REPUBLIKA E SHQIPËRISË
UNIVERSITETI POLITEKNIK I TIRANËS
FAKULTETI I TEKNOLOGJISË SË INFORMACIONIT

PROJECT ANALYSIS & DESIGN DOCUMENT

TITULLI : ALBART - SISTEM PËR PROMOVIMIN DHE SHITJEN E VEPRAVE TË ARTIT

LËNDA : INXHINIERI SOFTI

PEDAGOGE : MSC.BESJANA MURAKU

ANËTARËT E GRUPIT :

1.AIRIN SHESTANI
2.ENI MARSELA
3.POLIKSENI STEFANI

Tiranë,më 11.01.2026

1. PARATHËNIE

Ky dokument paraqet përshkrimin e plotë të kërkesave funksionale, jofunksionale dhe teknike për zhvillimin e sistemit i cili synon të ofrojë një mjedis unik për prezantimin, shitjen dhe blerjen e veprave artistike. Platforma mundëson ndërveprimin ndërmjet artistëve,të cilët publikojnë dhe promovojnë krijimet e tyre dhe klientëve, të cilët eksplorojnë, blejnë dhe vlerësojnë veprat e ekspozuara.

Dokumenti është përgatitur për t'u përdorur nga:

- Ekipi i zhvillimit të sistemit, për të ndjekur kërkesat dhe specifikimet teknike gjatë fazave të implementimit.
- Përdoruesit e sistemit (artistët dhe klientët), për të kuptuar funksionet e sistemit dhe mënyrën e përdorimit të tij.
- Mbikëqyrësit akademikë, për të vlerësuar përmbajtjen dhe korrektësinë e kërkesave të përcaktuara.

Ky është versioni përfundimtar i dokumentimit të kërkesave, i cili përfshin çdo detaj të sistemit, duke ofruar një pasqyrë të plotë dhe të detajuar të funksionaliteteve, strukturës dhe proceseve të tij.

Ky dokument është i kompletuar dhe i finalizuar, dhe nuk priten më versione të ardhshme, duke reflektuar gjendjen përfundimtare të projektit dhe duke qenë i gatshëm për përdorim ose implementim.

2. HYRJE

2.1 Përshkrimi i sistemit

Sistemi i zhvilluar përfaqëson një platformë online për artin, e cila mundëson ndërveprim të drejtpërdrejtë ndërmjet artistëve dhe klientëve, duke lehtësuar procesin e prezantimit, shitjes dhe promovimit të veprave artistike në mënyrë digjitale.
Artistët kanë mundësinë të publikojnë punimet e tyre dhe të komunikojnë drejtpërdrejt me klientët, ndërsa këta të fundit mund të eksplorojnë veprat, të realizojnë blerje dhe të japin vlerësime për eksperiencën e tyre. Veprat që do të promovohen në platformë do të përfshijnë piktura, skulptura, punë artizanale,punë "handmade" etj.
Platforma ofron mekanizma të strukturuar për publikimin e veprave artistike, menaxhimin e profileve të përdoruesve dhe realizimin e transaksioneve, duke siguruar transparencë dhe besueshmëri në procesin e shitblerjes.Sistemi mbështet kategori të ndryshme veprash artistike dhe ofron funksionalitete shtesë si komunikimi i drejtpërdrejtë ndërmjet përdoruesve, vlerësimi i veprave dhe menaxhimi i porosive. Për mbrojtjen e të drejtave të autorit, platforma implementon mekanizma sigurie, përfshirë aplikimin e watermark-ut në materialet vizuale. Integrimi me shërbime të jashtme, si sistemi bankar dhe shërbimi postar, mundëson përfundimin e ciklit të plotë të blerjes në mënyrë të automatizuar.

2.2 Problemi që zgjidh

Në ditët e sotme, ku gjithçka është e digjitalizuar, shumë artistë përballen me vështirësi në promovimin dhe shitjen e veprave të tyre në mënyrë digjitale, për shkak të mungesës së një platforme të dedikuar që të mbulojë të gjithë procesin në mënyrë të strukturuar.Ndaj, lind nevoja për të kërkuar mënyra bashkëkohore për të shitur dhe promovuar veprat e tyre online.

Nga ana tjetër, klientët hasin vështirësi në gjetjen e veprave artistike në një formë të organizuar dhe të besueshme. Mungesa e informacionit, e mekanizmave të verifikimit dhe e një procesi të qartë blerjeje krijon pasiguri dhe ul besimin në blerjet online të veprave të artit.
Gjithashtu, mungesa e një sistemi të integruar që lidh publikimin e veprës, komunikimin, pagesën dhe dorëzimin në një rrjedhë të vetme krijon procese joefikase dhe rrit mundësinë e gabimeve ose keqkuptimeve ndërmjet palëve.

Ky sistem synon të adresojë këto problematika duke ofruar një platformë të centralizuar që e bën më të thjeshtë, më të sigurt dhe më të organizuar promovimin, blerjen dhe menaxhimin e veprave të artit në mjedis digjital duke lidhur kështu krijimtarinë artistike me teknologjinë moderne.

2.3 Qëllimi kryesor dhe përdoruesit e synuar

Qëllimi kryesor i këtij sistemi është të krijojë një platformë online për artin, e cila mundëson ndërveprim të drejtpërdrejtë ndërmjet artistëve dhe klientëve, duke lehtësuar procesin e prezantimit, shitjes dhe promovimit të veprave artistike në mënyrë digjitale. Sistemi synon të ofrojë një hapësirë të strukturuar dhe të besueshme ku artistët mund të prezantojnë krijimet e tyre, ndërsa klientët mund të blejnë vepra arti në mënyrë të lehtë dhe transparente.

Sistemi është projektuar për tu shërbyer disa grupe përdoruesish që kanë interes të drejtpërdrejtë në fushën e artit dhe tregtimin e veprave artistike.

Përdoruesit e synuar të sistemit janë:

- Personat e interesuar në art, të cilët përdorin sistemin për të eksploruar, blerë dhe vlerësuar veprat artistike.
- Artistët, të cilët përdorin platformën për të promovuar dhe shitur veprat e tyre, si dhe për të komunikuar me klientët.
- Biznese ose institucione të lidhura me artin, të cilët mund të përdorin platformën për të gjetur artistë dhe vepra në varësi të nevojës së tyre.

2.4 Funksionimi i sistemit me sisteme të tjera

Sistemi do të mund të ndërveprojë me :

- Sistemin e sigurisë – përdoret për ruajtjen e të dhenave të përdoruesve dhe mbrojtjen e të drejtave të autorit përmes watermark-ut.
- Sistemin bankar – përdoret për verifikimin, autorizimin dhe konfirmimin e transaksioneve të pagesave.
- Sistemin postar – mundëson dërgimin dhe ndjekjen e porosive deri në dorëzimin përfundimtar te klienti.

2.5 Përshtatja e sistemit me biznesin e përgjithshëm

Ky sistem kontribuon drejtpërdrejt në objektivat strategjike dhe operacionale të organizatës që e përdor, duke sjellë përfitime në disa aspekte kryesore:

1. Kontribut ekonomik dhe tregtar

• Rrit të ardhurat përmes shitjeve online të veprave artistike.
• Krijon mundësi të reja për bashkëpunim mes artistëve dhe klientëve ndërkombëtarë.

2. Kontribut në imazhin dhe praninë digjitale të organizatës

• Rrit prezantimin publik dhe vizibilitetin e organizatës në tregun e artit.
• Përmirëson marrëdhëniet me klientët përmes komunikimit të drejtpërdrejtë dhe feedback-ut.

3. Kontribut kulturor dhe krijues

• Promovon artistët e rinj dhe veprat origjinale, duke krijuar një hapësirë për zhvillim kulturor.
• Nxit ruajtjen e të drejtave të autorit dhe vlerësimin e artit autentik.

4. Kontribut në menaxhimin organizativ

• Siguron transparencë dhe kontroll mbi proceset e porosive, pagesave dhe ankesave.
• Lehtëson vendimmarrjen përmes raporteve mbi aktivitetin e përdoruesve dhe tregun.

3.PROJECT SCOPE

3.1. Cfarë përfshin sistemi?

Sistemi për promovimin dhe shitjen e veprave të artit është projektuar për të mbështetur të gjitha funksionet kryesore që lidhen me ndërveprimin ndërmjet artistëve dhe klientëve në një mjedis digjital të sigurt dhe të strukturuar.

Në kuadër të këtij projekti, sistemi përfshin disa funksionalitete të përgjithshme që vijojnë si më poshtë :

- Regjistrimi i përdoruesve sipas kategorive që përfaqësojnë (artist ose klient)
- Krijimi i profileve personale nga cdo përdorues me të dhëna sipas kërkesave specifike
- Ngarkimi i veprave artistike nga artistët me përshkrime të detajuara për to si edhe watermark unik për të mos cënuar të drejtat e autorësisë
- Blerja e produkteve artistike nga klientët
- Komunikimi i drejtpërdrejtë midis artistëve dhe klientëve përmes chat-it individual
- Mundësia e dhënies së vlerësimeve nga secili klient për artistët dhe veprat e tyre

3.2. Cfarë nuk përfshin sistemi?

Në këtë fazë zhvillimi, sistemi nuk përfshin disa funksionalitete që nuk lidhen drejtpërdrejt me qëllimin kryesor të projektit ose që parashikohen për faza të ardhshme zhvillimi.

Jashtë fushës së këtij projekti janë:

- Menaxhimi fizik i magazinimit, paketimit dhe transportit të produkteve artistike, i cili realizohet nga shërbimi postar.
- Pagesat jashtë sistemit elektronik, si pagesat cash ose pagesat fizike.
- Integrimi me platforma të jashtme të tregtisë elektronike ose rrjete sociale për shitje direkte.
- Mbështetja për shitje me ankand (auction system) ose forma të tjera alternative të shitjes.

4.FJALOR

Termi	Përshkrimi
Përdorues	Çdo individ që ka një llogari ekzistuese në sistem.
Artist	Përdorues i regjistruar që publikon veprat e tij dhe detaje për to në sistem me qëllim që të promovojë artin e tij.
Klient	Përdorues i regjistruar që shikon,blen,dhe vlerëson veprat e artistëve përmes platformës.
Autorizim	Proces që përcakton çfarë veprimesh mund të kryejë çdo përdorues, bazuar në rolin e tij (artist ose klient).
Autentifikim (Login)	Procesi i hyrjes në sistem përmes kredencialeve të kërkuara (email/ fjalëkalim etj).
Watermark	Shenjë unike digjitale që i shtohet automatikisht çdo vepre të ngarkuar nga artisti për mbrojtjen e të drejtës së autorit.
Terms & Conditions	Kushtet ligjore të përdorimit të platformës,që përdoruesi duhet t'i pranojë përpara regjistrimit.
Modul sistemi	Njësi funksionale e pavarur që mbulon një grup specifik kërkesash (p.sh. moduli i profilit, moduli i chat-it, moduli i pagesave).
Shopping Cart	Hapësira ku klienti ruan përkohësisht produktet që dëshiron të blejë përpara përfundimit të blerjes.
Sistem review	Funksionalitet që i lejon klientëve të vlerësojnë artistët pas një blerje duke dhënë komente.
Rate (Vlerësim)	Funksionalitet që i lejon klientit të japë një notë ose vlerësim (p.sh. 1–5 yje) për një artist pas blerjes së një produkti.
Support	Modul që merret me menaxhimin e ankesave, raportimeve dhe kërkesave për ndihmë nga klientët.
Transaksion	Proces financiar i kryer gjatë blerjes së një produkti përmes sistemit të lidhur me bankën.

5. PËRCAKTIMI I AKTORËVE

5.1 Aktorët

Aktori	**Përshkrimi i rolit**
Admini	Menaxhon sistemin, përdoruesit, problemet dhe politikat e sigurisë.
Artisti	Krijon profilin e tij, prezanton punën dhe eksperiencat e tij dhe shet produktet.
Klienti	Regjistrohet në sistem, zgjedh produktet që dëshiron të blejë dhe jep vlerësime për artistët.

5.2 Aktorët e jashtëm

Aktori	**Përshkrimi i rolit**
Banka	Menaxhon procesin e pagesave mes klientit dhe artistit, verifikon vlefshmërinë e pagesës dhe jep përgjigje për statusin e saj.
Posta	Menaxhon procesin e transportit të produkteve të blera dhe konfirmon përmbylljen e tij.

6. PËRCAKTIMI I KËRKESAVE FUNKSIONALE

Kodi	**Kërkesa**
KP1.0	Përdoruesi duhet të regjistrohet në kategorinë e tij (klient/ artist).
KP1.1	Për regjistrimin e artistit, duhet të kërkohen të dhënat personale si dhe eksperienca dhe certifikata të cilat do të shfaqen në profilin e tij.
KP1.2	Për regjistrimin e klientit, duhet të kërkohen të dhënat personale dhe preferencat në lidhje me produktet.
KP1.3	Gjatë procesit të regjistrimit, secilit përdorues do t'i shfaqen Terms & Conditions në lidhje me përdorimin e software-it, të cilat duhet t'i pranojë që të vazhdojë më tutje.

