentie si a fost adoptat de recentul VTON de ultima generatie de modele.

In principiu fluxul de aspect este folosit ca o grila de esantionare pentru deformarea articolelor de imbracaminte, este deci informatie fara pierderi si conservare superioara in detaliu. Dincolo de VTON, fluxul de aspect este popular si in alte sarcini. Este aplicat pentru sinteza vederii noi si ideea de flux de aspect pentru a deforma harta caracteristicilor pentru pozitia persoanei de transfer.[TKA19]

Diferit de toate acest model de estimare al fluxului de aspect, prin modelarea stilului, aplica un vector de stil global pentru a estima fluxul de aspect. Asadar metoda utilizata in aplicatia noastra este superioara in capacitatea sa de a face fata unor dezechilibre mari.

Avand in vedere imaginea unei persoane $p \in R^{3\times H\times W}$ si imaginea imbracamintei dintr-un magazin $g \in R^{3\times H\times W}$, scopul incercarii virtuale este de a genera o imagine de incercare $t \in R^{3\times H\times W}$ unde imbracamintea din g se potriveste partilor corespunzatoare din p. In plus in t generat, ambele detalii din g si regiunile nonvestimentare din p ar trebui pastrate.

Cu alte cuvinte, la fel persoana din p ar trebui sa apara neschimbata in t, cu exceptia ca acum poarta g. Pentru a elimina efectul negativ al analizei inexacte al omului, modelul folosit(F in Fig 2.1) este proiectat sa fie un model fara analizator.

Urmand strategia adoptata de modelele exsitente fara parser, mai intai antrenam un model bazat pe analizator F^{PB}. Apoi este folosit ca profesor pentru distilare de cunostinte pentru a ajuta la formarea modelului final F fara analizator.

Atat (F) cat si (F^{PB}) este format din 3 parti, adica 2 feature extractors (ϵ_p^{PB}, ϵ_g^{PB} in F^{PB} si ϵ_p , ϵ_g in F), warping module (W^{PB} in F^{PB} si W in F), si un generator (G^{PB} in F^{PB} si G in F). Fiecare dintre ele va fi detaliat in sectiunile urmatoare.

Conform standardului in modele existente fara parser, modelul bazat pe analizator F^{PB} este mai intai antrenat. Se folosesc 2 modalitati in antranarea ulterioara a modelului fara analizator propus F:

(a) pentru a genera imaginea persoanei (p) pentru a fi utilizata de F ca intrare si

(b) pentru a supraveghea antrenarea lui F prin cunostinte de distilare.

Concret (F^{PB}) ia ca intrari reprezentarea semantica (harta de segmentare, poza punct cheie si pozitie densa) a imaginii unei persoane reale ($p_{gt} \in R^{3\times H\times W}$) in setul de antrenament si o haina nepereche ($g_{un} \in R^{3\times H\times W}$). Iesirea din ($F^{PB}$) este imaginea p unde persoana originala poarta gun. p va servi drep intrare pentru F in timpul antrenamentului. Acest design, conform [YGL21], beneficiaza de faptul ca acum avem pereche imaginea persoanei (p_{gt}) si imaginea imbracamintei (g) in p_{gt} la antrenarea modelului F fara analizator adica:

$$F^* = argmin \parallel t - p_{gt} \parallel, F$$

unde $t = F(p, g)$ este imaginea de incercare generata de la F. Sa nu uitam ca F^{PB} este folosit doar in timpul antrenamentului lui F.

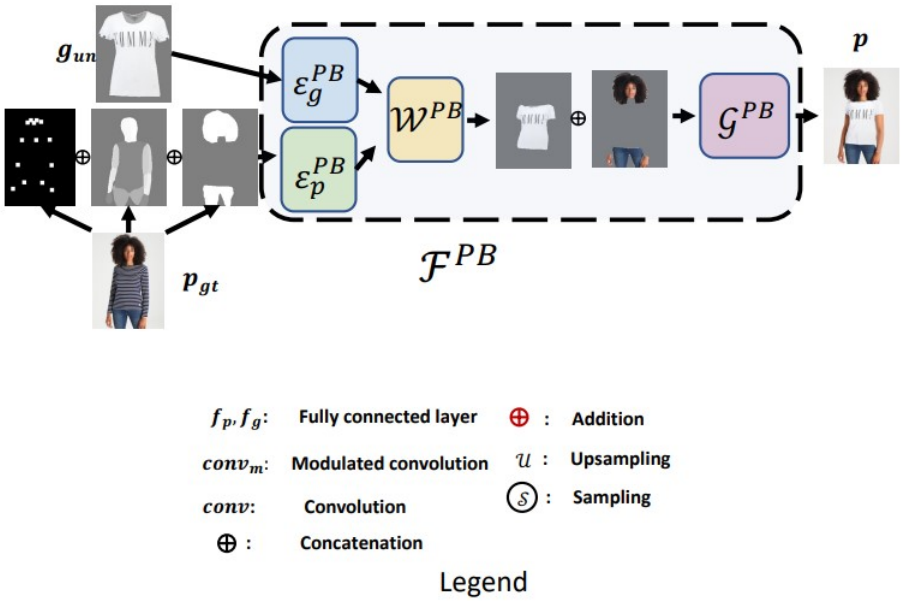

Figura 2.2: Modelul folosit bazat pe analizator [PIE17]

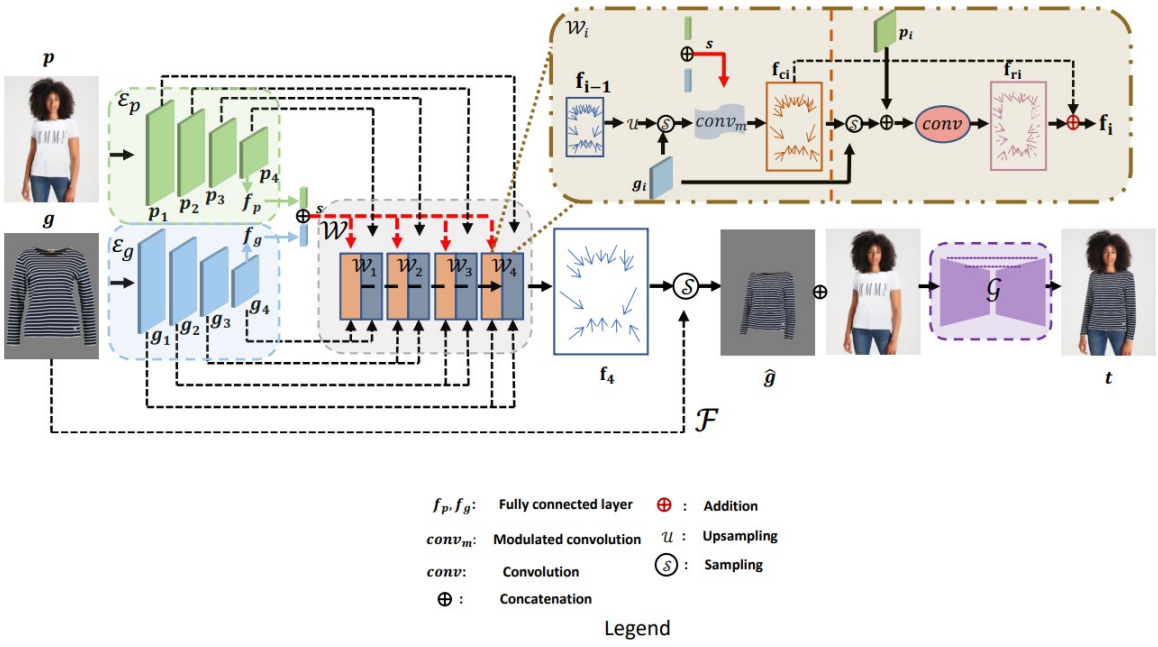

Figura 2.3: Modelul folosit overall [PIE17]

(Figura 2.2 si 2.3) O schema a sistemului folosit. Modelul (F^{PB}) bazat pe analizatorul pre-antrenat genereaza o imagine de iesire ca intrare a modelului F fara analizator.

Cele doua extractoare de caracteristici din F, extrag caracteristica imaginii peroanei si respectiv imaginea imbracamintei. Se extrage un vector de stil din harta caracteristicilor cu cea mai mica rezolutie din imaginea persoanei si imaginea imbracamintei. Modulul warping preia vectorul de stil si harta de caracteristici din imaginea persoanei si imaginea imbracamintei si scoate un plan al fluxului de aspect. Fluxul de aspect este apoi folosit pentru a deforma haina. In cele din urma, imbracamintea deformata este concatenata cu imaginea persoanei si introdusa in generator pentru a genera imaginea de incercare tinta. De retinut ca (F^{PB}) este utilizat numai in timpul antrenamentului.

