September, 2025 Volume 2, Issue 3

№ 007

Management and Future Technologies

SCIENTIFIC JOURNAL

journal.umft.uz
ISSN 3060-5008

Jurnal sohalari: menejment, dasturiy injiniring, sun’iy intellekt texnologiyalari, kompyuter injiniringi, infokommunikatsiya injiniringi, axborot xavfsizligi, raqamli iqtisodiyot, pedagogika va psixologiya, matematika va fizika

Jurnal O‘zbekiston Respublikasi Prezidenti Administratsiyasi huzuridagi Axborot va ommaviy kommunikatsiyalar agentligidan 212842-raqamli guvohnoma bilan 26.01.2024 sanasida ro‘yxatdan o‘tkazilgan.

MUNDARIJA

TEXNIKA FANLARI

Musayev Muhammadjon; Jurayev Dilshod
PROGRESSIV TRANSFORMER ALGORITMLARI ASOSIDA UZLUKSIZ IMO-ISHORA TILI HARAKAT NUQTALARINI HOSIL QILISH 6

Khamdamov Utkir, Khalilov Sirojiddin, Omonullayeva Faragiza, Akhmedov Farrukh, Umarov Mukhriddin
DEEP LEARNING MODEL FOR HUMAN FACE ANIMITY DETECTION IN A DISTANCE LEARNING PLATFORM 23

Rahmatullayev Ilhom; Abduraximov Baxtiyor
NEWHSA OQIMLI SHIFRLASH ALGORITMINING APPARAT DARAJADAGI SAMARALI TADBIQI VA ARXITEKTURAVIY TAHLILI 34

Turgunov Bekzod
AVTONOM HARAKATLANISH TIZIMLARIDA OBYEKTGACHA MASOFANI O‘LCHASH ANIQLIGINI OSHIRISH YECHIMINI ISHLAB CHIQISH 47

Shukurov Kamolidddin; Ochilov Mannon; Xasanov Umidjon
NUTQ SIGNALLARIGA INTELLEKTUAL ISHLOV BERISH TIZIMLARIDA SO‘ZLOVCHILARNI AJRATISHNING AHAMIYATI 58

Botir Elov; Abdulla Abdullayev
O‘ZBEK MATNLARI SENTIMENT TAHLIL TIZIMI: ARXITEKTURA VA LOYIHALASH 67

Aybek Xaytbayev
5G TARMOQLARIDA CHEGARAVIY HISOBLASH RESURSLARINI BOSHQARISHGA MO‘LJALLANGAN AQLLI TIZIM 88

Abduraximov Baxtiyor; Allanov Orif; Turdibekov Baxtiyor
BLOKCHAIN TEXNOLOGIYASIDA FOYDALANILADIGAN KRIPTOGRAFIK XESH FUNKSIYALAR TAHLILI 102

Maksud Sharipov; Ogabek Sobirov
DEVELOPMENT OF A LEMMATIZATION ALGORITHM: INTERPRETING OPEN COMPOUND WORDS 110

Saidkulov Elyor; Ibrohimova Zulayho
DOIRANI GEOMETRIK SHAKILLAR BILAN TO‘LDIRISH YORDAMIDA OROL DENGIZINING FRAKTAL O‘LCHOVINI ANIQLASH 117

Элмурод Уринов; Шохруз Тургуналиев
СОСТОЯНИЕ БЕЗОПАСНОСТИ УМНЫХ ДОМАШНИХ УСТРОЙСТВ: КОМПЛЕКСНЫЙ АНАЛИЗ 128

Saidkulov Elyor
FRAKTAL TUZILISHLI YER SIRTINI GEOMETRIK MODELLASHTIRISH VA VIZUALLASHTIRISH 137

Musaev Anvar
KIBERXAVFSIZLIK VA AXBOROT XAVFSIZLIGINING O'ZIGA HOS XUSUSIYATLARI 145

Umarov Muhriddin; Melibayeva Xilola; Xudoyberdiyev Shaxriyor; Giyasova Gulnoza; Samarov Sherzod
GRAFIK PROTSESSORLARDA TASVIRLARDAN YO'L BELGILARINI TANIB OLUVCHI DASTURIY VOSITALARINING FUNKSIONAL STRUKTURASI 150