KP2.0	Artistët duhet të kenë mundësinë të shtojnë në profil veprat e tyre së bashku me detajet specifike.
KP2.1	Artistët duhet të kenë mundësinë të ruajnë të drejtën e autorit, pasi secila vepër do të pajiset me një watermark specific të sistemit.
KP3.0	Në momentin që një përdorues logohet duhet të shfaqet faqja kryesore ku paraqiten produkte të ndryshme në varësi të preferencave dhe modulet e tjera që mund të përzgjidhen.
KP3.1	Modulet e tjera do të përfshijnë kategori të tilla si: • Profili i përdoruesit: shfaq të dhënat e përdoruesit • Search: i jep përdoruesit mundësinë për të kërkuar produkte specifike • Shopping Cart: mban të gjitha produktet që përdoruesi ka shtuar
KP4.0	Klientët duhet të kenë mundësi të kenë kontakte të drejtpërdrejta me artistët (chat individual).
KP5.0	Klienti duhet të ketë mundësinë që pasi të blejë një produkt, të japë vlerësimin e tij për artistin (sistem review) si dhe të shprehë ankesat e tij tek supporti.
KP5.1	Supporti duhet të menaxhojë ankesat e klientëve dhe të sigurojë zgjidhjen e problematikave të paraqitura.
KP6.0	Përdoruesi duhet të ketë mundësinë të shtojë dhe të heqë produkte nga Cart dhe në momentin që do të bëjë një blerje të shkojë tek Checkout.
KP6.1	Për procesin e shit-blerjeve, përdoruesi duhet të ketë mundësinë për pagesë online përmes kartës bankare.
KP6.2	Banka duhet të verifikojë transaksionin dhe t'i kthejë statusin e tij përdoruesit.
KP7.0	Për procesin e transportit, klienti duhet t'i marrë porositë me anë të transportit që menaxhohet nga sistemi i shërbimit postar. Gjatë procesit të Checkout-it përdoruesi duhet të japë adresën dhe numrin e telefonit.
KP7.1	Posta duhet t'i kthejë informacion klientit mbi statusin e porosisë (në proces/ dorëzuar).
KP8.0	Përdoruesit duhet të marrin njoftime automatike (notifications) për veprime të rëndësishme.

7. PËRCAKTIMI I KËRKESAVE JOFUNKSIONALE

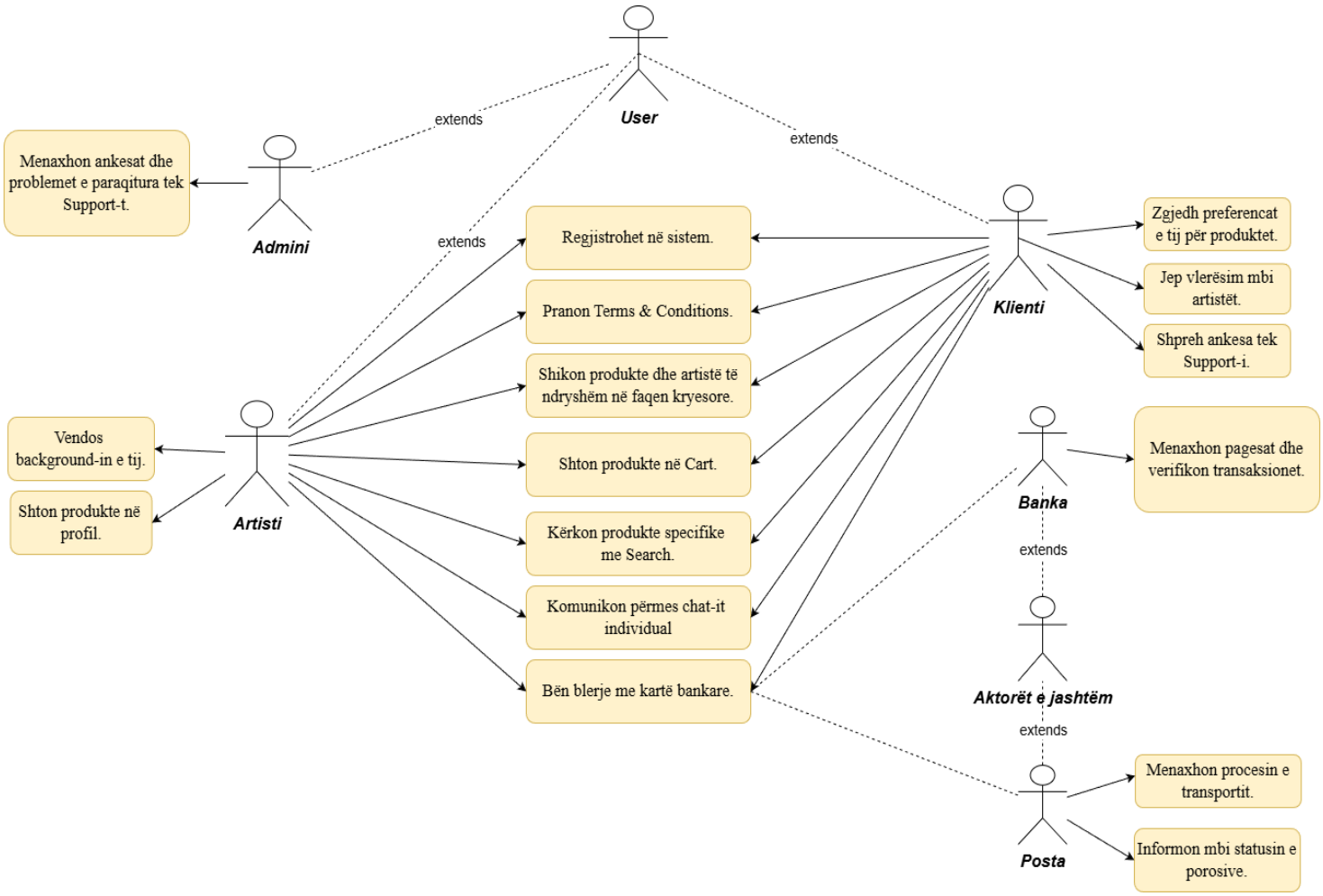
Kodi	Kategoritë	Përshkrimi
KJ1.0	Shpejtësia	Sistemi duhet të përballojë një numër të madh përdoruesish aktivë (deri në 1000 njëkohësisht) pa rënie të ndjeshme të performancës.
KJ1.1		Sistemi duhet të përgjigjet ndaj kërkesave të përdoruesit (p.sh ngarkimi i faqeve, kërkimi, shfaqja e produkteve) brenda maksimumi 3 sekondash.
KJ2.0	Madhësia	Sistemi duhet të menaxhojë hapësirën storage në mënyrë efikase, duke kompresuar imazhet e veprave të artit pa humbje të cilësisë vizuale.
KJ2.1		Të dhënat e përdoruesve, veprave dhe transaksioneve duhet të ruhen në një bazë të dhënash të optimizuar që mund të përballojë të paktën 10,000 përdorues aktivë.
KJ3.0	Thjeshtësia në përdorim	Ndërfaqja e sistemit duhet të jetë e thjeshtë, estetike dhe e lehtë për përdorim, në mënyrë që përdoruesit të mos kenë nevojë për trajnim paraprak.
KJ4.0	Besueshmëria	Sistemi duhet të garantojë sigurinë e të dhënave të përdoruesve përmes autentifikimit dhe enkriptimit të informacionit gjatë regjistrimit.
KJ4.1		Duhet të implementohet një sistem auditimi për gjurmimin e veprimeve (log actions) në mënyrë që çdo ndryshim të jetë i verifikueshëm dhe të ruhet për minimumi 12 muaj.
KJ5.0	Qëndrushmëria	Sistemi duhet të jetë i qëndrueshëm dhe i disponueshëm 24/7 dhe të ketë rikuperim automatik pas ndërprerjeve.
KJ6.0	Transporueshmëria	Sistemi duhet të jetë i aksesueshëm nga çdo pajisje dhe të funksionojë me browser të ndryshëm si dhe sisteme të ndryshme operative.
KJ6.1		Kodi burimor duhet të jetë i dokumentuar dhe i standardizuar për t'u transferuar lehtësisht në sisteme të tjera (P.sh aplikacioni mobile)

8.SPECIFIKIMI I KËRKESAVE TË SISTEMIT

Kodi	Përshkrimi i kërkesës	Lidhja me KP
KS1.0	Sistemi duhet të ofrojë një modul regjistrimi që identifikon përdoruesit (artistë ose klientë) gjatë krijimit të llogarisë.	KP1.0
KS1.1	Sistemi duhet të ruajë në bazën e të dhënave informacionet personale të artistit si emri, përshkrimi, eksperienca dhe certifikatat e ngarkuara.	KP1.1
KS1.2	Sistemi duhet të ruajë informacionet personale të klientit dhe preferencat e tij për kategori arti ose çmime.	KP1.2
KS1.3	Sistemi duhet të shfaqë automatikisht dokumentin "Terms & Conditions" gjatë regjistrimit dhe të regjistrojë pranim elektronik (checkbox) për secilin përdorues.	KP1.3
KS2.0	Sistemi duhet të lejojë artistët të ngarkojnë vepra arti në profilin e tyre, me informacione për to si titulli, përshkrimi, kategoria, çmimi dhe imazhi.	KP2.0
KS2.1	Sistemi duhet të aplikojë automatikisht një watermark unik në çdo imazh të ngarkuar për mbrojtje të autorësisë së artistëve.	KP2.1
KS3.0	Pas hyrjes në sistem, aplikacioni duhet të ngarkojë faqen kryesore që përmban produktet e rekomanduara sipas preferencave të përdoruesit.	KP3.0
KS3.1	Sistemi duhet të ofrojë module funksionale të ndara sic janë: • Profili i përdoruesit • Search • Shopping Cart	KP3.1
KS4.1	Sistemi duhet të implementojë një modul komunikimi "chat" përmes të cilit klientët mund të komunikojnë drejtpërdrejt me artistët.	KP4.0
KS5.0	Sistemi duhet të mundësojë që pas përfundimit të një blerjeje, klienti të dërgojë një vlerësim (review) për artistin dhe për produktin e blerë.	KP5.0
KS5.1	Sistemi duhet të ofrojë një panel administrativ për Support-in, që lejon menaxhimin e ankesave dhe përditësimin e statusit të tyre (në shqyrtim, zgjidhur, refuzuar).	KP5.1
KS6.0	Sistemi duhet të ruajë produktet e përzgjedhura nga klienti në Shopping Cart deri në momentin që përdoruesi vendos të bëjë porosinë. Sistemi më pas duhet ta dërgojë përdoruesin në Checkout.	KP6.0
KS6.1	Sistemi duhet të integrohet me një sistem bankar për procesimin e pagesave (autorizim, debitim, konfirmim).	KP6.1
KS6.2	Sistemi duhet të marrë përgjigje nga banka për çdo transaksion dhe të ruajë statusin e tij ("i suksesshëm", "i dështuar") në bazën e të dhënave.	KP6.2
KS7.0	Sistemi duhet të integrohet me një shërbim postar për dërgimin e porosive. Në momentin që kryhet një porosi, adresa dhe të dhëna të tjera të nevojshme të klientit duhet t'i dërgohen automatikisht postës.	KP7.0
KS7.1	Sistemi duhet të marrë dhe të ruajë informacion nga posta mbi statusin e porosisë (në proces/dorëzuar) dhe ta shfaqë në profilin e klientit.	KP7.1
KS8.0	Sistemi duhet të dërgojë mesazhe automatike për të njoftuar përdoruesit për veprime të rëndësishme si: porosi të reja, mesazhe, vlerësime, ndryshime statusi.	KP8.0

9.USE CASE DIAGRAM

9.1. Diagrama Use Case



9.2. Përshkrimi për cdo Use Case

9.2.1. Regjistrimi i përdoruesit

Qëllimi: Të mundësojë krijimin e një llogarie të re në sistem për përdoruesit (artistë ose klientë).
Aktorë: Përdorues (Artist / Klient)
Përshkrimi: Ky use case përshkruan procesin përmes të cilit përdoruesi regjistrohet në sistem duke plotësuar të dhënat e nevojshme dhe duke përcaktuar rolin e tij.
Rrjedha kryesore:
1.Përdoruesi zgjedh opsionin “Sign up”.
2.Sistemi shfaq formularin e regjistrimit.
3.Përdoruesi plotëson të dhënat e kërkuara.
4.Përdoruesi pranon Terms & Conditions.
5.Sistemi krijon llogarinë dhe konfirmon regjistrimin.

9.2.2. Shfaqja e produkteve dhe artistëve

Qëllimi: Të lejojë përdoruesin të eksplorojë produktet dhe artistët e disponueshëm në platformë.
Aktorë: Përdorues (Artist / Klient)
Përshkrimi: Sistemi shfaq në faqen kryesore një listë të produkteve artistike dhe profileve të artistëve.
Rrjedha kryesore:
1.Përdoruesi hap faqen kryesore.
2.Sistemi shfaq produktet dhe artistët.
3.Përdoruesi shikon produktet që i interesojnë.

9.2.3. Kërkimi i produkteve

Qëllimi: Të mundësojë gjetjen e produkteve specifike në mënyrë të shpejtë.
Aktorë: Përdorues (Artist / Klient)
Përshkrimi: Ky use case i lejon përdoruesit të kërkojë produkte duke përdorur fjalë kyçe ose kategori.
Rrjedha kryesore:
1.Përdoruesi vendos kriteret e kërkimit.
2.Sistemi filtron produktet.
3.Sistemi shfaq rezultatet përkatëse.

9.2.4. Shtimi i produkteve në Shopping Cart

Qëllimi: Të ruajë përkohësisht produktet e përzgjedhura për blerje.
Aktorë: Klienti
Përshkrimi: Klienti shton produkte në shportën e blerjeve përpara realizimit të pagesës.
Rrjedha kryesore:
1.Klienti zgjedh produktin.
2.Klienti shtyp “Add to Cart”.
3.Sistemi e shton produktin në shportë.

9.2.5. Realizimi i blerjes me kartë bankare

Qëllimi: Të përfundojë procesin e blerjes përmes pagesës elektronike.
Aktorë: Klienti, Banka
Përshkrimi: Ky use case përshkruan procesin e pagesës dhe verifikimit të transaksionit.
Rrjedha kryesore:
1.Klienti konfirmon shportën e blerjeve.
2.Sistemi kërkon të dhënat e pagesës.
3.Sistemi bankar verifikon transaksionin.
4.Sistemi konfirmon blerjen.

9.2.6. Komunikimi përmes chat-it individual

Qëllimi: Të mundësojë komunikimin direkt mes klientit dhe artistit.
Aktorë: Klienti, Artisti
Përshkrimi: Sistemi ofron chat individual për sqarime rreth produkteve.
Rrjedha kryesore:
1.Klienti hap chat-in dhe kerkon artistin qe do.
2.Artisti dhe klienti shkëmbejnë mesazhe.
3.Sistemi ruan historikun e komunikimit.