2.3 Extragerea caracteristicilor

Aplicam doi codificatori convolutionali ϵ_p si ϵ_g pentru a extrage caracteristicile lui p si g. Atat ϵ_p cat si ϵ_g impartasesc aceeasi arhitectura compusa din blocuri reziduale stivuite. Caracteristicile extrase din ϵ_p si ϵ_g pot fi reprezentate ca: $\{p_i\}_1^N$ si $\{g_i\}_1^N$ (N=4 in Figura 2.3), unde $p \in R^{c_i \times h_i \times w_i}$ si $g \in R^{c_i \times h_i \times w_i}$ sunt hartile caracteristicilor extrase din blocul rezidual din ϵ_p respectiv ϵ_g.

Hartile de caracteristici extrase vor fi utilizate in W pentru a prezice fluxul de aspect.

2.4 Estimarea fluxului de aspect bazat pe stil

Principala componentă nouă a modelului propus este modul de estimare a fluxului de aspect global bazat pe stil.

Diferite de metodele anterioare care estimează fluxul de aspect pe baza corespondenței caracteristicilor locale, inițial propusă în estimarea fluxului optic, metoda folosita in aplicatia noastră se bazeaza pe un vector de stil global, mai întâi estimează un flux de aspect grosier prin modularea stilului și apoi rafinează fluxul de aspect grosier prezis pe baza corespondenței caracteristicilor locale.

După cum este ilustrat în Fig 2.3, modulul nostru de deformare (W) constă din N blocuri de deformare stivuite $\{W_i\}_1^N$, fiecare bloc este compus dintr-un strat de predicție a fluxului de aspect bazat pe stil (dreptunghi portocaliu) și un strat de

rafinare a fluxului de aspect bazat pe corespondență locală (dreptunghi albastru).

Concret, noi mai întâi extrageți un vector de stil global ($S \in R^c$) folosind caracteristicile de ieșire din blocurile N-lea (finale) ale ϵ_p și ϵ_g, notate ca p_N și g_N, ca:

$$s = [f_p(p_N), f_g(g_N)][\text{HSX22}],$$

unde f_p și f_g sunt straturi complet conectate și $[\cdot, \cdot]$ denotă concatenare.

Intrinsec, stilul global extras din vectorul s conţine informaţiile globale ale persoanei şi îmbrăcăminte, de exemplu, poziție, structură etc.

Similar cu stilul bazat manipularea imaginii, ne așteptăm ca vectorii de stil global să captureze deformația necesară pentru deformare g în p. Este astfel folosit pentru modularea stilului într-un generator de stil StyleGAN pentru estimarea unui câmp de flux de aspect.

Mai precis, în fluxul de aspect bazat pe stil stratul de predicție al fiecărui bloc W_i, aplicăm modularea stilului pentru a prezice un flux grosier:

$$f_{ci} = conv_m(S(g_{N+1-i}, U(f_{i-1})), s)[\text{HSX22}],$$

unde $conv_m$ denota convolutia modulata, $S(\cdot, \cdot)$ este operatorul de esantionare, U este operatorul de supraesantionare, si $f_{ci} \in R^{2 \times h_{i-1} \times w_{i-1}}$ este debitul prezis de la ultimul bloc de warping.

De retinut că primul bloc W_1 din W primește doar harta caracteristicilor articolelor de îmbrăcăminte cu cea mai mică rezoluție și vectorul stilului, de exemplu $f_{c1} = conv_m(g_N, s)$.

După cum se poate observa din ecuația de mai sus, f_{ci} prezis depinde de harta caracteristicilor articolelor de îmbrăcăminte și vectorul de stil global.Are astfel un câmp receptiv global și este capabil să facă față dezechilibrelor mari între îmbrăcăminte și imaginile persoanei.

Cu toate acestea, ca stil vectorul s este o reprezentare globală, ca compromis, are a capacitatea limitată de a estima cu precizie granulația fină locală a fluxului de aspect (după cum se arată în Fig. 2.6).

Pentru fluxul grosier este deci nevoie de un rafinament local.

Pentru a rafina f_{ci}, introducem o corespondență locală bazată pe stratul de rafinare a fluxului de aspect în fiecare bloc W_i . El vizează pentru a estima un flux de aspect local cu granulație fină:

$$f_{ri} = conv_m([S(g_{N+1-i}, f_{ci}), p_{N+1-i}])[\text{HSX22}],$$

unde f_{ri} este fluxul de rafinare prezis și $conv$ denotă convoluţie.

În principiu, stratul de rafinare estimează fluxul rafinamentului prin corespondența locală, adică corespondenţa dintre trăsăturile persoanei deformate şi caracteristică vestimentară în același câmp receptiv.

Rețineți că după deformarea prin f_{ci}, putem presupune că corespunzător regiunilor/caracteristicilor din g_{N+1_i} și p_{N+1_i} sunt acum localizate în același câmp receptiv.

La final, adăugăm fluxul grosier și cea locală cu granulație fină fluxului de aspect împreună ca rezultat al fiecărei deformări bloc:

$$f_i = f_{ci} + f_{ri},$$

Fluxul de aspect prezis f_N din ultimul bloc în W este folosit pentru a deforma îmbrăcămintea:

$$\hat{g} = S(g, f_N)$$

Iar haina deformată $\hat{g}$ este apoi concatenată cu imaginea persoanei și introdusă într-un generator pentru generarea de imagini de încercare țintă:

$$\hat{t} = G[\hat{g}, p]$$

Generatorul G are o arhitectură codificator-decodor care omite conexiunile între ele.

2.5 Obiective de învățare

Pentru a ne antrena modelul, aplicăm mai întâi un perceptual loss între ieșirea lui F și imaginea persoanei adevărul de la bază p_{gt}:

$$L_p = \sum_i ||\phi_i(t) - \phi_i(p_{gt})||[\text{HSX22}],$$

unde ϕ_i este al i-lea bloc al retelei VGG pre-antrenate [SZ20].

Pentru a supraveghea antrenamentul modelului de deformare W , am aplicat un loss pe îmbrăcămintea deformată:

$$L_g = ||\hat{g} - m_g p_g t||[\text{HSX22}],$$

unde m_g este masca de îmbrăcăminte a $p_g t$ prezisă de un model de analiză umană

disponibilă.

Conform standardului din metodele anterioare de flux de aspect, aplicăm și o regularizare a netezirii asupra debitul prezis din fiecare bloc în W:

$$L_r = \sum_i ||\nabla f_i|| [\text{HSX22}],$$

unde $||\nabla f_i||$ este funcția generalizată de loss Charbonnier

Ca intrări (hartă de segmentare, poziție punct cheie și poziție densă) la codificatorul de persoană bazat pe analizator (ϵ^{PB}) conțin mai multe informații semantice decât cele ale modelului F parserfree (imaginea persoanei), aplicăm un loss prin distilare pentru a ghida învățarea codificatorului de persoană ϵ^P în F:

$$L_D = \sum_i ||p_i^{PB} - p_i|| [\text{HSX22}],$$

unde p_i^{PB} este harta caracteristicilor de ieșire de la al i-lea bloc din codificator persoanei ϵ_p^{PB} în modelul bazat pe analizator pre-antrenat F_{PB}.

Obiectivul general de învățare este:

$$L = \lambda_p L_p +, \lambda_g L_g + \lambda_R L_R + \lambda_D L_D [\text{HSX22}],$$

unde λ_p, λ_g, λ_R si λ_D denota hiperparametrii pentru echilibrarea celor patru obiective.

2.6 Experimente si rezultate

Experimentăm modelul nostru pe setul de date VITON.Este cel mai popular set de date folosit în celelalte modele anterioare.

VITON conține un set de antrenament care conține 14.221 de perechi de imagini și un set de date de testare de 2.032 de perechi. Atât imaginile persoanei, cât și cele ale hainelor sunt de rezoluție 256 × 192.

De asemenea, creăm un set de date de testare, notat cu augmentat VITON, pentru a evalua robustețea modelului față de imaginea persoanei poziționate aleatoriu cu dezechilibre mai mari cu imaginile de îmbrăcăminte din setul de date original. (exemplu Fig 2.5)

Deoarece majoritatea imaginilor persoanei de testare în VITON sunt bine poziționate,

astfel încât imaginea persoanei și îmbrăcămintea să fie bine pre-aliniat (de exemplu, majoritatea regiunilor corespunzătoare din persoană imaginea și imaginea îmbrăcămintei sunt situate aproximativ în același câmp receptiv), nu este potrivit pentru această evaluare.