Abdulxayev Nodir
UNITY PLATFORMASIDA YARATILGAN VIRTUAL REALLIK SIMULYATSIYASIDA INSON HARAKATINI REAL VAQTDA HIS QILISH ALGORITMINI ISHLAB CHIQISH 161

Botir Elov; Mastura Primova
O'ZBEK MATNLARI UCHUN N-GRAM TIL MODELI 166

Xalilov Sirojiddin
SUN'IY INTELLEKT USULLARI YORDAMIDA VIDEOTASVIRDAN SHAXSNING YUZ TASVIRI HAMDA OBEYKTLARNI ANIQLASH ALGORITMLARI 177

Jumaboyeva Pokiza
SUN'IY INTELLEKT ASOSIDAGI O'ZBEK IMO-ISHORA TILINI TANIB OLISH VA TABIIY TILGA INTEGRATSIYA QILISH 188

Xudoyberdiyev Rahmatillo
SUN'IY YO'LDOSH ALOQA KANALLARI MODELLASHTIRISH USULLARI VA SIMULYATSIYA TEXNOLOGIYALARI 194

Xolmatov Orzumurod
TABIIY TILNI QAYTA ISHLASHGA ASOSLANGAN SAVOLLARGA JAVOB BERISH YONDASHUVLARI VA ALGORITMLARI TAHLILI 202

Axmedov Farrux
O'ZBEK TILIDA MATNDAN NUTQ SINTEZLASH UCHUN MODEL TANLOVI VA UNING O'QITILISHI 212

Fotima Alimova
INTEGRATING BPMN, DMN, AND UML FOR COMPARATIVE ASSESSMENT OF PUBLIC AND COMMERCIAL DIGITAL SERVICES 222

Sotvoldiyeva Dildora; Abdullayeva Mohigul
MADANIY MEROSNI RAQAMLASHTIRISH: FOTOGRAMMETRIYA VA 3D SKANERLASHNING AHAMIYATI, MUAMMOLAR VA ISTIQBOLLAR 230

Akmuradov Baxtiyor; Berdimuradov Mirzohid
SERVERLARDA YUKLAMANI TAQSIMLASH ALGORITIMLARINI TADQIQ QILISH 241

Xudayberganov Temur; Davlatmuratov Valijon
BIOMETRIK AUTENTIFIKATSIYA JORIY ETISH ASOSIDA XAVFSIZLIK DARAJASINI OSHIRISH MODELI VA ALGORTIMLARI TAHLILI 252

Sharipov Maksud; Adinayev Xushnudbek
DEVELOPING A CONDITIONAL RANDOM FIELD (CRF) MODEL FOR DETECTING INTERROGATIVE SENTENCES IN UZBEK TEXTS 257

Amirkulov Marufjon
WORKING WITH PARALLEL CORPORA: APPROACHES AND TOOLS IN COMPUTATIONAL LINGUISTICS 263

Shohida Yusupova; Ogabek Matyoqubov
ADVANCING EDUCATIONAL METHODS THROUGH ICT INTEGRATION 269

Omonullayeva Farangiza
JURNALISTIK FAOLIYATNI MONITORING QILISH VA JURNALISTLAR REYTINGINI BAHOLASH JARAYONLARINING MODELLARI VA AXBOROT TIZIMINI DOLZABRLIGI 278

Bozorov Jasurbek
VIRTUAL TA'LIM TEXNOLOGIYASI YORDAMIDA O'QITISH METODIKASINI SHAKLLANTIRISH 282

Yarashbek Yusupov, Nargiza Baymatova
YUQORI POZITSIYALI FAZA MODULYATSIYA TURLARI UCHUN BIT XATOLIK EHTIMOLLIGINI BAHOLASH USULLARI 286