9.2.7. Vlerësimi i artistëve

Qëllimi: Të lejojë klientin të japë review pas blerjes.
Aktorë: Klienti
Përshkrimi: Klienti vlerëson artistin dhe produktin e blerë.
Rrjedha kryesore:
1.Klienti zgjedh porosinë e përfunduar.
2.Klienti jep vlerësim dhe koment per artistin.
3.Sistemi e publikon vlerësimin.

9.2.8. Ngarkimi i produkteve në profil

Qëllimi: Të lejojë artistin të publikojë veprat e tij.
Aktorë: Artist
Përshkrimi: Artisti ngarkon produkte dhe përshkrimet përkatëse në profilin personal.
Rrjedha kryesore:
1.Artisti zgjedh “Add Product”.
2.Plotëson të dhënat e produktit.
3.Sistemi e publikon produktin.

9.2.9. Menaxhimi i kërkesave dhe ankesave te Supporti

Qëllimi: Të trajtojë problemet dhe ankesat e përdoruesve.
Aktorë: Klienti, Admini
Përshkrimi: Përdoruesit raportojnë ankesa që trajtohen nga Support-i.
Rrjedha kryesore:
1.Klienti paraqet ankesën.
2.Support-i analizon problemin.
3.Support-i jep zgjidhje ose përgjigje.

9.2.10. Menaxhimi i procesit të transportit

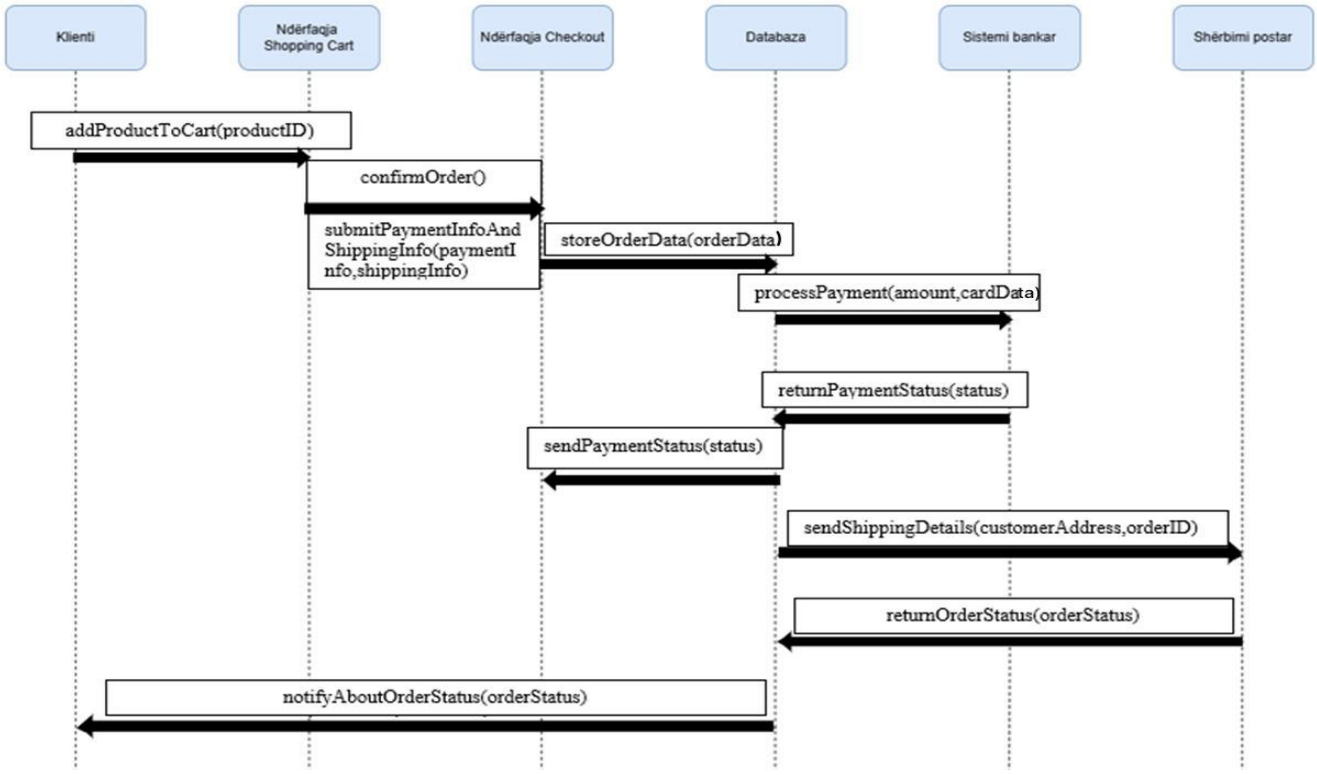
Qëllimi: Të realizojë dërgimin e produkteve tek klienti.
Aktorë: Posta
Përshkrimi: Shërbimi postar menaxhon transportin e porosive.
Rrjedha kryesore:
1.Sistemi njofton shërbimin postar.
2.Produkti transportohet.
3.Dorëzimi përfundohet.

9.2.11. Informimi mbi statusin e porosive

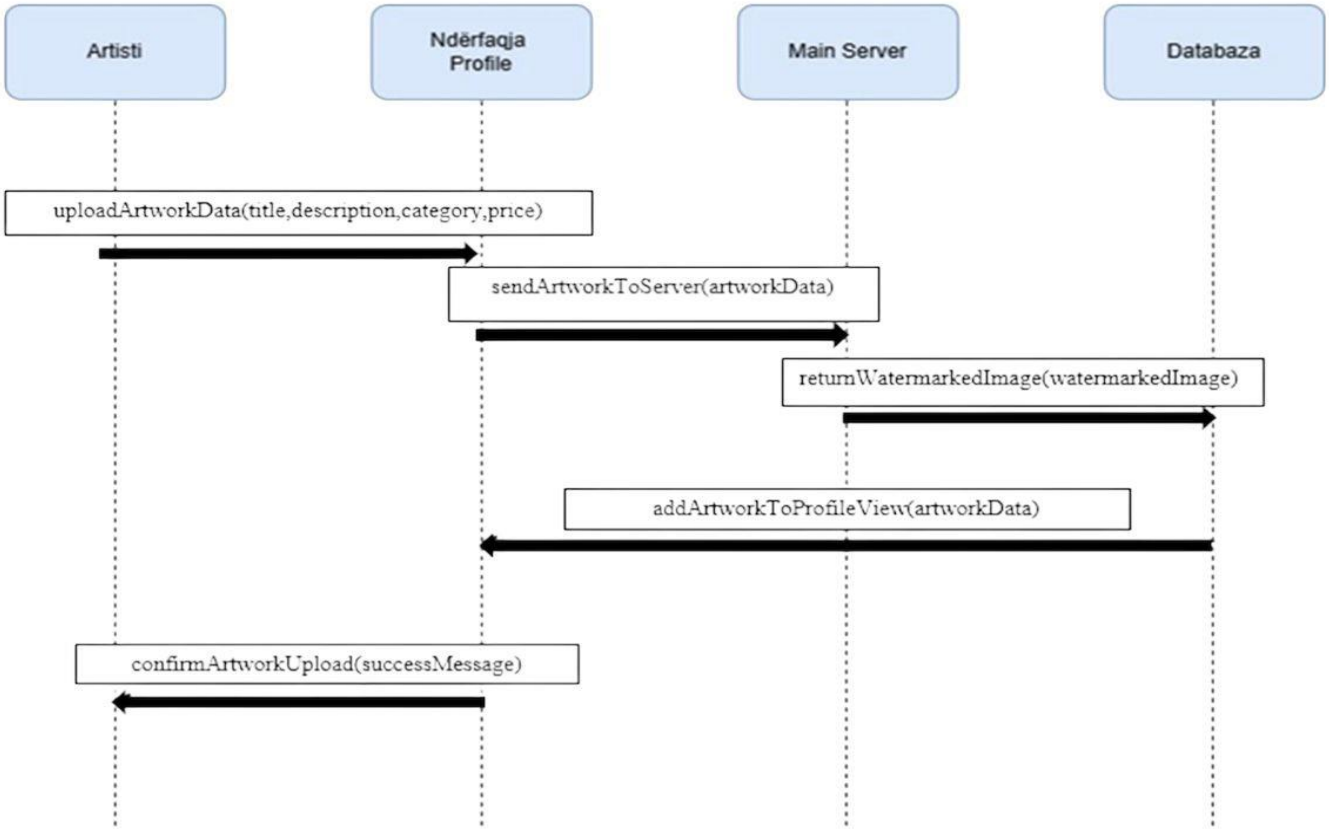
Qëllimi: Të informojë klientin mbi gjendjen e porosisë.
Aktorë: Sistemi, Klienti
Përshkrimi: Sistemi përditëson statusin e porosisë gjatë procesit të transportit.
Rrjedha kryesore:
1.Sistemi përditëson statusin.
2.Klienti merr njoftim.
3.Klienti kontrollon porosinë**.**

10.SEQUENCE DIAGRAMS

10.1. SEQUENCE DIAGRAM 1 – Procesi i blerjes së një produkti nga një klient



10.2. SEQUENCE DIAGRAM 2 – Procesi i ngarkimit të një produkti nga një artist



10.3. Shpjegimi i fluksit të ekzekutimit

10.3.1. Fluksi i ekzekutimit për SEQUENCE DIAGRAM 1

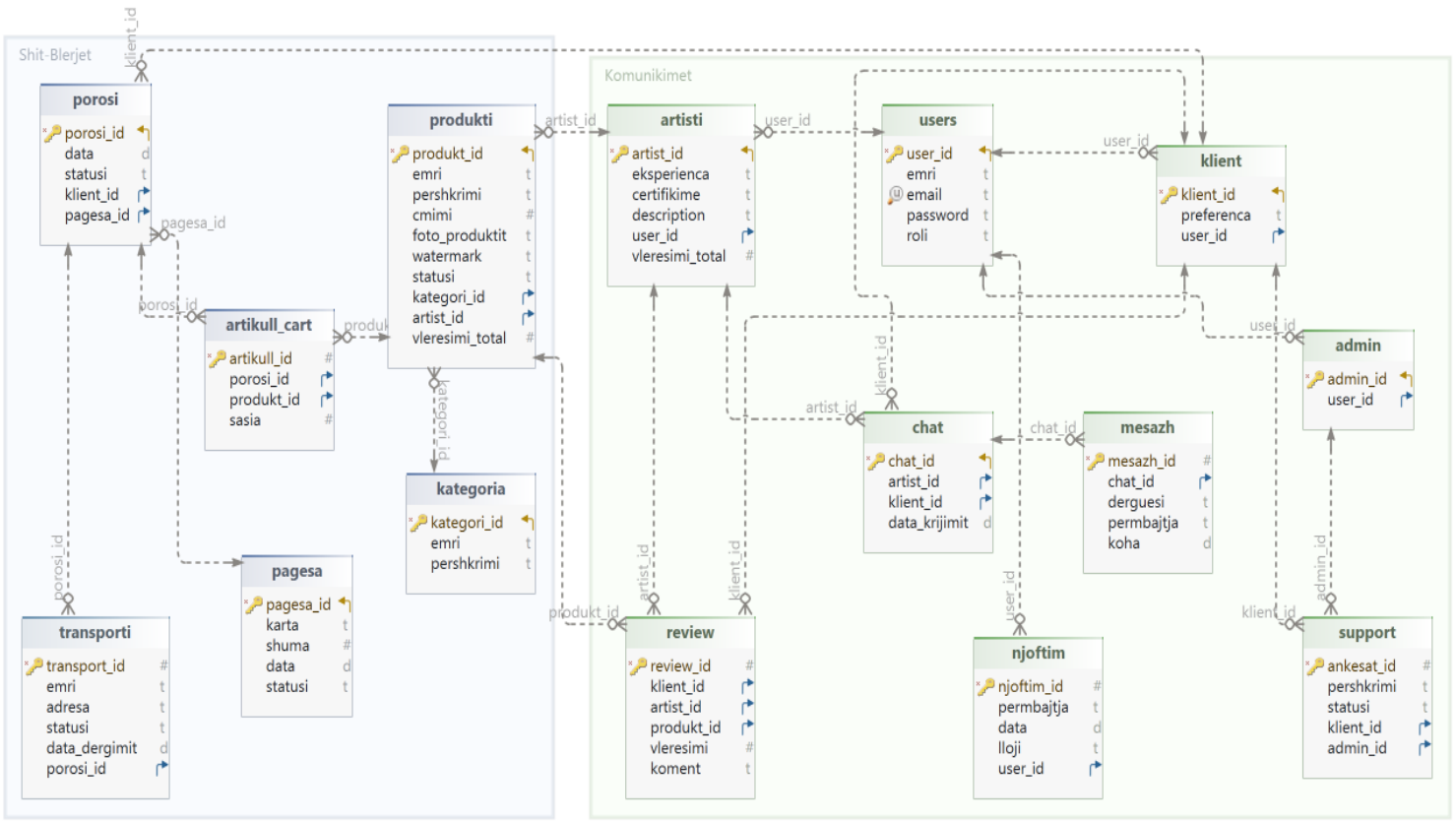
Fluksi i ekzekutimit fillon në momentin kur klienti hyn në platformë dhe ndërvepron me ndërfaqen e sistemit për të shfletuar produktet artistike të disponueshme. Sistemi i shfaq klientit informacionin përkatës të produktit të përzgjedhur, duke përfshirë përshkrimin dhe çmimin.
Pas shqyrtimit të produktit, klienti zgjedh opsionin për ta shtuar atë në shopping cart. Sistemi e regjistron këtë veprim dhe përditëson përmbajtjen e shportës. Klienti mund të vazhdojë me përzgjedhjen e produkteve të tjera ose të kalojë në fazën e përfundimit të blerjes.
Në fazën e konfirmimit, klienti vazhdon me procesin e pagesës. Sistemi kërkon të dhënat e nevojshme të pagesës dhe dërgon një kërkesë drejt sistemit bankar për verifikimin dhe autorizimin e transaksionit. Sistemi bankar përpunon kërkesën dhe kthen përgjigjen përkatëse.
Në rast të një transaksioni të suksesshëm, sistemi regjistron porosinë në databazë, përditëson statusin e saj dhe njofton klientin për përfundimin e blerjes. Në fund të fluksit, sistemi përgatit informacionin e nevojshëm për procesin e transportit dhe e kalon porosinë në fazën e dërgimit.