Concret, setul de date VITON augmentat este creat prin creșterea aleatorie a imaginii persoanei de testare în VITON si prin deplasarea și mărirea/micșorarea imaginii. În special, noi marim la intamplare cu 1/3 imaginile persoanei de testare în VITON prin schimbarea poziției persoanei în imagine și creștem aleatoriu cu încă 1/3 imaginia de testare în VITON prin mărirea/micșorarea persoanei din imagine și păstreaza cu încă 1/3 imaginea de testare neschimbata.

Modelul nostru este implementat în PyTorch. Ne antrenăm modelul cu un Nvidia GTX GPU 1660-Ti. Setăm batch size-ul la 4 și antrenăm modelul cu 100 de epoci. Antrenăm modelul cu optimizatorul Adam [KB15]. Rata de învățare inițială este setată la 5e - 4. Fiecare bloc rezidual din ϵ_p și ϵ_g este urmat de un pooling layer pentru a reduce dimensiunea spatiala. Am stabilit N = 5 și c = 256 în implementare.[HSX22]

Ne evaluăm modelul atât automat cât și manual. În evaluarea automată, conform standardului în VTON, evaluăm performanța modelului folosind similaritatea structurii (SSIM) și Frechet Inception Distance (FID). Conform Parser-free virtual try-on via distilling appearance flows. In CVPR, 2021, scorul inițial (IS) nu este adecvat pentru evaluarea imaginilor VTON, deci nu o adoptăm în evaluare.

În evaluarea manuală, efectuăm un studiu vizual pentru a compara calitatea imaginilor de încercare generate de la diferite modele.

Ne comparăm metodele cu alte metode bazate pe parser VTON, CP-VTON, Cloth-flow, CPVTON++, ACGPN, DCTON și ZFlow. De asemenea, comparăm cu metoda SOTA fără analizator PFAFN.[HSX22]

Metode	Warping	Parser	SSIM	FID
VTON	TPS	Y	0.74	55.71
PF-AFN	AF	N	0.89	10.09
Cloth-flow	AF	Y	0.84	14.43
CP-VTON++	TPS	Y	0.75	21.04
ACGPN	TPS	Y	0.84	16.64
Ours	AF	N	0.91	8.89

Tabela 2.1: Rezultatele diferitelor modele VTON in comparatie cu modelul folosit [HSX22]

Rezultatele cantitative ale testării VITON pe setul de date este prezentat în tabelul 2.1. Acest model atinge performante de ultima generatie in comparatie cu

celelalte modele din literatura.

Modelele cu deformarea pe bază de flux de aspect au în general rezultate mai bune decât deformarea bazată pe TPS metode.Deși necesită mai mult timp de antrenament, metodele fără parser sunt mult mai bune decât metodele bazate pe parser.

Rezultatele evaluarii umane sunt prezentate în tabelul .

Modelul nostru depășește toate modelele comparate cu o rată de preferință de peste 10

Rezultatele calitative de la diferite modele sunt ilustrate în Fig 2.4

Per total, metoda noastră generează imagini de încercare mai bune. De exemplu, poza dură și ocluzia în al doilea și al treilea rând.

Exemplele calitative sunt ilustrate în Fig 2.4. Modelul nostru poate genera consecvență (de exemplu, mâneca stângă a articolelor de îmbrăcăminte) și imagini de înaltă calitate de probă pentru marile dezalinieri.

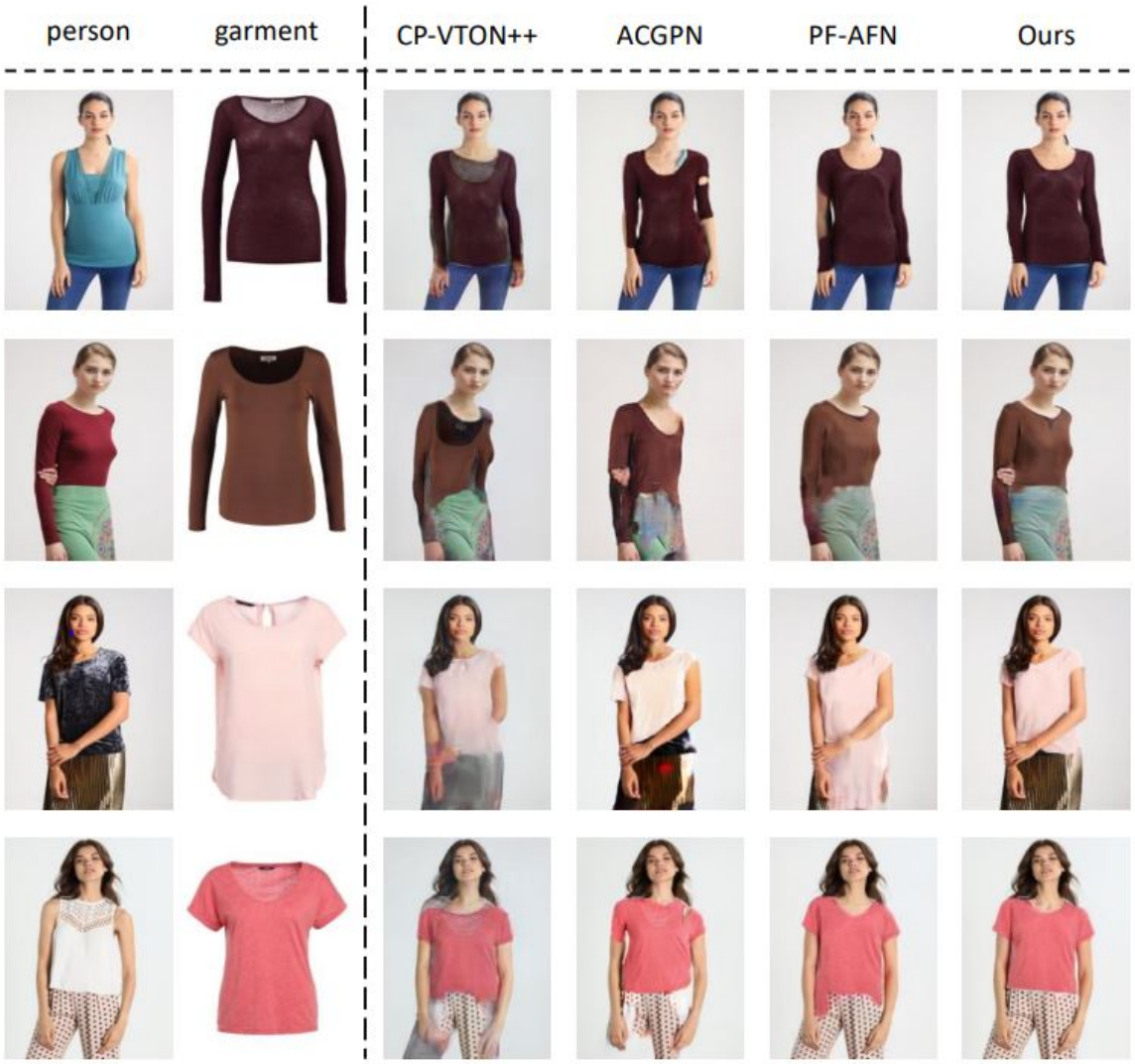

Figura 2.4: Rezultate calitative de la diferite modele (CP-VTON++, ACGPN, PF-AFN și al nostru) pe setul de date de testare VITON.[HSX22]

Comparatie metode	rata de preferinta
CP-VTON++	12.7% / 87.3%
ACGPN	20.2% / 79.8%
Cloth-flow	38.5% / 61.5%
AF-PFN	43.2% / 56.8%

Tabela 2.2: Rata de preferință comparând alte modele cu ale noastre model (alte modele/modelul nostru) în evaluarea umană. [HSX22]

Figura 2.5: Ilustrarea robusteței diferitelor modele VTON la imaginea persoanei poziționate aleatoriu. Primul rând folosește imaginea originală a persoanei ca intrare. Și al doilea rând folosește ca intrare imaginea persoanei deplasată vertical. ACGPN, Cloth-flow, PF-AFN. [HSX22]

Figura 2.6: Compararea rezultatelor cu doar f_{ci} folosit în W_i și $f_{ci} + f_{ri}$ folosit în W_i. [HSX22]

Capitolul 3

Tehnologii folosite

3.1 Backend

In dezvoltarea software, termenii frontend si backend se refera la separarea elementelor ce definesc interfata de prezentare a unei aplicatii (front end a.k.a ce vede userul final) si nivelul de acces si redare date (backend a.k.a actiunile intreprinse de server). Dezvoltarea backend functioneaza in tandem cu partea frontala pentru a oferi utilizatorului produsul final.[Aca]

Pe scurt, in backend sunt gestionate functionalitatile de culise a aplicatiilor web. Codul specific componentei de backend gestioneaza practic:

- legatura dintre interfata si baza de date
- conexiunea utilizatorilor
- alimentarea aplicatiei web

3.1.1 Python,MongoDB,Flask

Python este un limbaj de programare pentru calculator folosit adesea pentru a construi site-uri web și software, pentru a automatiza sarcini și pentru a efectua analize de date. Python este un limbaj de uz general, ceea ce înseamnă că poate fi folosit pentru a crea o varietate de programe diferite și nu este specializat pentru probleme specifice.[Cou]

Flask este un framework web, este un modul Python care vă permite să dezvoltați cu ușurință aplicații web. Are un nucleu mic și ușor de extins: este un microcadru care nu include un ORM (Object Relational Manager) sau astfel de caracteristici. Are multe caracteristici interesante, cum ar fi rutarea URL, motorul de șablon. Este un cadru de aplicație web WSGI.