O‘ZBEK MATNLARI SENTIMENT TAHLIL TIZIMI: ARXITEKTURA VA LOYIHALASH

Botir Elov; Abdulla Abdullayev

Texnika fanlari falsafa doktori, dotsent. ToshDO’TAU

Annotatsiya. Ushbu maqolada o‘zbek tilidagi matnlarni avtomatik sentiment tahlil qilishga mo‘ljallangan axborot tizimining arxitekturasi va uning komponentlarini loyihalash bosqichlari tahlil qilinadi. Tizim uch qavatli model (frontend, backend, ma’lumotlar bazasi) asosida ishlab chiqilgan bo‘lib, Flask asosidagi REST API orqali foydalanuvchi interfeysi bilan integratsiyalashgan. Backend qismida transformer modellariga asoslangan chuqur o‘rganish algoritmlari (masalan, XLM-RoBERTa) orqali sentiment klassifikatsiyasi amalga oshiriladi. Ma’lumotlar bazasi sifatida SQLServer tanlangan bo‘lib, tizim docker konteyner orqali joylashtirilgan. Shuningdek, maqolada tizimning UML diagrammalari, modul tuzilmasi, ishlov berish oqimi va foydalanuvchi interfeysi dizayni yoritilgan. Ilmiy va amaliy natijalar asosida ishlab chiqilgan arxitektura real vaqtli tahlil uchun barqaror va masshtablanuvchan yechim sifatida baholanadi.

Kalit so‘zlar: sentiment tahlil, tizim arxitekturasi, REST API, Flask, React, SQLServer, Transformer modeli, SDLC, Docker, XLM-RoBERTa, UML.

Kirish

Hozirgi raqamli davrda matnlardagi fikr-mulohazalarni avtomatik ravishda tahlil qilish (sentiment tahlil) – axborot texnologiyalari va sun’iy intellekt sohalarining muhim yo‘nalishiga aylandi. Sentiment tahlil yordamida foydalanuvchilarning matnlarda ifodalagan kayfiyati va munosabatini (ijobiy, salbiy yoki betaraf) aniqlash imkoniyati paydo bo‘lib, bu turli sohalarda, ijtimoiy tarmoqlar tahlilidan tortib, mijozlar fikrlarini o‘rganish va hatto siyosiy kayfiyatni bashoratlashgacha keng qo‘llanmoqda. So‘nggi yillarda sun’iy intellekt va mashinali o‘qitish usullarining jadal rivojlanishi natijasida sentiment tahlil texnologiyasi yanada takomillashib, ko‘plab tillar uchun yuqori samaradorlikka erishilmoqda. Biroq, o‘zbek tilidagi matnlarni avtomatik baholash masalasi hali to‘liq yechimini topmagan va dolzarb muammo bo‘lib qolmoqda. O‘zbek tilining boy morfologik qurilishi, so‘z yasalishining affiksal tabiati va gap tuzilishidagi o‘ziga xosliklar sentiment tahlilni qiyinlashtiradi. Masalan, bir so‘zning turli grammatik shakllari va ma’nodagi o‘zgarishlari uning hissiy bo‘yog‘ini aniqlashni murakkablashtiradi. Natijada, jahon miqyosida keng qo‘llanilayotgan ba’zi metodlar o‘zbek tilida kutilgan darajada samarali natija bermasligi mumkin.

Shunga qaramay, o‘zbek tili uchun ham sentiment tahlil sohasida dastlabki ilmiy izlanishlar boshlangan. Xususan, ayrim tadqiqotlarda ijtimoiy tarmoqlar va sharhlar korpusi yig‘ilib, mashinali o‘qitish modellari yordamida yuqori aniqlikka erishilganligi xabar berilgan [1]. Bu yutuqlar o‘zbek tilida sentiment tahlil usullarini qo‘llash mumkinligini ko‘rsatadi. Biroq, mavjud ishlanmalar asosan algoritmik yechimlar va dastlabki prototiplar darajasida bo‘lib, to‘liq funksional axborot tizimi shaklida joriy etilmagan. Hali o‘zbek tilida sentiment tahlilini amalga oshiruvchi, foydalanuvchiga qulay interfeysga ega va yuqori samarali komponentlardan tashkil topgan integrallashgan tizimlar kam uchraydi.