10.3.2. Fluksi i ekzekutimit për SEQUENCE DIAGRAM 2

Fluksi i ekzekutimit nis kur artisti hyn në sistem dhe zgjedh funksionalitetin për ngarkimin e një produkti të ri artistik. Sistemi i shfaq artistit formularin përkatës për plotësimin e të dhënave të produktit. Artisti plotëson informacionet e nevojshme, si emri i veprës, përshkrimi, çmimi dhe ngarkon imazhin përkatës të produktit. Pas përfundimit të këtij hapi, sistemi kryen validimin e të dhënave të futura për të siguruar korrektësinë dhe saktësinë e informacionit.
Më pas, sistemi aplikon mekanizmat e sigurisë për mbrojtjen e veprës artistike, duke shtuar watermark-un digjital mbi imazhin e produktit. Pas përpunimit të suksesshëm, sistemi ruan produktin në databazë.
Në përfundim të fluksit, sistemi publikon produktin në profilin e artistit dhe e bën atë të disponueshëm për shikim dhe blerje nga klientët e platformës.

11.ENTITY RELATIONSHIP (ER) DIAGRAM

11.1. Entitetet e databazës



11.2 Marrëdhëniet dhe kardinaliteti

11.2.1. Trashëgimia

Trashëgimia përfaqëson një marrëdhënie hierarkike midis klasave, në të cilën klasat fëmijë trashëgojnë atributet dhe karakteristikat e klasës prind.

Në këtë sistem, entiteti **users** shërben si klasa bazë që përmban informacionin e përgjithshëm të çdo përdoruesi.

Nga kjo klasë trashëgohen rolet specifike të përdoruesve: **klient, artisti dhe admin**. Secila prej këtyre klasave trashëgon atributet bazë të identifikimit dhe autentikimit nga users dhe shton fusha specifike që pasqyrojnë funksionet e tyre përkatëse brenda sistemit.

Struktura e tille mundëson ndarjen e qartë midis të dhënave të përbashkëta dhe atyre specifike për secilin rol, duke siguruar që ndërveprimet të realizohen sa më mirë.

11.2.2. Marrëdhëniet specifike

Tabela KLIENT

- **Klient - Porosi**

- **Marrëdhënia:** one-to-many (1:N)
- Një klient mund të realizojë disa porosi gjatë përdorimit të sistemit, ndërsa çdo porosi lidhet me një klient të vetëm.

- **Klient - Review**

- **Marrëdhënia:** one-to-many (1:N)
- Një klient mund të japë disa vlerësime për produkte ose artistë të ndryshëm.

- **Klient - Chat**

- **Marrëdhënia:** one-to-many (1:N)
- Një klient mund të hapë disa biseda për të komunikuar me artistë të ndryshëm.

- **Klient - Support**

- **Marrëdhënia:** one-to-many (1:N)
- Një klient mund të paraqesë disa kërkesa supporti për probleme ose paqartësi të ndryshme.

Tabela ARTISTI

- Artisti - Produkti

- **Marrëdhënia:** one-to-many (1:N)
- Një artist mund të publikojë disa produkte artistike, ndërsa çdo produkt krijohet nga një artist i vetëm.

- **Artisti - Review**

- **Marrëdhënia:** one-to-many (1:N)
- Një artist mund të marrë disa review nga klientë të ndryshëm për veprat e tij.

- **Artisti - Chat**

- **Marrëdhënia:** one-to-many (1:N)
- Një artist mund të marrë pjesë në disa biseda për të komunikuar me klientë të ndryshëm.

Tabela ADMIN

- **Admin - Support**

- **Marrëdhënia:** one-to-many (1:N)
- Një administrator mund të trajtojë disa kërkesa supporti.

Tabela PRODUKTI

- **Produkti - Artikull_Cart**

- **Marrëdhënia:** one-to-many (1:N)
- Një produkt mund të shfaqet në disa artikuj shporte brenda porosive të ndryshme.

- **Produkti - Review**

- **Marrëdhënia:** one-to-many (1:N)
- Një produkt mund të ketë disa vlerësime nga klientë të ndryshëm.

- **Produkti - Kategoria**

- **Marrëdhënia:** many-to-one (N:1)
- Përshkrim: Çdo produkt klasifikohet në një kategori të vetme.

Tabela POROSI

- **Porosi - Artikull_Cart**

- **Marrëdhënia:** one-to-many (1:N)
- Një porosi përmban disa artikuj që përfaqësojnë produktet e përzgjedhura.

- **Porosi - Pagesa**

- **Marrëdhënia:** one-to-one (1:1)
- Çdo porosi shoqërohet me një pagesë të vetme përkatëse.

- **Porosi - Transporti**

- **Marrëdhënia:** one-to-one (1:1)
- Çdo porosi ka një proces transporti për dorëzimin e produkteve.

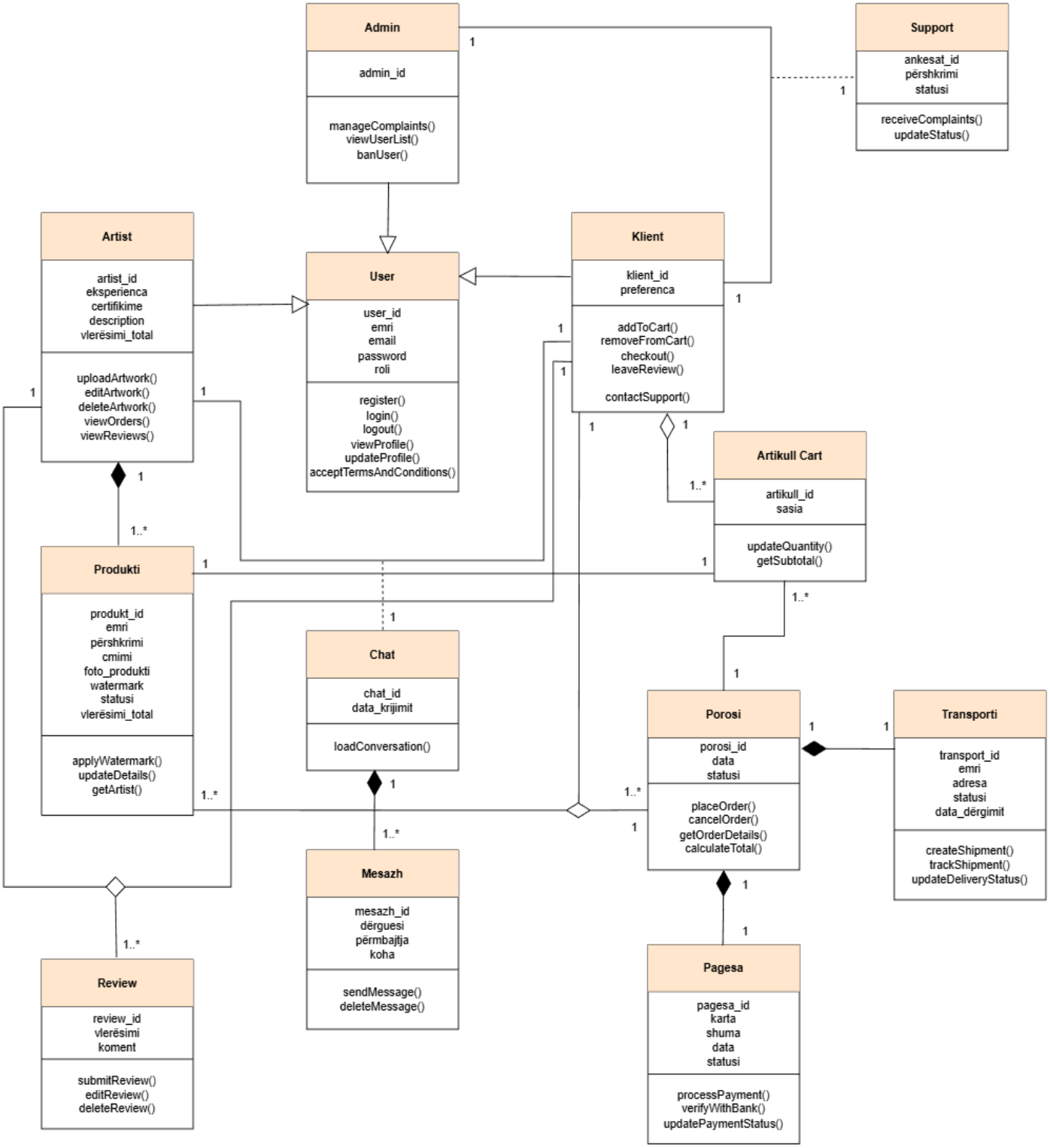
Tabela CHAT

- **Chat - Mesazh**

- **Marrëdhënia:** one-to-many (1:N)
- Një chat përmban disa mesazhe të dërguara gjatë komunikimi

12.CLASS DIAGRAM

12.1. Paraqitja me diagram



12.2. Marrëdhëniet midis klasave

Class Diagram-i përfaqëson strukturën statike të sistemit dhe paraqet klasat kryesore, atributet e tyre, metodat dhe marrëdhëniet ndërmjet tyre. Marrëdhëniet e paraqitura përfshijnë trashëgimi, asociim, agregim dhe kompozim.

- Klasat Artist, Klient dhe Admin trashëgojnë nga klasa bazë User.
  Kjo marrëdhënie trashëgimie mundëson ndarjen e atributeve dhe metodave të përbashkëta.Trashëgimia redukton përsëritjen e kodit dhe siguron një strukturë të zgjerueshme për rolet e sistemit.

- Klasa Artist ka një marrëdhënie kompozimi one-to-many me klasën Produkti.
  Një artist mund të krijojë dhe menaxhojë disa produkte, ndërsa çdo produkt i përket vetëm një artisti. Marrëdhënia është kompozim, pasi ekzistenca e produktit varet nga artisti përkatës.

- Klasa Klient ka një marrëdhënie one-to-many me klasën Porosi.
  Një klient mund të realizojë disa porosi, ndërsa çdo porosi i përket një klienti të vetëm.

- Klasa Porosi ka një marrëdhënie agregimi one-to-many me klasën Artikull_Cart.
  Një porosi përmban disa artikuj shporte, ndërsa çdo artikull shporte lidhet me një porosi të vetme. Kjo strukturë realizon ndarjen e marrëdhënies many-to-many midis porosisë dhe produktit.

- Klasa Artikull_Cart ka një marrëdhënie many-to-one me klasën Produkti.
  Çdo artikull shporte i referohet një produkti të vetëm, ndërsa një produkt mund të shfaqet në disa artikuj shporte të ndryshme. Kjo lidhje lejon menaxhimin e sasisë dhe çmimit për çdo porosi.

- Klasa Porosi ka një marrëdhënie kompozimi one-to-one me klasën Pagesa.
  Çdo porosi ka një pagesë të vetme dhe ekzistenca e pagesës varet nga porosia. Kjo marrëdhënie siguron integritetin e procesit të pagesës.

- Klasa Porosi ka një marrëdhënie kompozimi one-to-one me klasën Transporti.
  Çdo porosi shoqërohet me një proces transporti unik. Transporti nuk mund të ekzistojë pa porosinë përkatëse.

- Komunikimi midis Klientit dhe Artistit realizohet përmes klasës Chat.
  Klasa Chat vepron si ndërmjetëse dhe lidhet me Klientin dhe Artistin përmes marrëdhënieve many-to-one, duke mundësuar komunikimin.

- Klasa Chat ka një marrëdhënie kompozimi one-to-many me klasën Mesazh.
  Një chat përmban disa mesazhe, ndërsa çdo mesazh i përket një chat-i të vetëm. Kjo lidhje siguron ruajtjen e historikut të komunikimit.

- Klasa Review ka marrëdhënie many-to-one me klasat Klient, Produkti dhe Artisti.
  Çdo review krijohet nga një klient dhe lidhet me një produkt dhe një artist të caktuar. Kjo strukturë mundëson vlerësimin e cilësisë së produkteve dhe përvojës së blerjes.

- Klasa Support ka një marrëdhënie many-to-one me klasën Klient dhe klasën Admin.
  Një klient mund të hapë disa kërkesa supporti, ndërsa çdo kërkesë trajtohet nga një admin i vetëm. Kjo lidhje siguron menaxhim të centralizuar të problemeve.

- Klasa Admin ka marrëdhënie asociimi me komponentët kryesorë të sistemit.
  Admini ka rol mbikëqyrës dhe menaxhues, duke siguruar kontroll, siguri dhe mirëmbajtje të platformës, pa ndërhyrë drejtpërdrejt në rrjedhën e biznesit.

13.ARKITEKTURA E SISTEMIT

13.1. Arkitektura e përzgjedhur

Arkitektura e sistemit të platformës për shitjen dhe blerjen e veprave të artit është projektuar duke ndjekur modelin e arkitekturës së shtresëzuar (Layered Architecture). Ky model siguron ndarje të qartë të përgjegjësive, modularitet të lartë dhe mirëmbajtje të lehtësuar, duke i organizuar funksionalitetet e sistemit në katër shtresa kryesore:

- User Interface
- Configuration Services
- Core Business Logic/Application Functionality
- System Support.

13.2. Shpjegimi i shtresave dhe përgjegjësive

Në shtresën **User Interface**, përdoruesit ndërveprojnë me platformën përmes ndërfaqeve të specializuara si: homepage, galeria e veprave, profili i artistit, ndërfaqja e chat-it, shporta e blerjeve dhe faqja e checkout-it. Kjo shtresë fokusohet në paraqitjen vizuale dhe aksesin e përdoruesve ndaj funksioneve të sistemit.