Interfața Web Server Gateway (Web Server Gateway Interface, WSGI) a fost folosită ca standard pentru dezvoltarea aplicațiilor web Python. WSGI este specificația unei interfețe comune între serverele web și aplicațiile web.[pyt]

De ce Flask este o alegere bună pentru un web framework?
- Există un server de dezvoltare încorporat și un depanator rapid
- Usor de folosit
- Cookie-urile securizate sunt acceptate
- Template folosind Jinja2
- Expediarea cererilor folosind REST
- Suportul pentru testarea unitară este încorporat

Flask REST API - Serviciile REST API vă permit să interacționați cu baza de date prin simpla efectuare a solicitărilor HTTP.

Acesta este adesea modul în care este creat backend-ul aplicațiilor web. Datele returnate sunt în format JSON, iar solicitările pe care le folosim sunt PUT, DELETE, POST și GET

- GET - Cea mai comună metodă. Se trimite un mesaj GET, iar serverul returnează date
- POST - Folosit pentru a trimite date de formular HTML către server. Datele primite prin metoda POST nu sunt stocate în cache de către server.
- PUT - inlocuieste datele curente ale resursei țintă cu conținut încărcat
- DELETE - sterge toate detele curente ale resursei țintă date de adresa URL

Flask mapează cererile HTTP la funcțiile Python. În acest caz, am mapat o cale URL (‘/’) la o funcție, home. Când ne conectăm la serverul Flask la http://127.0.0.1:5000/, Flask verifică dacă există o potrivire între calea furnizată și o funcție definită. Deoarece /, sau nicio cale suplimentară furnizată, nu a fost mapat la funcția de acasă, Flask rulează codul în funcție și afișează rezultatul returnat în browser. În acest caz, rezultatul returnat este un marcaj HTML pentru o pagină de pornire care întâmpină vizitatorii site-ului care găzduiește API-ul nostru.

MongoDB este un program de gestionare a bazelor de date NoSQL open source. NoSQL este folosit ca alternativă la bazele de date relaționale tradiționale.

Bazele de date NoSQL sunt destul de utile pentru lucrul cu seturi mari de date distribuite. MongoDB este un instrument care poate gestiona informații orientate către documente, poate stoca sau prelua informații.

Simplifică integrarea datelor și oferă o scalabilitate mai bună decât bazele de date relaționale tradiționale.

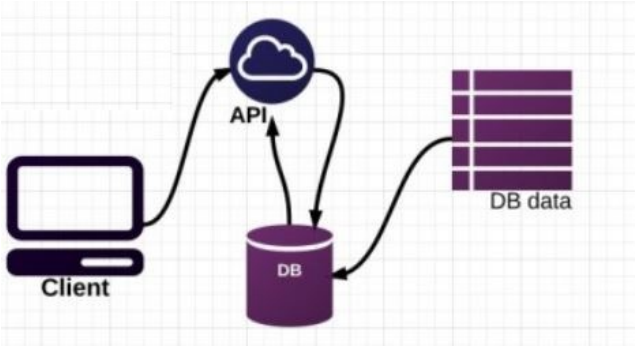

Figura 3.1: Api interactiune cu client si baza de date [ARY]

Site-urile de comerț electronic de astăzi au nevoie de modele bogate de date și interogări dinamice. MongoDB oferă acest lucru, făcându-l o alegere populară pentru multe companii. Există o gamă largă de funcții pe care le puteți construi pentru magazinul dvs. folosind MongoDB.

Diferentele dintre MongoDB si o baza de date traditionala: Majoritatea bazelor de date sunt cunoscute ca baze de date relationale RDBMS (Relational Data Base Management Systems) si sunt structurate in tablele si folosesc in general un limbaj SQL(Structured Query Language) pentru a face modificari si pentru a seta anumite reguli datelor.

Asadar MongoBD stocheaza datele fara o structura de tabel, le stocheaza intr-un format JSON. Asta ne permite ca programatori sa facem modificari frecvente structurii datelor noastre cu flexibilitate ridicata.

Figura 3.2: Structura MongoDB [Tim]

Asadar MongoDb are baze de date, o baza de date este alcatuita din colectii si o colectie este alcatuita din documente. Documentele stocheaza toate datele noastre

iar aceste date sunt categorizate in colectii iar colectiile formeaza o singura baza de date. Exemplu Figura 3.2

Datele dintr-un document sunt stocate intr-un camp de valori pereche exact ca un dictionar in Python.

Deci pe plan intern MongoDB foloseste o ceva numit BSON (Binary javaScript Object Notation) similar cu JSON dar are cateva diferente minore. [Tim]

3.2 FrontEnd

Front-End-ul este partea aplicației care interacționează în mod direct cu utilizatorul. Cunoscută și sub numele de partea client, aceasta se ocupă de aspectul și funcționalitatea interfeței paginii web.

Elementele implementate de către un Front-End developer aplicației includ, dar nu sunt limitate la: stilul, paleta de culori, imaginile, butoanele, meniurile de navigație, comportamentul elementelor vizibile de către utilizator.

3.2.1 React,Typescript,MaterialUI

React este o bibliotecă JavaScript declarativă, eficientă și flexibilă pentru construirea de interfețe cu utilizatorul. Vă permite să compuneți interfețe de utilizare complexe din bucăți mici și izolate de cod numite „componente".

TypeScript este un limbaj de programare conceput pentru a funcționa cu JavaScript. Deși nu este conceput special pentru React, poate fi folosit cu React. TypeScript oferă unele beneficii față de JavaScript, cum ar fi siguranța îmbunătățită a tipului și suport mai bun pentru programarea orientată pe obiecte.

Material-UI este pur și simplu o bibliotecă care ne permite să importăm și să folosim diferite componente pentru a crea o interfață cu utilizatorul în aplicațiile noastre React. Acest lucru economisește o cantitate semnificativă de timp, deoarece dezvoltatorii nu trebuie să scrie totul de la zero.

Redux este o bibliotecă JavaScript open-source pentru gestionarea stării aplicației. Funcționează cel mai bine în aplicații extinse, extinse. Cu toate acestea, pentru a-l utiliza corect, mai întâi trebuie să vă pregătiți. Totul ar trebui să meargă fără probleme atâta timp cât middleware-ul este complet și aveți control deplin asupra preluărilor, stărilor etc. În Redux, nu trebuie să preluați totul tot timpul. Acesta este motivul pentru care Redux rămâne cel mai popular instrument bazat pe flux pentru managementul statului. [Bat]

Din punct de vedere al arhitecturii, Redux ajută la menținerea ordinii în folderele și fișierele proiectului. Ajută programatorii să înțeleagă structura aplicației și să introducă oameni noi în proiect (cu condiția ca aceștia să aibă cunoștințe anterioare despre Redux). Componentele sale sunt: obiect JS global, funcții de reducere, acțiuni și abonamente.

Avantajele utilizării Redux:

- Este larg răspândit, așa că există o comunitate activă care te poate ajuta.
- Usor de testat
- Depanare bună, permițându-vă să înregistrați stări și acțiuni.
- Structură de cod bună – Aplicațiile Redux au de obicei o arhitectură similară, astfel încât programatorii experimentați pot trece cu ușurință la alt proiect.
- Este conceput pentru a fi utilizat cu date actualizate frecvent.
- Redux elimină re-rendările inutile, iar vizualizarea se reîmprospătează atunci când se schimbă un anumit obiect din magazin.

Capitolul 4

Dezvoltarea aplicației

4.1 Dezvoltarea aplicației client

Folosind React pentru o flexibilitate mai bună a clientului, structura fișierelor si a folderelor a fost creată automat folosind cele mai bune practici. Pentru a creea un proiect nou in react folosind typescript am folosit următoarea comandă: npx create-react-app my-app –template typescript

Se poate observa in Figura 4.1 structura folderelor și a fișierelor:

• node_modules folderul cu toate modulele instalate de NPM (ex: redux, material ui, axios)

• package.json conține atât date despre aplicație (ex: nume, versiune) cât și numele și versiunea pentru toate modulele și bibliotecile necesare ca aplicația să ruleze.