Ushbu maqolada aynan o‘zbek matnlari uchun sentiment tahlil axborot tizimini ishlab chiqishga e’tibor qaratiladi. Tadqiqotning maqsadi – o‘zbek tilidagi matnlardan foydalanuvchi fikrlarini avtomatik tahlil qilib, ularning hissiy rangini aniqlovchi tizim arxitekturasini loyihalash va prototipini yaratishdir. Tizimni loyihalashda uning ko‘p moduldan iborat arxitekturasi, ma’lumotlar oqimini boshqaruvchi biznes jarayonlar modeli, dasturiy ta’minot kod tuzilmasi hamda foydalanuvchi uchun qulay interfeys elementlarini puxta o‘ylash talab etiladi. Maqolada aynan shu jihatlar tizimning umumiy arxitekturasi va qatlamlari, UML kabi modellashtirish vositasi yordamida loyihalash metodologiyasi, dasturiy yechimlar tanlovi, ishlab chiqish bosqichlari va yakuniy foydalanuvchi interfeysi ilmiy asosda yoritib beriladi. O‘zbekistonda bunday tizimni yaratish masalasi dolzarb bo‘lib, u milliy axborot resurslaridan samarali foydalanish, davlat va jamiyat ehtiyojlariga moslashtirilgan til texnologiyalarini rivojlantirishda muhim ahamiyat kasb etadi.

Adabiyotlar sharhi

Dunyo miqyosida sentiment tahlil tizimlarining arxitekturasi odatda ko‘p qatlamli va modular tuzilishga ega bo‘lib, u bosqichma-bosqich matnni qayta ishlash jarayonini aks ettiradi. Ilmiy manbalarda sentiment tahlil tizimlari odatda ma’lumotni kiritish, matnni oldindan qayta ishlash, klassifikatsiya (model) va natijalarni chiqish kabi asosiy komponentlardan tashkil topishi ta’kidlangan [2]. Xususan, Zamoto kabi restoran sharhlarida sentiment tahlil qilish bo‘yicha yaratilgan integrallashgan tizim misolida ikki asosiy qatlamli arxitektura taklif qilingan: birinchisi, matnlarni tahlil qiluvchi NLP pipeline (oldindan qayta ishlash – tokenizatsiya, tozalash, vektorlashtirish va klassifikatsiya); ikkinchisi esa foydalanuvchilar bilan o‘zaro aloqada bo‘ladigan veb-ilova qatlami. Bunday yondashuvda tahlil qilish moduli alohida komponent sifatida offline tarzda ishlaydi va o‘z natijalarini saqlaydi, veb-qism esa ushbu natijalarni foydalanuvchiga taqdim etadi va real vaqt rejimida yangi kiritilayotgan matnlarni ham yengil tahlil qilish imkonini beradi. Bunday modullik tamoyili tizimni kengaytirish va yangilashni yengillashtiradi va kelgusida matn hajmi oshsa yoki klassifikatsiya usullarini chuqur o‘rganishga (deep learning) almashtirish zarurati tug‘ilsa, arxitektura shunga mos holda o‘zgartirilishi mumkin.

O‘zbekistonda ham axborot tizimlarini loyihalashda shunga o‘xshash arxitektura tamoyillariga rioya etilgan ishlar mavjud. Masalan, o‘zbek tilida gaplar semantik tahliliga doir yaratilgan tizim Django veb-frameworki asosida **MVT (Model-View-Template)** arxitekturasida ishlab chiqilgan [3,4]. Tadqiqotchilar dastlab tizim uchun konseptual model va biznes jarayonlarini ishlab chiqqan holda, keyinchalik Django platformasi imkoniyatlaridan foydalangan holda, ma’lumotlar **modeli (Model), amalga oshirish qismi (View)** va **taqdimot qatlami (Template)**ni ajratgan holda dasturiy realizatsiyani amalga oshirishgan [5]. Natijada yaratilgan prototip tizim polifunksional so‘zlarni aniqlash masalasini hal qilishga mo‘ljallangan bo‘lsa-da, uning arxitekturaviy yechimi boshqa sentiment tahlil tizimlariga ham o‘xshash tarzda ko‘p qatlamli tuzilgan. Umuman, tajriba shuni ko‘rsatadiki, **uch qavatli arxitektura** (ma’lumotlar bazasi, biznes mantiq va prezentatsiya qatlamlari) yoki maxsus holatlarda ikki asosiy modulga bo‘lingan arxitektura (tahlil moduli + interfeys moduli) sentiment tahlil tizimlarida samarali qo‘llanishi mumkin [6]. Bu modulli yondashuv yangi funksiyalar qo‘shish, algoritmlarni takomillashtirish yoki tizimni kengaytirishda katta qulaylik yaratadi.