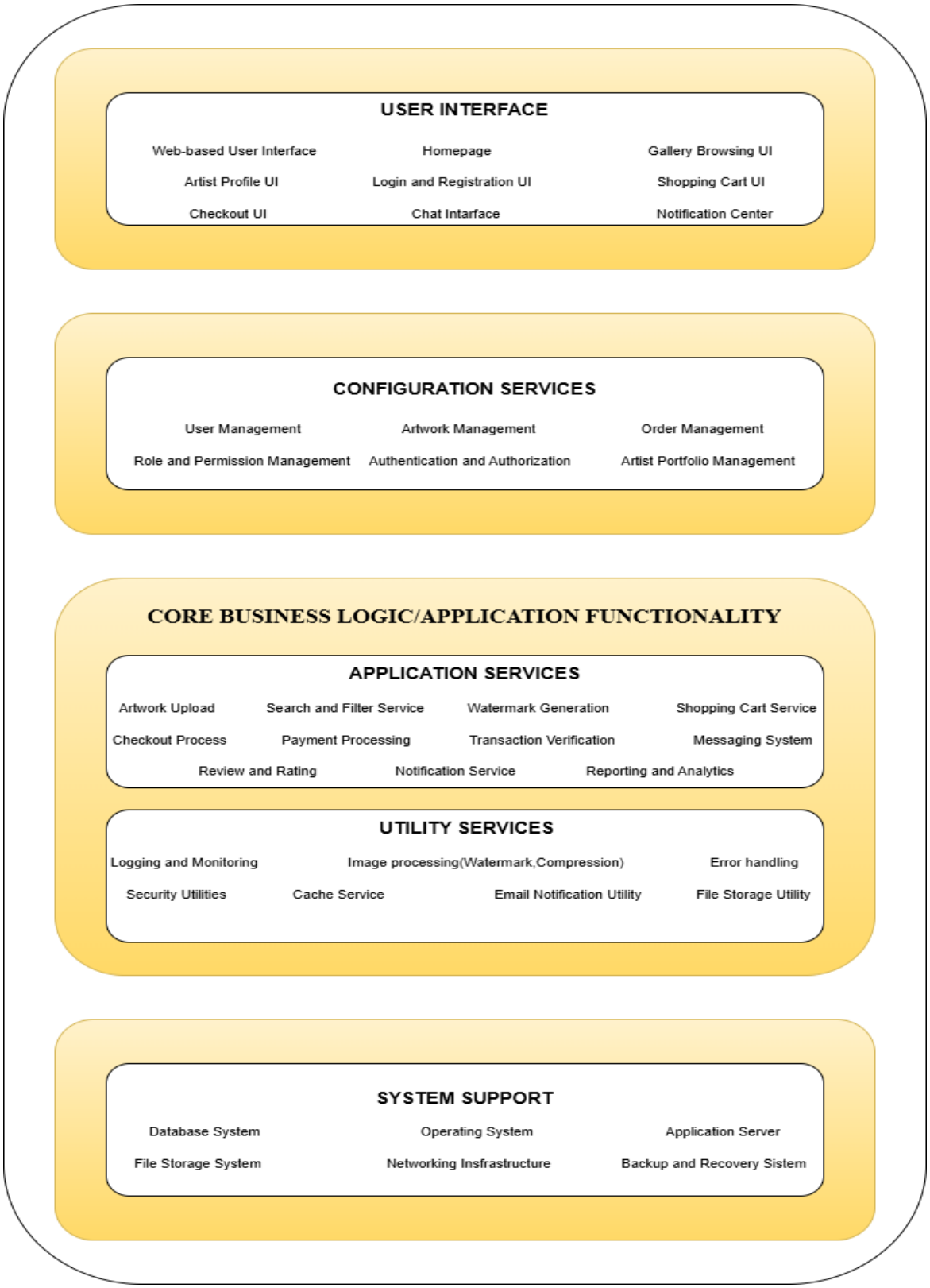
Shtresa **Configuration Services** përmban komponentët që menaxhojnë konfigurimin funksional të sistemit, duke përfshirë: menaxhimin e përdoruesve, menaxhimin e veprave të artit, menaxhimin e porosive, menaxhimin e roleve dhe autorizimeve, si dhe menaxhimin e portofolit të artistit. Kjo shtresë vepron si një urë që lidh ndërfaqen e përdoruesit me logjikën kryesore të aplikacionit duke siguruar siguri dhe kontroll të qasshmërisë.

Shtresa **Core Business Logic/Application Functionality** është shtresa më thelbësore, e cila përmban modulin e aplikacionit dhe shërbimet utilitare. Këtu përpunohen funksionet më kritike të platformës, si: ngarkimi i veprave, kërkimi, filtrimi, gjenerimi i watermark-ut, procesimi i pagesave, verifikimi i transaksioneve, sistemi i mesazheve, procesi i checkout-it, vlerësimet, njoftimet dhe analiza e të dhënave. Shërbimet utilitare mbështesin funksionalitetet kryesore përmes logimit, procesimit të imazheve, trajtimit të gabimeve, sigurisë dhe ruajtjes së skedarëve.

Shtresa **System Support** përfaqëson bazën teknike të sistemit dhe përfshin databazën, sistemin e ruajtjes së skedarëve, serverin e aplikacionit, sistemin operativ, infrastrukturën e rrjetit dhe mekanizmat e backup-it. Kjo shtresë siguron funksionimin e pandërprerë të aplikacionit, integrimin me shërbime të jashtme dhe ruajtjen e të dhënave.

Kjo strukturë e ndarë në shtresa siguron funksionim të qëndrueshëm, rrit modularitetin, thjeshton mirëmbajtjen dhe mundëson zhvillim të mëtejshëm të sigurt dhe të shkallëzuar të sistemit.

13.3. Paraqitja me diagram e arkitekturës



13.4. Përbërësit arkitekturorë që ripërdoren

Përbërësit e ripërdorshëm janë module ose njësi funksionale të sistemit që përdoren nga disa pjesë të ndryshme të aplikacionit, duke rritur efikasitetin, reduktuar duplikimin e kodit dhe thjeshtuar mirëmbajtjen.

Më poshtë janë përbërësit kryesorë të ripërdorshëm të platformës për shitjen dhe blerjen e veprave të artit :

1. Moduli i Autentifikimit dhe Autorizimit
Ky modul përdoret për të verifikuar identitetin e përdoruesve dhe për të kontrolluar aksesin e tyre në funksione specifike.Mund të ripërdoret nga të gjitha pjesët e sistemit që kërkojnë hyrje të sigurt, duke përfshirë: Cart-in, chat-in, ngarkimin e veprave, menaxhimin e porosive dhe njoftimet.

2. Moduli i Përpunimit të Imazheve
Ky modul përfshin krijimin e watermark-ut, kompresimin e imazheve dhe gjenerimin e thumbnail-eve. Ai përdoret sa herë që një artist ngarkon ose përditëson një vepër, por edhe gjatë shfaqjes së veprave në galeri për të ofruar versione të optimizuara të imazheve.

3. Moduli i Njoftimeve
Ky modul është përgjegjës për gjenerimin dhe shpërndarjen e njoftimeve në pjesë të ndryshme të sistemit, përfshirë njoftimet për porosi të reja, konfirmimet e pagesave, mesazhet e reja në chat, vlerësimet e përdoruesve si dhe përditësimet e statusit të porosive. Ai është ndërtuar si një komponent i ripërdorshëm që integrohet me module të ndryshme të platformës, duke mundësuar komunikim të centralizuar dhe koherent ndërmjet pjesëve të sistemit.

4. Moduli i Pagesave
Ky modul realizon komunikimin me procesorin e pagesave, verifikimin e transaksioneve dhe gjenerimin e faturave. Ripërdoret në procesin e checkout-it si edhe në funksione administrative që lidhen me monitorimin e pagesave.

5. Moduli i Chat-it
Ky modul mundëson komunikimin në kohë reale ndërmjet përdoruesve të platformës (artistëve dhe klientëve). Ai shërben si komponent i ripërdorshëm në disa pjesë të sistemit, duke u integruar me mekanizmat e njoftimeve, menaxhimin e sesioneve aktive me qëllim garantimin e një eksperience të qëndrueshme dhe të sigurt komunikimi.

6. Moduli i Menaxhimit të Përdoruesve
Ky modul përfshin regjistrimin, ruajtjen e profileve, përditësimin e kredencialeve dhe menaxhimin e roleve. Ripërdoret në shumë fusha, si menaxhimi i portofolit të artistit, menaxhimi i porosive dhe proceset e autorizimit.

7.Moduli i Logimit dhe Monitorimit
Ky modul është përgjegjës për regjistrimin dhe monitorimin e aktiviteteve kritike të sistemit, duke mundësuar identifikimin e gabimeve, analizën e performancës dhe rritjen e nivelit të sigurisë.Ripërdoret në module si pagesat, porositë, login-i, ngarkimi i veprave dhe procesimi i transaksioneve.

8.Moduli i Cache-it dhe Utility-t
Përdoren për të përmirësuar performancën duke ruajtur të dhëna të përdorura shpesh, si lista e veprave, rezultatet e kërkimit dhe preferencat e përdoruesve. Këto komponentë ripërdoren në shërbimet e kërkimit, galeri, filtrimit dhe profileve.

13.5. Versionimi i kodit nëpërmjet GIT

Për menaxhimin dhe versionimin e kodit burimor të këtij projekti është përdorur sistemi i kontrollit të versioneve **Git.** Projekti është ruajtur në një repository online, i cili mundëson ruajtjen e historikut të ndryshimeve dhe gjurmueshmërinë e zhvillimit të kodit gjatë fazave të ndryshme të projektimit dhe implementimit.

Gjatë zhvillimit, çdo ndryshim i rëndësishëm në kod është regjistruar përmes commit-eve përkatëse, të shoqëruara me përshkrime që pasqyrojnë funksionalitetin e shtuar ose përmirësuar. Kjo qasje ka kontribuar në organizimin më të mirë të kodit dhe në ruajtjen e stabilitetit të projektit.

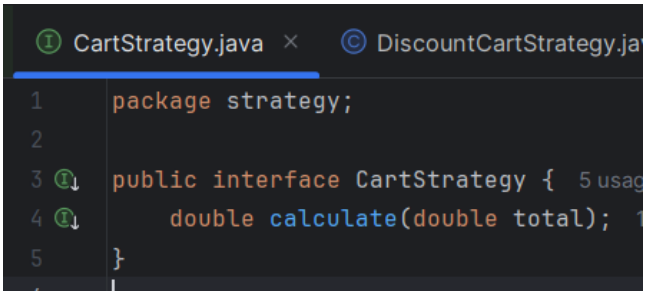
Repository-ja e projektit është e aksesueshme në link-un e mëposhtëm:

https://github.com/enimarsela/AlbArt

14. DESIGN PATTERNS

Në këtë projekt janë aplikuar disa Design Patterns të rëndësishme, të cilat përmirësojnë arkitekturën e sistemit, rrisin fleksibilitetin e kodit dhe mundësojnë mirëmbajtje më të lehtë në të ardhmen. Konkretisht janë përdorur modelet: Strategy, Singleton dhe Iterator. Në vijim paraqitet funksioni i secilit prej tyre, arsyeja e përdorimit dhe përfitimet që sjellin në aplikacion.

14.1. Strategy Pattern



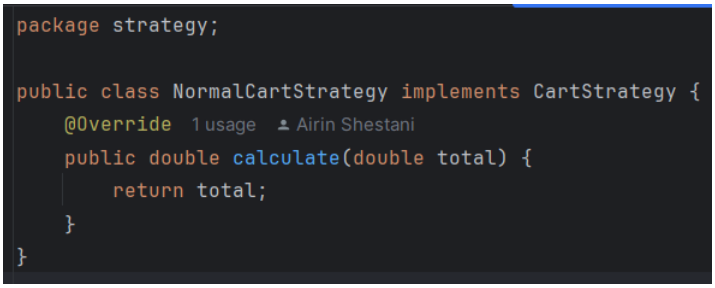
14.1.1. Arsyeja e përdorimit

Strategy Pattern përdoret për situata kur ekzistojnë disa mënyra të ndryshme për të kryer të njëjtin funksion, dhe është e nevojshme që këto mënyra të mund të ndërrohen në mënyrë dinamike pa ndryshuar strukturën e kodit kryesor. Ky pattern ndihmon në shmangien e strukturave të ndërlikuara if-else dhe e bën kodin më të zgjerueshëm.

14.1.2. Shembuj konkretë nga kodi

Në këtë projekt, ky pattern është përdorur për llogaritjen e totalit të Shopping Cart, ku ekzistojnë disa strategji të ndryshme kalkulimi, si p.sh.:

- Strategjia normale pa zbritje



- Strategjia me zbritje në përqindje

```java
package strategy;

public class DiscountCartStrategy implements CartStrategy {

    private double discount;  2 usages

    public DiscountCartStrategy(double discount) {  1 usage
        this.discount = discount;
    }

    @Override  1 usage  Airin Shestani
    public double calculate(double total) {
        return total - (total * discount);
    }
}
```

- Strategjia VIP me zbritje të veçantë

```java
package strategy;

public class VipCartStrategy implements CartStrategy {  2

    @Override  1 usage  Airin Shestani
    public double calculate(double total) {
        return total * 0.80; // Ulja e kartes VIP: 20%
    }
}
```

Këto strategji implementojnë të njëjtin interface CartStrategy, ndërsa ShoppingCart zgjedh strategjinë e duhur në kohë ekzekutimi.