• App.tsx aceasta este principala noastră componentă React. Extensia .tsx ne spune că fișierul folosește atât TypeScript, cât și JSX.

• src/components - toate componentele utilizate in dezvoltarea aplicației

• src/components/app - store.tsx permite componentelor să partajeze starea

Celelalte dependențe din proiect cum ar fi axios, material-ui, react-router-dom, react-redux le-am instalat folosind comanda npm install 'package-name'

Unul dintre lucrurile care m-au convins să folosesc React au fost componentele funcționale. „Modul vechi" de a face componente este cu componente de clasă. Și pot păstra statul pe clasă. Statul este ca recuzita, dar privat și controlat doar de componentă.

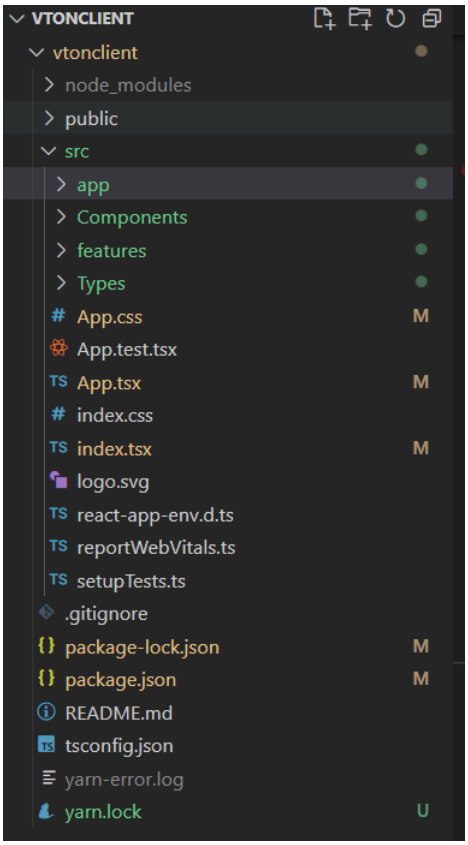

Figura 4.1: Structura fișierelor

4.1.1 Interfața autentificare si înregistrare

În Figura 4.2 putem observa interfața grafică pentru funcționalitatea de autentificare. Aici utilizatorul se va putea autentifica cu ajutorul unui nume de utilizator și parola cu care și-a creat contul. În cazul în care utilizatorul este deja conectat se va deschide pagina principala. Tot în această pagina putem să navigăm la pagina de register pentru a ne creea un cont de utilizator în cazul in care nu avem unul.

La apăsarea butonului LOGIN sunt trimise datele formularului (username si parolă) pintr-o cerere HTTP de tip POST la server, folosind funcția axios.post din libraria axios. În funcție de răspunsul primit de la server care este de tip JSON, ori afisăm mesajul de eroare dacă autentificarea nu a fost realizată cu succes.

Înregistrarea utilizatorului se face prin accesarea formularului de înregistrare, unde datele sunt transmise către server aproape la fel ca la autentificare, iar după validarea lor aceleași acțiuni sunt luate în funcție de răspunsul primit.

După cum putem observa in Fig 4.4 cu ajutorul axios facem o cerere de tip POST la server cu datele de autentificare, date create cu ajutorul interfeței RegisterProps, și primește ca răspuns datele cu care a fost creat contul.

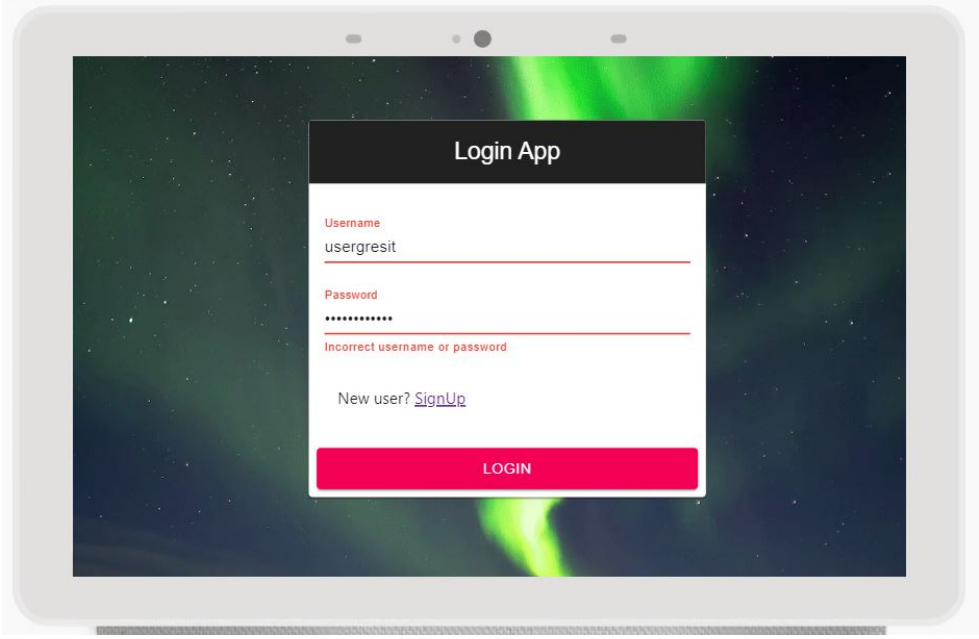

Figura 4.2: Interfața de autentificare

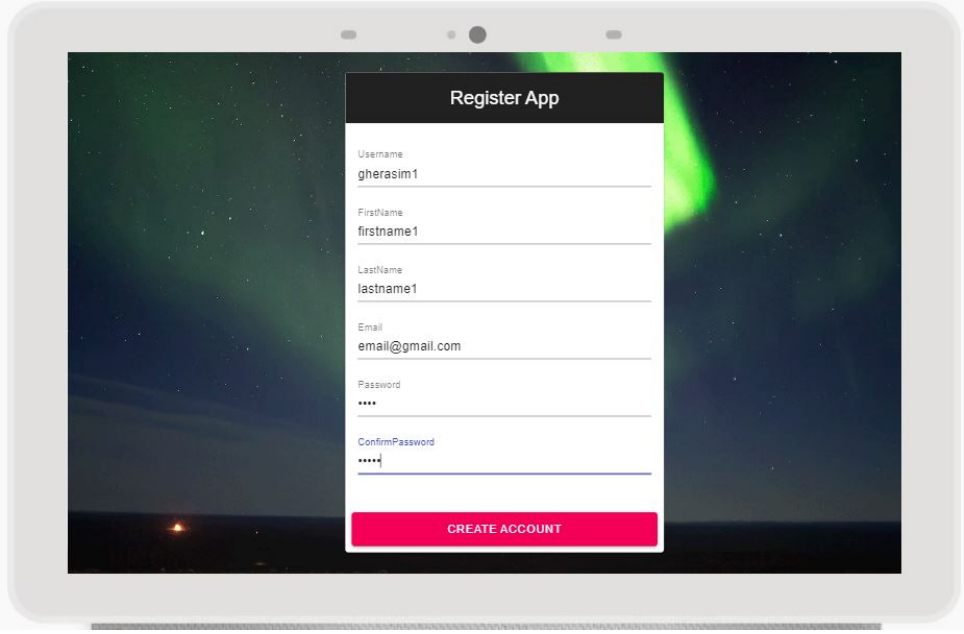

Figura 4.3: Interfața de inregistrare

```
export const userRegister = createAsyncThunk("user_current/userRegister", async (dataRequest:RegisterProps) => {
    const response = await axios.post<IServerToDoDataRegister>(
      "http://localhost:5000/register", dataRequest
    );

      const user = {
        username: response.data.username,
        firstname: response.data.firstname,
         lastname: response.data.lastname,
        email:response.data.email}
  return { user_current: user };
});
```

Figura 4.4: Secțiune de cod: Cererea HTTP de tip POST pentru înregistrare

4.1.2 Interfața home page si vizualizare produse

După ce utilizatorul se autentifică cu succes va fi redirectionat către pagina principală a site-ului HomePage. Această pagina conține o bară de navigare cu ajutorul careia putem sa vizualizăm produsele existente pe site la momentul actual.