Keng ko‘lamli dasturiy tizimlarni ishlab chiqishda Unified Modeling Language (UML) va Business Process Model and Notation (BPMN) kabi standartlashtirilgan modellashtirish

vositalaridan foydalanish jahon tajribasida keng tarqalgan. UML dasturiy ta’minot arxitekturasi va komponentlari o‘rtasidagi munosabatlarni grafik shaklda tasvirlashga xizmat qilsa, BPMN biznes jarayonlarning oqimlarini tartibli ravishda aks ettiradi. Sentiment tahlil axborot tizimi singari murakkab tuzilishga ega loyihalarni bosqichma-bosqich loyihalashda ushbu vositalar katta yordam beradi [7]. Xususan, UML ning *use-case diagrammalari, klass diagrammalari* va *faoliyat diagrammalari* tizim qanday ishlashini, foydalanuvchi va dastur o‘rtasidagi aloqalarni va dasturiy modul va sinflarning tarkibini ko‘rsatib beradi. BPMN esa foydalanuvchi so‘rovini qayta ishlashdan boshlab natijani taqdim etishgacha bo‘lgan ish jarayonini ketma-ket bosqichlar sifatida ifodalaydi.

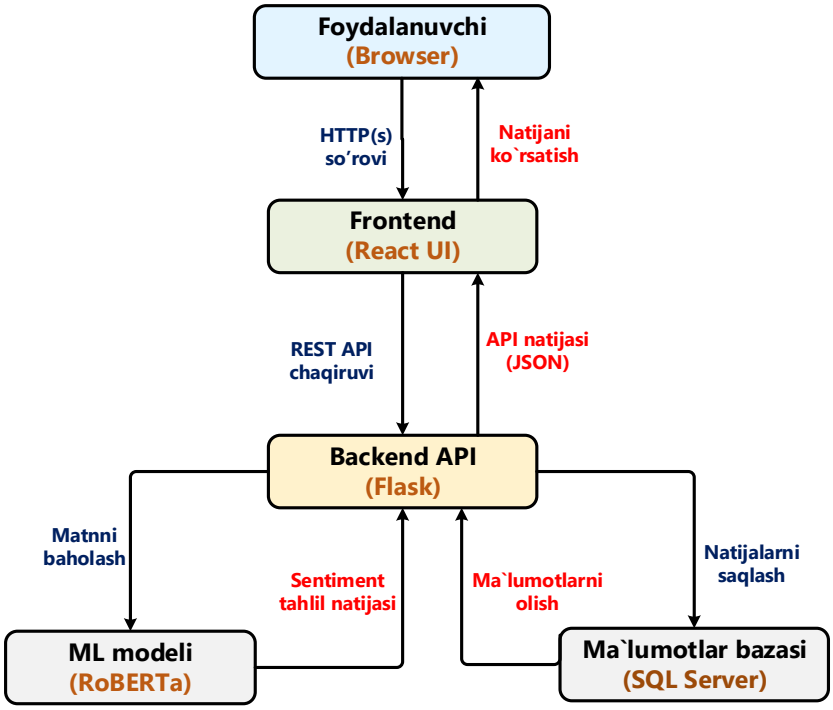
Ilmiy adabiyotlarda o‘zbek tilidagi matnlarni tahlil qiluvchi tizimlar loyihasida aynan BPMN modellashtirishidan foydalanilgan misollar ham uchraydi. Masalan, polifunksional so‘zlarni aniqlashga doir yaratilgan axborot tizimi loyihasida tadqiqotchilar tizimning har bir funksional jarayonini tahlil qilib, ularni BPMN notatsiyasida alohida biznes jarayon modellari shaklida ifodalashgan. Natijada, tizimda yuz beruvchi jarayonlar (masalan, matnni kiritish, uni qismlarga ajratish, so‘z turkumini aniqlash, natijani bazaga saqlash va hokazo) alohida sxemalar ko‘rinishida tasvirlanib, ularni oson tushunish va tahlil qilish imkoni yaratilgan [8]. Bundan tashqari, loyihalash jarayonida BPMN diagrammalari orqali turli modullar o‘rtasidagi o‘zaro aloqa va ma’lumotlar oqimi aniq belgilab olinganligi kelgusida dasturlash bosqichida xatoliklarni kamaytirishga xizmat qilgan [9]. Xulosa qilib aytganda, UML va BPMN metodologiyalari sentiment tahlil tizimini loyihalashda ham muhim yordamchi vositalar bo‘lib, ular yordamida tizimning konseptual tuzilmasini boshidanoq to‘g‘ri rejalashtirish va manfaatdor tomonlar (dasturchilar, buyurtmachilar) bilan samarali aloqa o‘rnatish mumkin.