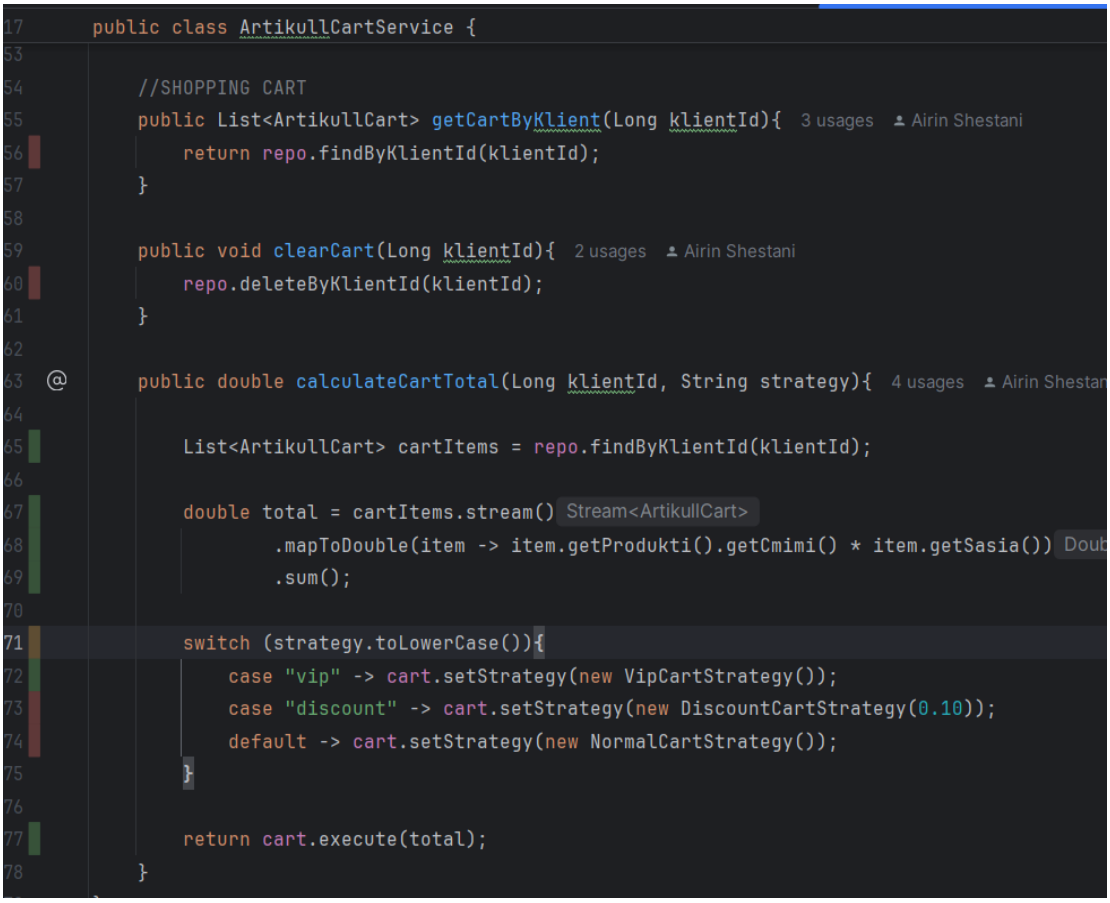
Sic ilustrohet dhe ne figurë, strategjitë përdoren për të llogaritur totalin e Shopping Cart të një klienti të caktuar. Kjo bëhët duke thirrur klasën ShoppingCart në servisin e artikujve ArtikullCartService. Metoda calculateCartTotal zgjedh strategjinë dhe në bazë të saj llogarit totalin.

```java
package strategy;

public class ShoppingCart {  3 usages  Airin Shestani

    private CartStrategy strategy;  2 usages

    public void setStrategy(CartStrategy strategy){
        this.strategy = strategy;
    }

    public double execute(double total){  1 usage
        return strategy.calculate(total);
    }
}
```

```java
@Service  6 usages  Airin Shestani +1
public class ArtikullCartService {

    private final ArtikullCartRepository repo;  9 usages
    private final ProduktiRepository produktiRepository;  1 usage
    private final ShoppingCart cart = new ShoppingCart();  4 usag
```

Përfitimet kryesore janë:

- Rritje fleksibiliteti
- Kod i lehtë për t'u zgjeruar
- Eliminim i kushteve të panevojshme "if–else"



```
public class ArtikullCartService {

    //SHOPPING CART
    public List<ArtikullCart> getCartByKlient(Long klientId){  3 usages  Airin Shestani
        return repo.findByKlientId(klientId);
    }

    public void clearCart(Long klientId){  2 usages  Airin Shestani
        repo.deleteByKlientId(klientId);
    }

    public double calculateCartTotal(Long klientId, String strategy){  4 usages  Airin Shestan

        List<ArtikullCart> cartItems = repo.findByKlientId(klientId);

        double total = cartItems.stream()  Stream<ArtikullCart>
                .mapToDouble(item -> item.getProdukti().getCmimi() * item.getSasia())  Doub
                .sum();

        switch (strategy.toLowerCase()){
            case "vip" -> cart.setStrategy(new VipCartStrategy());
            case "discount" -> cart.setStrategy(new DiscountCartStrategy(0.10));
            default -> cart.setStrategy(new NormalCartStrategy());
        }

        return cart.execute(total);
    }
```

14.2. Singleton Pattern

Singleton Pattern siguron që nga një klasë të ekzistojë vetëm një instancë gjatë gjithë funksionimit të aplikacionit.
Në Spring Boot, ky pattern është i implementuar natyrshëm përmes mekanizmit të Application Context, ku çdo **@Service**, **@Repository**, **@Controller** dhe **@Component** krijohet vetëm një herë dhe ripërdoret sa herë që kërkohet.

Kjo sjell:

- Menaxhim efikas të burimeve
- Kursim memorjeje
- Performancë më të mirë
- Centralizim të logjikës së biznesit

14.3. Iterator Pattern

Iterator Pattern përdoret për të kaluar nëpër koleksione të dhënash pa ekspozuar strukturën e tyre të brendshme.

Në projekt, ai përdoret automatikisht nga:

- Koleksionet Java (List)
- Rezultatet nga JPA Repository (findAll(), etj.)
- Streams API

Ky model ofron:

- Traversim më të sigurt të të dhënave
- Abstraksion nga mënyra e ruajtjes
- Kod më të thjeshtë dhe më të lexueshëm

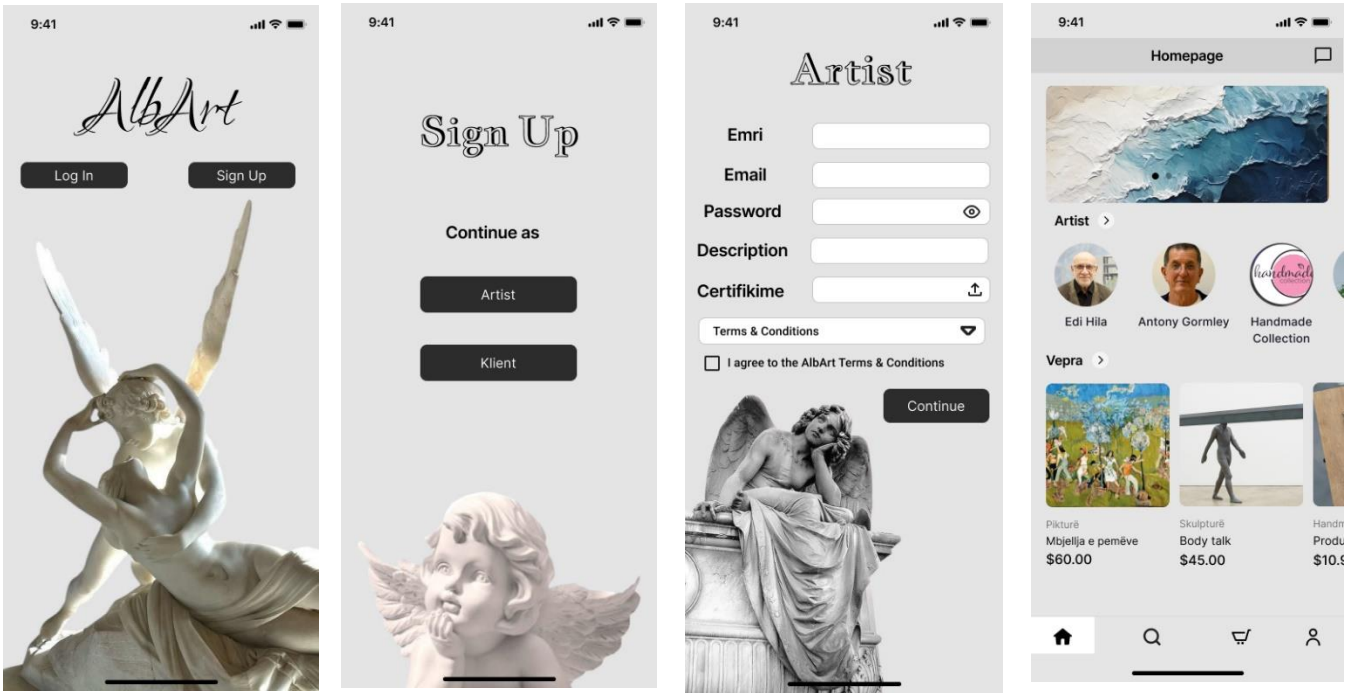
14.4. MVC Pattern

Gjithashtu programi në vetvete përmban pattern-in MVC (Model-View-Controller), në bazë të të cilit është krijuar një strukturë stabël për aplikacionin Spring Boot.

Përdorimi i këtyre Design Patterns në aplikacionin Spring Boot ndihmon në krijimin e një sistemi më profesional dhe më të mirëorganizuar.

15.MOCKUPS

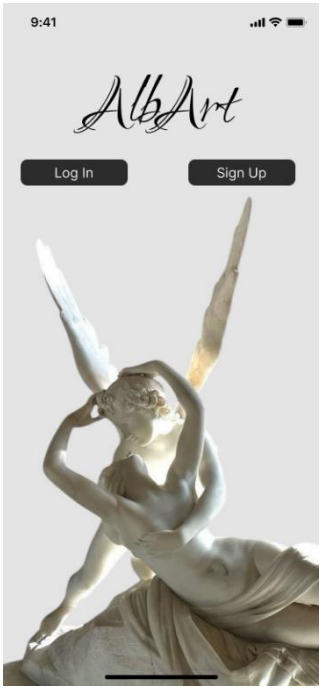
15.1. Use Case 1 : Regjistrimi i një artisti

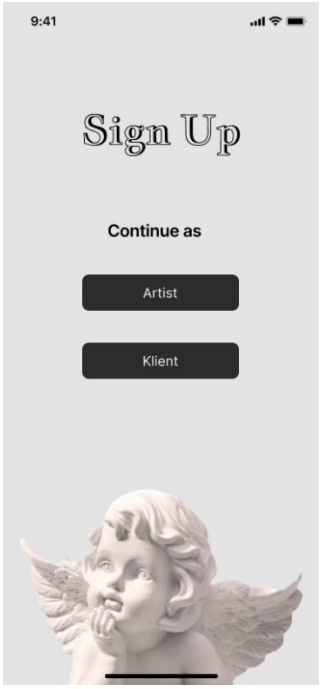


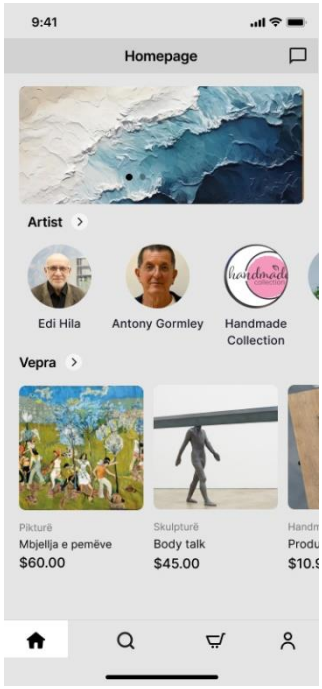
Ky mockup paraqet ndërfaqet e regjistrimit të artistëve në platformë, e cila u mundëson atyre të krijojnë një llogari të re në sistem. Ndërfaqja synon të sigurojë një proces të thjeshtë dhe të kuptueshëm regjistrimi.

- Përdoruesi hap menunë kryesore të sistemit.
- Sistemi shfaq formularin e regjistrimit ku përdoruesi duhet të përzgjedhë në qoftë se është artist apo klient (në këtë rast artist).
- Artisti plotëson të dhënat personale të kërkuara.
- Artisti pranon termat dhe kushtet e përdorimit.
- Sistemi verifikon të dhënat dhe krijon llogarinë e përdoruesit.
- Pasi klikon butonin Continue artistit i shfaqet Homepage me përmbajtjen e saj.

15.2. Use Case 2 : Regjistrimi i një klienti





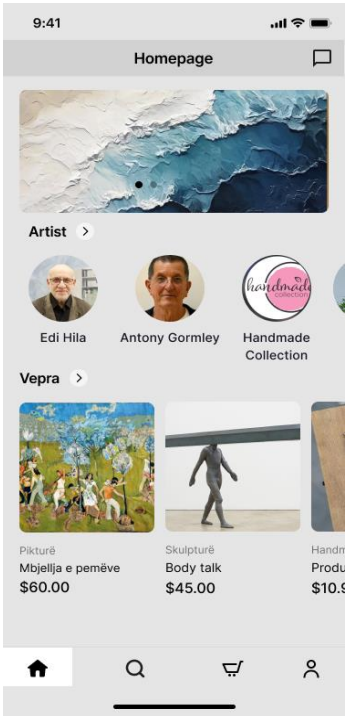


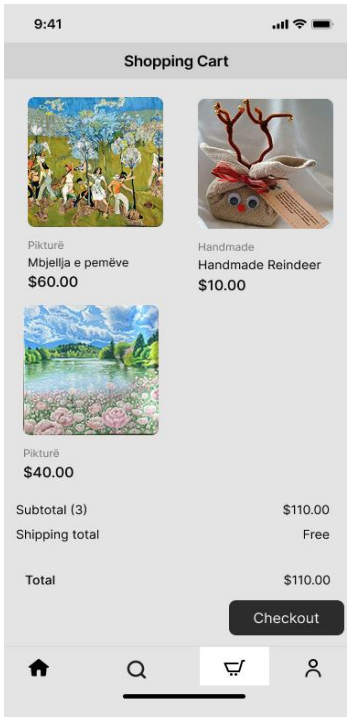
Ky mockup paraqet ndërfaqet e regjistrimit të klientëve në platformë, e cila u mundëson atyre të krijojnë një llogari të re në sistem. Ndërfaqja synon të sigurojë një proces të thjeshtë dhe të kuptueshëm regjistrimi.