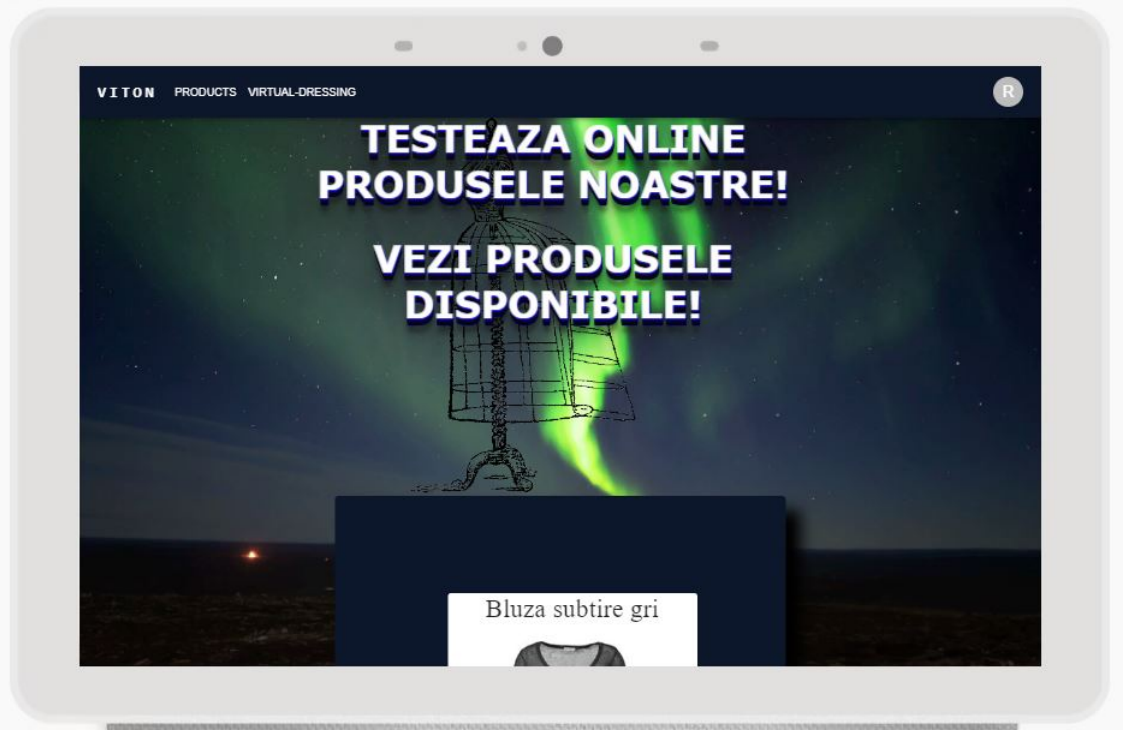

Figura 4.5: Interfața home page

Pentru a vizualiza lista de produse disponibile în baza de date trebuie să selectăm din pagina principală opțiunea products din bara de navigație sau accesând ruta http://localhost/products. Putem selecta orice produs de pe site pentru al vizualiza mai detaliat dând click pe acesta. Bara de navigatie este o componenta ce poate fi refolosita cat timp utilizatorul este logat.

Pentru a vizualiza un produs mai în detaliu putem sa-l mai accesam folosind ruta http://localhost/products/id-produs, unde id-produs este id-ul produsului selectat. Acest id este de tip numeric pozitiv. Fiecare produs are în componență o denumire, descriere și un preț. Apăsând butonul „Încearcă proba virtuală" vom fi redirectionați către pagina de virtual-dressing. În figura 4.5 este ilustrată pagina de homepage, figura 4.6 lista tuturor produselor iar figura 4.7 vizualizarea unui produs selectat.

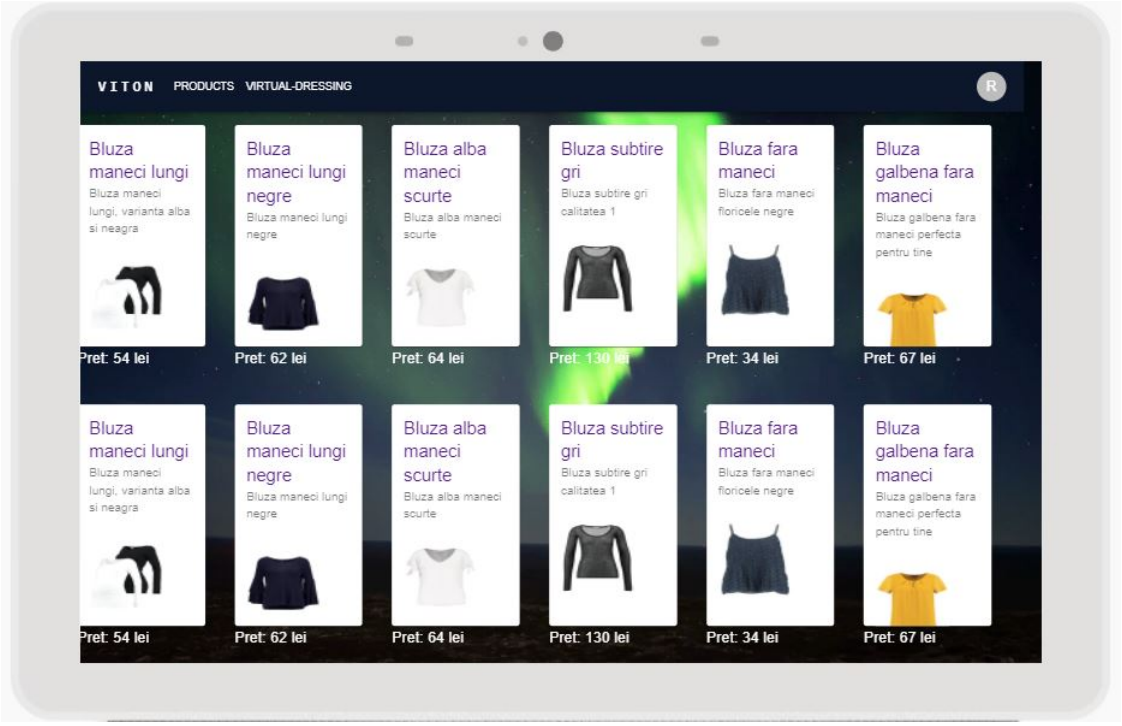

Figura 4.6: Interfața selectare produse

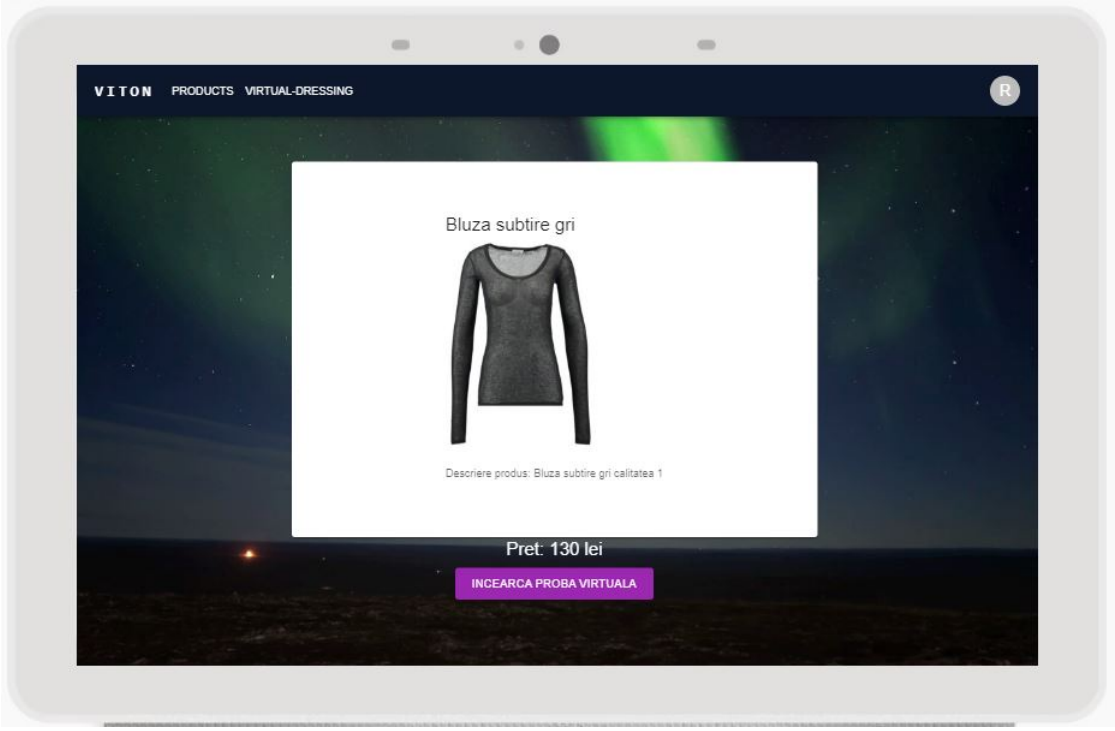

Figura 4.7: Interfața vizualizare produs selectat

4.1.3 Interfața virtual dressing

Odată selectat un produs din lista de produse utilizatorul poate să încarce o imagine cu corpul/silueta acestuia iar server-ul folosind algoritmi de modelare va genera o imagine de încercare cu poza aleasă de pe site si cea încarcată de el.