Tizim arxitekturasi (modul va qatlamlar)

Tizimning umumiy arxitekturasi uch qavatli modelga asoslanadi: **tashqi (frontend) qatlami**, **ilova (backend) qatlami** va **ma’lumotlar qatlami [10]**. Frontend qatlami foydalanuvchi uchun grafik interfeys (React asosida veb-ilova)ni ta’minlaydi va brauzerda ishlaydi. Backend qatlami esa Flask yordamida yaratilgan REST API xizmatidan iborat bo‘lib, biznes mantiqini bajaradi va foydalanuvchidan kelgan so‘rovlarni qayta ishlaydi [11]. Ma’lumotlar qatlami sifatida SQLServer2022 ma’lumotlar bazasi qo‘llanilib, tahlil qilingan matnlar va ularning natijalari shu yerda saqlanadi. Shu qatlamlar o‘rtasida integratsiya xizmati (ichki API so‘rovlar yoki xabarlar tizimi) orqali o‘zaro ma’lumot almashinuvi yo‘lga qo‘yiladi. ML modeli alohida modul sifatida backend ichiga kiritilgan bo‘lib, matnlarni baholash uchun chaqiriladi. Bunday arxitekturaviy ajratish tizimning har bir qatlami mustaqil ishlab chiqilishi va kerak bo‘lsa alohida o‘lchamda masshtablanishini ta’minlaydi [12]. Frontend va backend o‘rtasidagi aloqalar HTTP(s) protokoli orqali JSON formatdagi so‘rovlar va javoblar shaklida tashkil etiladi. Quyidagi 1-rasmda o‘zbek matnlari sentiment tahlil tizim komponentlari va ularning o‘zaro aloqasi aks etgan arxitektura keltirilgan.

Bu arxitekturada foydalanuvchi brauzer orqali tizimga murojaat qiladi va frontend komponenti orqali matnlarni kiritadi. Frontend UI komponenti kiritilgan ma’lumotni REST API orqali backend serveriga uzatadi. Backend esa kelgan so‘rovni qabul qilib, avvalo **matnni oldindan tayyorlash (preprocessing)** moduliga yuboradi va bu modul matndagi *keraksiz belgilarni tozalash, normalizatsiya* va *tokenizatsiya* kabi ishlarni bajaradi. Tozalangan matn so‘ng ML modeliga beriladi, u matnning sentymentini aniqlaydi. Natijada hosil bo‘lgan ijobiy, salbiy yoki neytral baho qiymati backend tomonidan qayta ishlanadi va ma’lumotlar bazasida natijalar saqlanadi.