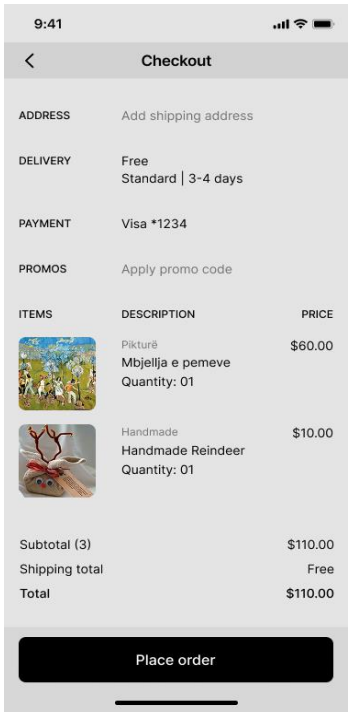
- Përdoruesi hap menunë kryesore të sistemit.
- Sistemi shfaq formularin e regjistrimit ku përdoruesi duhet të përzgjedhë në qoftë se është artist apo klient (në këtë rast klient).
- Klienti plotëson të dhënat personale të kërkuara.
- Klienti pranon termat dhe kushtet e përdorimit.
- Klienti përzgjedh preferencat e tij në lidhje me artin në mënyrë që ne Homepage t'i shfaqen produktet të cilat i interesojnë.

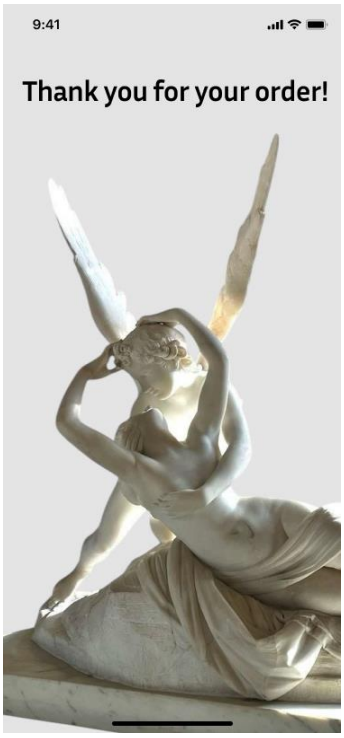
- Sistemi verifikon të dhënat dhe krijon llogarinë e përdoruesit.
- Pasi klikon butonin Continue klientit i shfaqet Homepage me përmbajtjen e saj.

15.3. Use Case 3 : Blerja e një produkti nga klienti







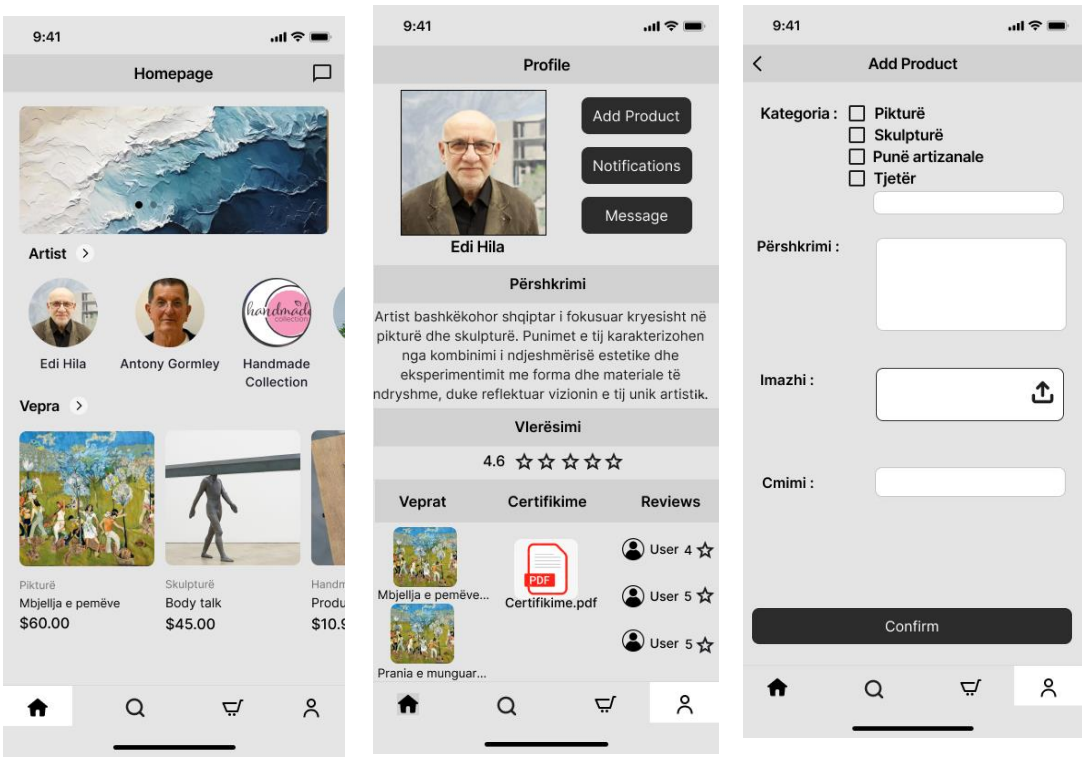


Ky mockup paraqet ndëfaqet e detajeve të produktit, ku shfaqet informacioni i plotë mbi veprën dhe mundësia për ta shtuar atë në shportën e blerjeve. Ndërfaqja mbështet vendimmarrjen e përdoruesit për blerje.

- Përdoruesi klikon mbi një produkt për të cilin interesohet në Homepage .
- Sistemi shfaq informacionin e plotë të produktit bashkë me çmimin dhe informacionin e artistit.
- Përdoruesi zgjedh opsionin “Add to Cart”.
- Sistemi përditëson përmbajtjen e shportës.
- Në momentin që përdoruesi do të mbyllë porosinë shkon te Shopping Cart dhe klikon Checkout.
- Në faqen e Checkout-it do i shfaqen të gjitha detajet e produkteve që do të porosisë si dhe hapësirat për të plotësuar detajet personale.

- Për ta mbyllur porosinë përdoruesi duhet të klikojë butonin Place Order dhe pastaj i shfaqet ndërfaqja përfundimtare ku e falenderon për porosinë.

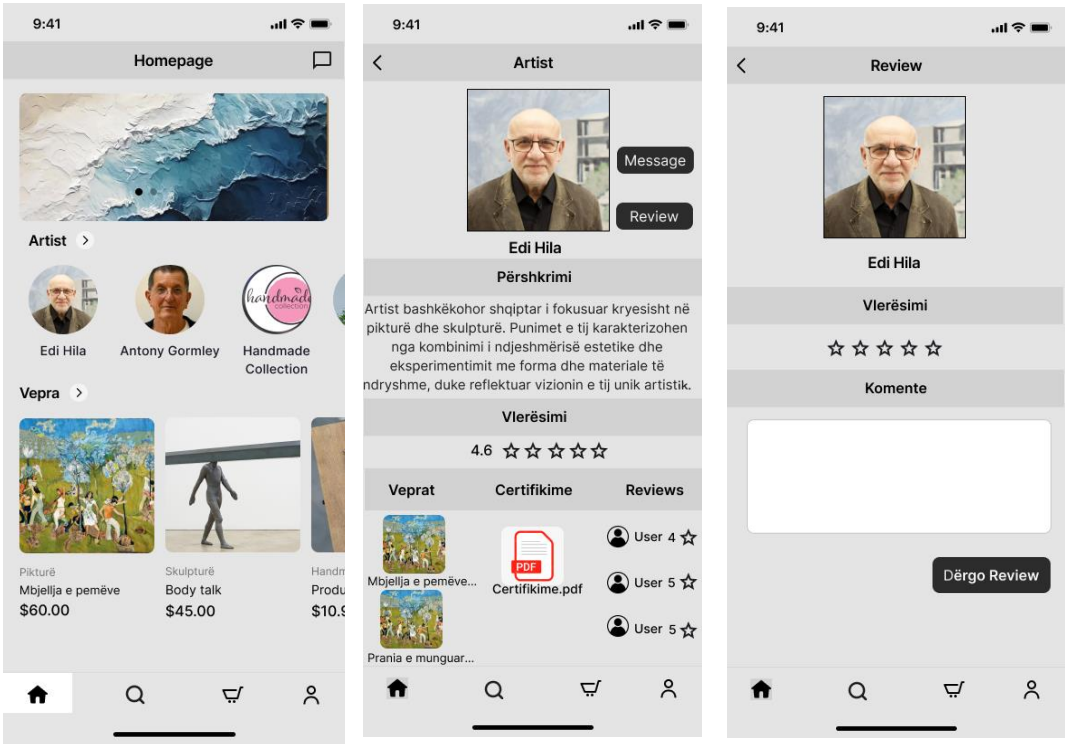
15.4.Use Case 4 : Shtimi i një vepre arti nga artisti



Ky mockup ilustron ndërfaqet për ngarkimin dhe publikimin e produkteve nga artisti në platformë. Ndërfaqja i mundëson artistit të menaxhojë përmbajtjen e tij dhe të prezantojë veprat për klientët.

- Artisti hyn në profilin e tij personal.
- Sistemi shfaq opsionin për ngarkimin e një produkti të ri.
- Artisti plotëson informacionin e kërkuar për produktin.
- Artisti ngarkon imazhin e produktit.
- Sistemi kryen validimin e të dhënave dhe aplikon watermark-un digjital.
- Produkti publikohet në profilin e artistit.

15.5.Use Case 5 : Një klient jep review për një artist



Ky mockup paraqet ndërfaqet për dhënien e vlerësimit nga klienti për një artist të caktuar. Ndërfaqja i mundëson klientit të shprehë mendimin e tij mbi produktin ose artistin, duke kontribuar në përmirësimin e transparencës dhe besueshmërisë së platformës.

- Klienti hyn në profilin personal të artistit.
- Sistemi shfaq opsionin për dhënien e review-t.
- Klienti jep vlerësimin (rating) dhe komentin përkatës.
- Klienti konfirmon dërgimin e review-t.
- Sistemi ruan vlerësimin dhe e shfaq atë në profilin e produktit ose artistit.

16. UNIT TESTING & CODE COVERAGE

16.1. Strategjia e Testimit të Aplikacionit

Procesi i testimit përbën një komponent thelbësor në ciklin e zhvillimit të aplikacioneve softuerike, pasi ndihmon në verifikimin e korrektësisë funksionale, qëndrueshmërisë, besueshmërisë dhe performancës së sistemit. Për aplikacionin në fjalë është zbatuar një strategji e strukturuar testimi, e projektuar për të mbuluar në mënyrë sa më gjithëpërfshirëse komponentët kryesorë të sistemit dhe logjikën e biznesit.

16.2. Qëllimet e Testimit

Strategjia e testimit është orientuar drejt arritjes së këtyre objektivave kryesore:

- Verifikimi i korrektësisë së funksionaliteteve kryesore të sistemit;
- Sigurimi i funksionimit të duhur të operacioneve CRUD;
- Validimi i sjelljes së sistemit në raste të zakonshme si dhe në skenarë me gabime;
- Kontrollimi i përgjigjeve të sakta HTTP nga kontrollorët e aplikacionit;
- Rritja e stabilitetit dhe besueshmërisë së aplikacionit para vendosjes në përdorim.

16.3. Llojet e Testimit të Zbatuara

16.3.1. Unit Testing

Unit test-et janë përdorur për të testuar në mënyrë të pavarur komponentët individualë të sistemit, veçanërisht shtresën e shërbimeve (Service Layer). Përmes këtyre testeve është vlerësuar korrektësia e logjikës së biznesit, funksionimi i metodave individuale si dhe trajtimi i rasteve te gabuara.

Teknologjitë e përdorura:

- JUnit 5
- Mockito për simulimin (mocking) e dependencies dhe repository-ve

Këto teste mbulojnë:

- Funksionalitete të krijimit, përditësimit dhe fshirjes së entiteteve;
- Proceset e autentifikimit dhe regjistrimit të përdoruesve;
- Verifikimin e kodimit të fjalëkalimeve dhe kontrollin e duplikimit të email-eve;
- Trajtimin e rasteve kur të dhënat nuk ekzistojnë në sistem.

16.3.2. Error Handling (Testimi i Trajtimit të Gabimeve)

Një aspekt i rëndësishëm i strategjisë së testimit ka qenë vlerësimi i sjelljes së sistemit në kushtet e gabimeve funksionale. Janë testuar skenarë të tillë si:

- kërkimi i entiteteve që nuk ekzistojnë,
- tentativat për regjistrim me të dhëna të përsëritura (p.sh., email ekzistues),
- dhënia e kredencialeve të pasakta gjatë autentifikimit,
- dërgimi i të dhënave të pavlefshme.

Kjo siguron që sistemi të kthejë mesazhe të sakta gabimi me kode të përshtatshme HTTP.

16.4. Përfitimet e Strategjisë së Testimit

Zbatimi i kësaj strategjie testimi ka sjellë disa përfitime të rëndësishme: rritje të cilësisë dhe qëndrueshmërisë së kodit, reduktim të gabimeve në fazat e mëvonshme të zhvillimit si dhe lehtësim të mirëmbajtjes së aplikacionit në të ardhmen.