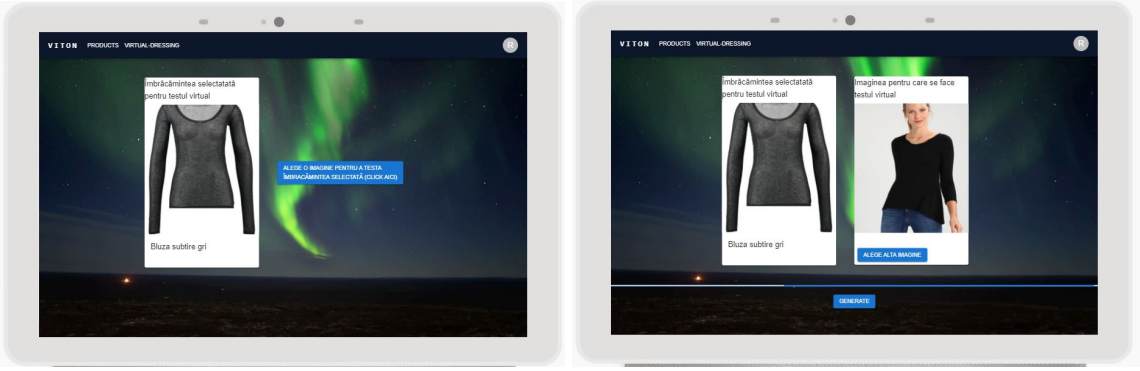

Figura 4.8: Interfata virtual dressing - inserare imagine

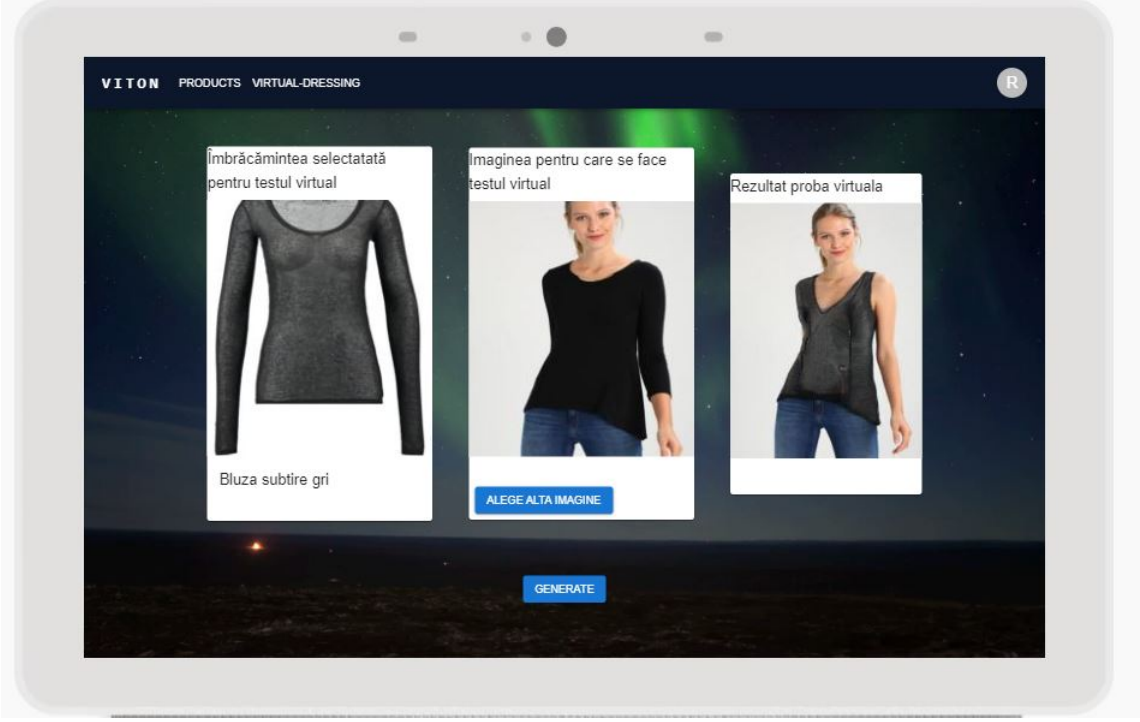

Figura 4.9: Interfata virtual dressing - rezultat

Imaginea selectată cu îmbrăcămintea din pagina de vizualizare produs este vizibilă și în pagina de virtual-dressing. Dacă utilizatorul consideră că poza încarcată nu este potrivită pentru proba virtuală poate selecta altă imagine apăsând butonul „Alege altă imagine” doar după ce a încărcat deja o imagine. După ce introduce imaginea dorită în aceeași pagină va apărea un buton „Generate”. Odată apăsat o

bară de așteptare va aparea deasupra acestui buton până cand serverul va returna imaginea generată cu proba virtuală.

În figura 4.8 stânga este ilustrată pagina exact după selectarea articolului vestimentar alături de un buton de selectare imagine din computerul personal. În aceeași figură dar în partea dreaptă este prezentată bara de asteptare după apasarea butonului generate. Figura 4.9 ne prezintă rezultatul mult așteptat.

4.2 Dezvoltarea aplicatiei server

Pentru dezvoltarea aplicației server am utilizat limbajul de programare Python. Înainte de a incepe implementarea propriu zisă am creat o variabilă de mediu pentru a stoca toate dependințele/pachetele instalate. Folderul server_app conține următoarea structură de fișiere/foldere:

- server_app
  - domain
    - Product.py
    - User.py
  - repository
    - Repository.py
  - service
    - Service.py
  - test
    - test.py
  - app.py

In folderul domain avem cele două entitați ale problemei Produs respectiv User. Un produs conține urmatoarele atribute: id, denumire, descriere, preț, url_image respectiv stoc pe când un obiect de tip user conține: username, firstname, lastname, email respectiv password. Repository-ul, Service-ul si api-ul au acces direct la clasele din domeniu problemei.

Api-ul este implementat în fisierul app.py. În acest fișier am creat o instanță de clasă Service care primește un obiect de tpi Repository ca parametru. Api-ul apelează funcțiile din repository prin intermediul acestui service definit în api. Clasa Repository face legatura dintre domeniul problemei si baza de date. Când clientul face un request către server, service-ul este apelat, iar acesta face apel mai departe la

repository. Repository-ul citește sau scrie datele in funcție de request într-o bază de date MongoDB.

În folderul test se află fisierul test.py care conține implementarea modelului de VTON (Virtual Try-On). Acest model conține un AFWN(Architecture of adaptive feature weighting model) si un ResUnetGenerator. Modulul de deformare preia vectorul de stil și hărțile caracteristicilor din imaginea persoanei și imaginea îmbrăcămintei și scoate o hartă a fluxului de aspect. Fluxul de aspect este apoi folosit pentru a deforma îmbrăcămintea. În cele din urmă, îmbrăcămintea deformată este concatenată cu imaginea persoanei și introdusă în generator pentru a genera imaginea de încercare țintă.

4.2.1 Cererile HTTP

Cererea de autentificare /login

La autentificarea clientului prin introducerea numelui de utilizator și a parolei în formular, pe partea de client se face o cerere la server pentru validarea celor două. În cazul în care username-ul si parola deja există în baza de date se va returna un json ce conține username-ul și un successLogin cu valoarea true altfel false în cazul în care username-ul si parola nu există in baza de date. Aceasta cerere este de tip get si post. Cererea de tip post necesită un corp(body) în care definim datele entitatii care urmează sa fie verificate/create. Pentru a obține datele dintr-un request, flask ne pune la dispoziție obiectul request cu ajutorul caruia putem obține datele trimise de la client. De exemplu în aplicatia noastra, pentru a obtine atributul username folosim request.json['username'].

Cererea de înregistrare /register

Dacă clientul nu are cont trebuie să se înregistreze, completând numai un formular ce conține numele de utilizator, numele de familie, prenumele, adresa de email și parola. Aceste date sunt apoi trimise server-ului pentru validare. Înregistrarea clientului este foarte asemănătoare cu autentificarea clientului. Diferența este că avem nevoie de mai multe informații despre utilizator. Cu ajutorul datelor primite de la client creăm un obiect de tip User. Odată creat obiectul, îl trimitem la service, cu ajutorul funcției createAccount care primeste ca parametru un obiect de tip User iar mai apoi trimite aceste informații mai departe către repository. Repository-ul preia obiectul User si îl salvează in baza de date.

Cererea de produse /products

Această cerere este de tipul get. Principalul ei scop este de obține o listă cu toate produsele din baza de date. Funcția get_all_products returnează o listă de obiecte de tipul Product. Odată ce avem lista de produse trebuie să ne asigurăm că o trimitem în format JSON către client. Pentru aceasta folosim funcția dumps din libraria json.

Cererea de produs după un id /getproductbyid

Dacă dorim să vizualizăm un anumit produs de pe site trebuie să obținem datele doar pentru acel produs, nu întreaga listă, așadar cu ajutorul unui id de produs trimis ca request facem o căutare în baza de date. Odată gasit, produsul este trimis către client.

Cererea de obținere a unei imagini in funcție de id-ul acestuia /imgs/<path:path>

Toate imaginiile sunt stocate pe server, așadar pentru a obține o imagine stocăm doar id-ul acesteia. În cazul nostru path este un id pentru imaginea noastră primit prin url. Imaginea este cautată într-un path specific de pe server. După ce obținem path-ul complet de pe server trimitem imaginea cu ajutorul funcției send_file(path, mimetype), unde path este path-ul imaginii si mimetype este tipul de ex: 'image/jpg'

Cererea de virtual dressing /virtual-dressing/<path:path>

Pentru utiliza funcționalitatea de virtual dressing trebuie să alegem o imagine din computerul nostru cu silueta noastra, odată aleasă această imagine trebuie să o trimitem catre server pentru a o procesa. Accesul din server la imagine ne este oferit cu ajutorul comenzii request.files['image']. În url avem datele despre îmbrăcămintea aleasă. Așadar avand datele necesare pentru a aplica algoritmul descris în capitolul de Machine Learning, ni se returnează o imagine de test cu îmbracamintea pe silueta noastră. Citim imaginea procesată si cu ajutorul unui algoritm de conversie, convertim imaginea in base64 pentru a putea fi mai usor de preluat de către client in react.

4.2.2 Baza de date

Pentru stocarea datelor legate de utilizatorii aplicației am folosit MongoDB. Pentru a folosi MongoDB trebuie să instalăm mai întai dependențele necesare cu ajutorul comenzii 'pip install pymongo'. Conectarea la baza de date se face cu ajutorul unui

connectionString definit în clasa Repository.

Clasa repository interacționează direct cu baza de date, toate funcțiile necesare find implementate acolo. Pentru crearea unui colecții am folosit aplicația MongoDB Compass. În cazul aplicației noastre avem doua colecții: produs si client iar aceste colecții conțin multiple documente care stochează datele legate de colecție. Asadar în colecția produs avem multiple documente ce stochează id-ul, denumirea, descrierea, pretul, url imaginii și stocul unor produse.

Atunci când un obiect de tip Repository este creat funcția loadDataBase() este apelată, se preia dintr-un fișier .env parola pentru connectionString si se creează o instantă client cu ajutorul clasei MongoClient care primește ca parametru connectionString-ul generat pentru aplicația noastră. Colecțiile fiind create din interfața grafică nu ne mai ramane decât să creăm documentele necesare aplicației.

```
class Repository:
    def __init__(self):
        print("load repo")
        self.client_mongo = self.loadDatabase()
        self.user_collection = self.getUserCollection()
        self.product_collection = self.getProductCollection()

    def loadDatabase(self):
        load_dotenv(find_dotenv())
        password = os.environ.get("MONGODB_PWD")
        connection_string = f"mongodb+srv://GherasimGeorgian:"+ quote(password) +"@cluster0.gckzd.mongodb.net/?retryWrites=true&w=majority"
        return  MongoClient(connection_string)

    def getUserCollection(self):
        users_db = self.client_mongo.users
        return users_db.users

    def create_client(self, new_user):
        user_document = {
            "username":new_user.get_UserName(),
            "firstName":new_user.get_FirstName(),
            "lastName":new_user.get_LastName(),
            "email":new_user.get_email(),
            "password":new_user.get_Password(),
        }
        inserted_id = self.getUserCollection().insert_one(user_document).inserted_id
        print(inserted_id)
```

Figura 4.10: Secțiune de cod: Adaugarea unui client in MongoDB

Pentru adăugarea unui nou client în baza de date, se apelează funcția create_client (self,new_user), new_user fiind un obiect de tip User, trebuie să cream un document nou (user_document) cu datele clientului, iar apoi obținem o referință catre colecție și respectiv apelul funcției insert_one care primește ca parametru documentul ce dorim sa-l adăugam în baza de date.

În figura 4.10 de mai sus puteți observa cum se face o adăugare a unui client in baza de date, respectiv crearea unui client MongoDB și un getter la coleția user. Pentru obținerea tuturor documentelor dintr-o colecție se foloseste funcția find().

Dacă dorim să obținem/căutam un document de tip produs din colecția produs ce are id-ul un parametru „our_id" vom folosi funcția find_one("id:our_id").

Capitolul 5

Concluzii

Scopul acestei lucrări a fost acela de a satisface unele curiozități și identificarea provocărilor întâlnite în dezvoltarea unei aplicatii în care sunt aplicate tehnicile de inteligență artificială, comunicarea în timp real și de a găsi soluții optime în dezvoltarea lor.

Bibliografie

[Aca] LEC Academy. Programare web: Ce inseamna backend si ce limbaje de programare foloseste? url:http://www.lec-academy.ro/programare-backend-limbaj-java/ (accessed: 09.06.2022).

[ADB] Eddy Ilg Philip Hausser Caner Hazirbas Vladimir Golkov Patrick Van Der Smagt Daniel Cremers Alexey Dosovitskiy, Philipp Fischer and Thomas Brox. Flownet: Learning optical flow with convolutional networks. in cvpr, 2015.

[ARY] NAREN ARYA. Build an api under 30 lines of code with python and flask url:https://impythonist.wordpress.com/2015/07/12/build-an-api-under-30-lines-of-code-with-python-and-flask/ (accessed: 01.06.2022).

[Bat] Syeda Aimen Batool. O introducere în redux și modul în care starea este actualizată într-o aplicație redux url:https://www.routech.ro/o-introducere-in-redux-si-modul-in-care-starea-este-actualizata-intr-o-aplicatie-redux/ (accessed: 01.06.2022).

[Cou] Coursera. What is python used for? a beginner's guide url:https://www.coursera.org/articles/what-is-python-used-for-a-beginners-guide-to-using-python (accessed: 01.06.2022).

[Duc77] Jean Duchon. Splines minimizing rotation-invariant seminorms in sobolev spaces.in constructive theory of functions of several variables, pages 85–100. springer,. 1977.

[HSX22] Sen He, Yi-Zhe Song, and Tao Xiang. Style-based global appearance flow for virtual try-on. 2022.

[HYL20] Xiaobao Guo Wei Liu Wangmeng Zuo Han Yang, Ruimao Zhang and Ping Luo. Towards photo-realistic virtual try-on by adaptively generating-preserving image content. 2020.

[KB15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2015.

[ORB15] Philipp Fischer Olaf Ronneberger and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. in miccai. 2015.

[PIE17] Tinghui Zhou Phillip Isola, Jun-Yan Zhu and Alexei A Efros. Image-to-image translation with conditional adversarial networks. 2017.

[pyt] pythonbasics. What is flask python url:https://pythonbasics.org/what-is-flask-python/ (accessed: 01.06.2022).

[SZ20] Karen Simonyan and Andrew Zisserman. Tdo not mask what you do not need to mask: a parser-free virtual try-on. 2020.

[SZ21] Yujun Shen and Bolei Zhou. Closed-form factorization of latent semantics in gans. in cvpr. 2021.

[TIC15] Jeremie Mary Thibaut Issenhuth and Clement Calauzenes. Very deep convolutional networks for large-scale image recognition. 2015.

[Tim] Tech With Tim. Mongodb + python 1 - crud, relationships and more url:https://www.youtube.com/watch?v=upszdgutpzct=1287s (accessed: 01.06.2022).

[TKA19] Samuli Laine Tero Karras and Timo Aila. A style-based generator architecture for generative adversarial networks. in cvpr. 2019.

[XHD18] Zhe Wu Ruichi Yu Xintong Han, Zuxuan Wu and Larry S Davis. Viton: An image-based virtual try-on network. in cvpr. 2018.

[XHS19] Weilin Huang Xintong Han, Xiaojun Hu and Matthew R Scott. Clothflow: A flow-based model for clothed person generation. 2019.

[YGL21] Ruimao Zhang Chongjian Ge Wei Liu Yuying Ge, Yibing Song and Ping Luo. Disentangled cycle consistency for highlyrealistic virtual try-on. in cvpr. 2021.