16.5. Shembuj Unit Tests nga kodi

```java
@ExtendWith(MockitoExtension.class)  Airin Shestani
class ArtistiServiceTest {

    @Mock  6 usages
    private ArtistiRepository repo;

    @InjectMocks  5 usages
    private ArtistiService service;

    @Mock  2 usages
    private BCryptPasswordEncoder encoder;

    @Test  Airin Shestani
    void registerArtist_success() {
        ArtistiDto dto = new ArtistiDto();
        User user = new User();
        user.setEmail("a@test.com");
        user.setPassword("123");
        dto.setUser(user);

        when(repo.existsByEmail(user.getEmail())).thenReturn( t: false);
        when(repo.save(any())).thenReturn(new Artisti());

        Artisti result = service.registerArtist(dto);
        assertNotNull(result);
    }
```

```java
class ArtistiServiceTest {
    @Test  Airin Shestani
    void login_success() {
        UserDto login = new UserDto( emri: "a@test.com",  email: "123",  password: "password",  roli: "ARTIST");

        Artisti artist = new Artisti();
        artist.setUser(new User());
        artist.getUser().setPassword("encoded");

        when(repo.findByEmail(login.getEmail())).thenReturn(Optional.of(artist));
        when(encoder.matches(login.getPassword(),  encodedPassword: "encoded")).thenReturn( t: true);

        assertNotNull(service.login(login));
    }

    @Test  Airin Shestani
    void login_badPassword_throws() {
        UserDto login = new UserDto( emri: "a@test.com",  email: "123",  password: "password",  roli: "ARTIST");

        Artisti artist = new Artisti();
        artist.setUser(new User());
        artist.getUser().setPassword("encoded");

        when(repo.findByEmail(login.getEmail())).thenReturn(Optional.of(artist));
        when(encoder.matches(login.getPassword(),  encodedPassword: "encoded")).thenReturn( t: false);

        assertThrows(RuntimeException.class, () -> service.login(login));
    }
```

```java
class ProduktiServiceTest {

    @Test  Airin Shestani
    void createProdukt_ShouldSaveProdukt() {
        when(produktiRepository.save(any(Produkti.class))).thenReturn(produkti);

        Produkti saved = produktiService.createProdukt(produktiDto);

        assertNotNull(saved);
        assertEquals( expected: "Produkt Test", saved.getEmri());
        verify(produktiRepository, times( wantedNumberOfInvocations: 1)).save(any(Produkti.class));
    }

    @Test  Airin Shestani
    void getProduktiById_ShouldReturnProdukt() {
        when(produktiRepository.findById(1L)).thenReturn(Optional.of(produkti));

        Produkti result = produktiService.getProduktiById( produktId: 1L);

        assertNotNull(result);
        assertEquals( expected: 1L, result.getProduktId());
        verify(produktiRepository).findById(1L);
    }
```

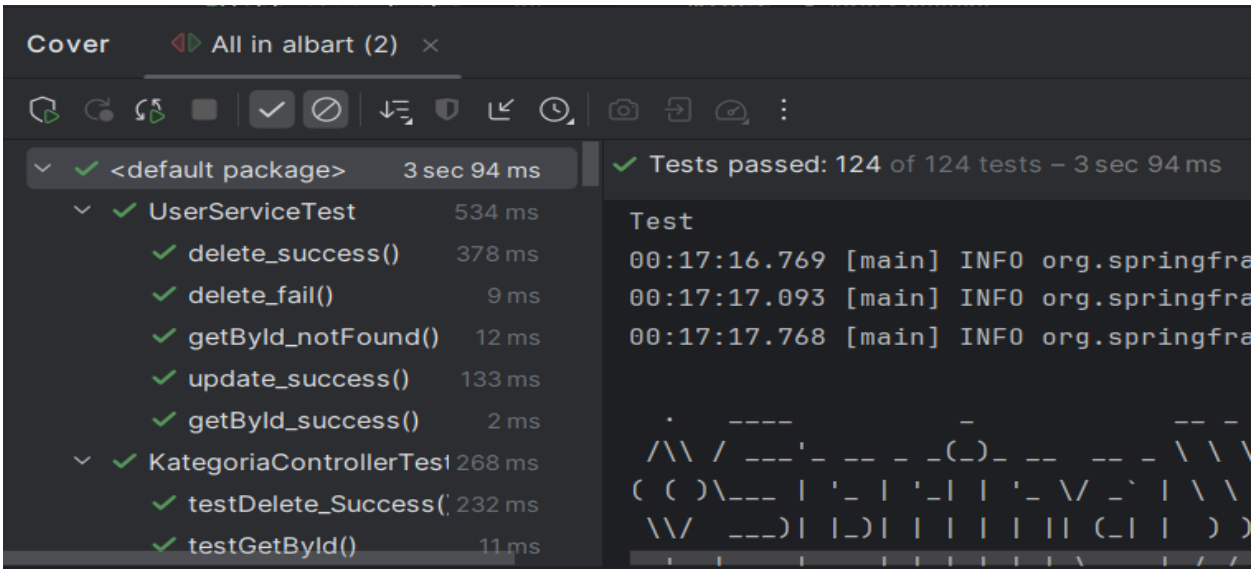
```java
class ArtikullCartServiceTest {
    @Test  Airin Shestani
    void getById_notFound_throws() {
        when(repo.findById(1L)).thenReturn( t: Optional.empty());
        assertThrows(EntityNotFoundException.class, () -> service.getById(1L));
    }

    @Test  Airin Shestani
    void delete_notExists_false() {
        when(repo.existsById(1L)).thenReturn( t: false);
        assertFalse(service.delete( id: 1L));
    }

    @Test  Airin Shestani
    void calculateCartTotal_vip() {
        Produkti p = new Produkti();
        p.setCmimi(100);

        ArtikullCart a = new ArtikullCart();
        a.setProdukti(p);
        a.setSasia(1);

        when(repo.findByKlientId(1L)).thenReturn(List.of(a));

        double total = service.calculateCartTotal( klientId: 1L,  strategy: "vip");
        assertTrue( condition: total > 0);
    }
}
```

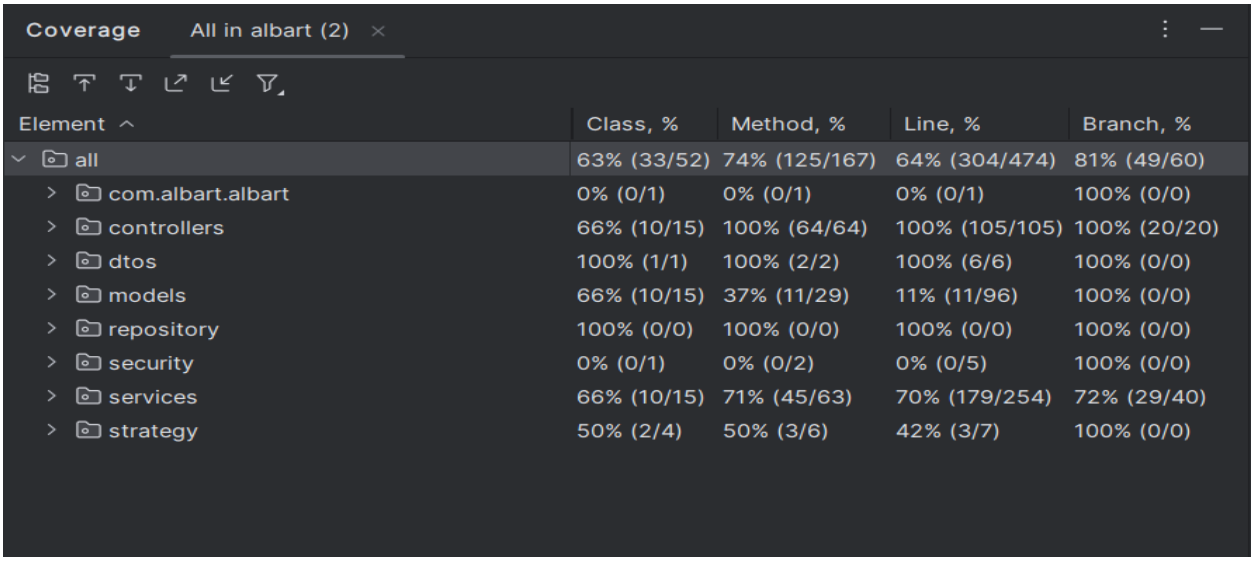
```java
class ArtikullCartControllerTest {  Airin Shestani

    @Test  Airin Shestani
    void testClearCart() {
        ResponseEntity<String> response = controller.clearCart( klientId: 1L);

        assertEquals(HttpStatus.OK, response.getStatusCode());
        assertEquals( expected: "Shopping Cart cleared!", response.getBody());
        verify(service).clearCart( klientId: 1L);
    }

    @Test  Airin Shestani
    void testCalculateTotal() {
        when(service.calculateCartTotal( klientId: 1L,  strategy: "normal")).thenReturn( t: 100.0);

        ResponseEntity<Double> response = controller.calculateTotal( klientId: 1L,  strategy: "normal");

        assertEquals(HttpStatus.OK, response.getStatusCode());
        assertEquals( expected: 100.0, response.getBody());
        verify(service).calculateCartTotal( klientId: 1L,  strategy: "normal");
    }
}
```

```java
class KlientControllerTest {  Airin Shestani
    @Mock  7 usages
    private KlientService klientService;

    @InjectMocks  7 usages
    private KlientController controller;

    private Klient klient;  11 usages
    private KlientDto dto;  5 usages
    private UserDto userDto;  3 usages

    @BeforeEach  Airin Shestani
    void setUp() {
        MockitoAnnotations.openMocks( testClass: this);
        klient = new Klient();
        klient.setKlientId(1L);
        dto = new KlientDto();
        userDto = new UserDto();
    }

    @Test  Airin Shestani
    void testRegister() {
        when(klientService.register(dto)).thenReturn(klient);
        ResponseEntity<Klient> response = controller.register(dto);
        assertEquals(HttpStatus.OK, response.getStatusCode());
        assertEquals(klient, response.getBody());
    }
```

```java
@Test  Airin Shestani
void testUpdate() {
    when(service.update( id: 1L, dto)).thenReturn(porosi);

    ResponseEntity<Porosi> response = controller.update( id: 1L, dto);

    assertEquals(HttpStatus.OK, response.getStatusCode());
    assertEquals(porosi, response.getBody());
    verify(service, times( wantedNumberOfInvocations: 1)).update( id: 1L, dto);
}
```

16.6. Rezultatet e testeve dhe Code Coverage





Element	Class, %	Method, %	Line, %	Branch, %
all	63% (33/52)	74% (125/167)	64% (304/474)	81% (49/60)
com.albart.albart	0% (0/1)	0% (0/1)	0% (0/1)	100% (0/0)
controllers	66% (10/15)	100% (64/64)	100% (105/105)	100% (20/20)
dtos	100% (1/1)	100% (2/2)	100% (6/6)	100% (0/0)
models	66% (10/15)	37% (11/29)	11% (11/96)	100% (0/0)
repository	100% (0/0)	100% (0/0)	100% (0/0)	100% (0/0)
security	0% (0/1)	0% (0/2)	0% (0/5)	100% (0/0)
services	66% (10/15)	71% (45/63)	70% (179/254)	72% (29/40)
strategy	50% (2/4)	50% (3/6)	42% (3/7)	100% (0/0)

17.KONKLUZIONE

17.1. Përmbledhje

Platforma AlbArt është zhvilluar si një zgjidhje digjitale për menaxhimin dhe ndërmjetësimin e veprave artistike në një mjedis online. Ajo synon të krijojë një hapësirë të centralizuar ku artistët mund të prezantojnë dhe të shesin krijimet e tyre, ndërsa klientët mund të eksplorojnë, të blejnë dhe të vlerësojnë veprat në mënyrë të sigurt dhe të strukturuar.

Zhvillimi i platformës përfshiu identifikimin e kërkesave kryesore të sistemit, projektimin e arkitekturës logjike dhe fizike, modelimin e proceseve përmes diagramave standarde dhe ndërtimin e një strukture databaze që mbështet funksionalitetet kryesore. Funksione të tilla si menaxhimi i profileve, publikimi i veprave, pagesat elektronike, ndjekja e porosive dhe komunikimi midis përdoruesve janë integruar për të ofruar një rrjedhë të plotë funksionale.

Në tërësi, platforma përfaqëson një zbatim praktik të parimeve të inxhinierisë së softuerit në ndërtimin e një sistemi të mirëorganizuar, të zgjerueshëm dhe të përshtatshëm për përdorim në mjedise reale.

17.2.Probleme të hasura

Gjatë zhvillimit të platformës u identifikuan disa sfida teknike dhe organizative që kërkuan rishikim dhe përshtatje të vazhdueshme të zgjidhjeve të propozuara.

Një nga vështirësitë kryesore ishte përkthimi i nevojave të përdoruesve në kërkesa funksionale të qarta dhe të matshme, veçanërisht në fazën fillestare të analizës së kërkesave, ku disa funksione u riformuluan për të shmangur paqartësitë dhe interpretimet e ndryshme.

Një tjetër problem lidhej me modelimin korrekt të marrëdhënieve në databazë, të cilat kërkonin përdorimin e entiteteve ndërmjetëse për të ruajtur integritetin e të dhënave.

Integrimi i moduleve të ndryshme të sistemit, si menaxhimi i përdoruesve, veprave artistike, porosive dhe pagesave, kërkoi koordinim të kujdesshëm për të siguruar që rrjedha e të dhënave të ishte e saktë dhe e qëndrueshme në të gjithë sistemin.

Gjithashtu, projektimi i ndërfaqes së përdoruesit kërkoi disa iterime për të balancuar kompleksitetin funksional me thjeshtësinë dhe lehtësinë e përdorimit për përdoruesin përfundimtar.

17.3.Përmirësime të ardhshme

Në të ardhmen, platforma mund të zgjerohet me funksionalitete më të avancuara që synojnë përmirësimin e performancës, sigurisë dhe përvojës së përdoruesve. Integrimi i inteligjencës artificiale paraqet një mundësi të rëndësishme zhvillimi, duke mundësuar rekomandime më të personalizuara të veprave artistike, analiza inteligjente të preferencave të përdoruesve, kategorizim automatik të përmbajtjes dhe optimizim të proceseve të brendshme të sistemit.

Gjithashtu, platforma mund të adoptojë arkitekturë më të shkallëzueshme për të mbështetur një numër më të madh përdoruesish dhe transaksionesh, të integrohet me më shumë platforma pagesash dhe shërbimesh transporti, si dhe të ofrojë mbështetje për shumë gjuhë dhe valuta. Forcimi i mekanizmave të sigurisë dhe shtimi i moduleve analitike për artistët dhe administratorët do të kontribuonin në një menaxhim më efikas dhe një eksperiencë më të plotë përdorimi.

Këto përmirësime e bëjnë platformën të gatshme për zhvillime të mëtejshme dhe për përdorim në një mjedis dinamik dhe real.

18.SHTOJCA

Tabela

Diagrama

19.INDEKSI