Integratsiya xizmati agar tizim boshqa tashqi servislar (masalan, ijtimoiy tarmoqlardan ma’lumot olish yoki messenjer botiga ulanish) bilan bog‘lanishi zarur bo‘lsa, buni amalga oshiradi. Matnlarni Twitter API orqali yig‘ish moduli yoki natijalarni chat-bot orqali yuborish xizmati shunday integratsion komponent sifatida qo‘shilishi mumkin. Yakunda backend olingan sentiment natijasini frontendga JSON javob ko‘rinishida qaytaradi. Frontend esa foydalanuvchiga natijani qulay ko‘rinishda, matnning umumiy sentimenti va ehtimoliy baholari bilan ekranga chiqaradi.



1-rasm. O‘zbek matnlari sentiment tahlil tizim arxitekturasi.

Tizimni loyihalash jarayonida Unified Modeling Language (UML) vositalari yordamida uning tarkibi va xatti-harakatlari modellashtiriladi. UML – bu dasturiy tizimlar arxitekturasini vizuallashtirish, belgilash, qurish va hujjatlashtirish uchun xizmat qiluvchi standartarashgan grafik modellash tili hisoblanadi. Ushbu tizim uchun bir nechta asosiy diagrammalar tuziladi:

1) Foydalanish holatlari diagrammasi (Use Case diagram);

2) Oqim diagrammasi (Flowchart yoki Activity diagram);

3) *Komponentlar diagrammasi (Component diagram).*

Foydalanish holatlari diagrammasi tizimdan foydalanuvchi va tizimning o‘zaro aloqalarini ssenariy shaklida aks ettiradi. Masalan, foydalanuvchi uchun *“Matn kiritish”, “Tahlil natijasini ko‘rish”* kabi use-case’lar shakllantirilib, ular ushbu diagrammada foydalanuvchi aktori va tegishli use-case oval shakllari bilan ko‘rsatiladi. Use-case diagramma foydalanuvchining tizim orqali bajarishi mumkin bo‘lgan barcha asosiy funksional holatlarni umumiy ko‘rinishda beradi.

Aktivlik (oqim) diagrammasi esa sentiment tahlil jarayonining bosqichma-bosqich oqimini ifodalaydi. Bunda ma’lumot oqimi va amallar ketma-ketligi bloklar va o‘qlar yordamida chizilib, masalan: **“Matnni yuklash → Oldindan qayta ishlash → ML modelda baholash → Natijani chiqarish”** kabi ketma-ketlik shaklida tasvirlanadi. Oqim diagrammasi bosqichlar orasidagi shartlar va tarmoqlanishlarni ham ko‘rsatib, tizimning ichki ishlash jarayonini yaxlit tushunishga yordam beradi.

Ma’lumotlar oqimi diagrammasi vositasida matnlar qanday manbadan olinishi, qanday qayta ishlanib, natijalar qayerga yuborilishi ketma-ketlikda tasvirlanadi: **“Foydalanuvchi sharhini yig‘ish → Polaritetini aniqlash → Natijani bazaga yozish → Vizualizatsiya” oqimi**. Tadqiqotlarda keltirilishicha, bunday diagramma sentiment tahlil jarayonining bosqichlarini aniq tushunishga yordam beradi.

Komponentlar diagrammasi tizimning moddiy tuzilishini, ya’ni dasturiy modullar va ularning bog‘lanishlarini ko‘rsatadi. Bizning holatda komponentlar diagrammasida quyidagilar bo‘ladi: **Foydalanuvchi interfeysi moduli (frontend), REST API moduli (backend), ML model moduli, ma’lumotlar bazasi** va **Integratsiya xizmati**. Ushbu diagrammada har bir modul alohida blok bilan ifodalanib, ular orasidagi bog‘lanishlar **frontend → backend, backend → ML model, backend → MB** o‘qlar bilan ko‘rsatiladi. Komponentlar diagrammasi tizim arxitekturasi haqida yuqori darajadagi tasavvur berib, dasturiy ta’minot qismlari qanday tashkil etilganini hujjatlashtiradi.

O‘zbek matnlari sentiment tahlil tizimi ishlab chiqish bosqichlari

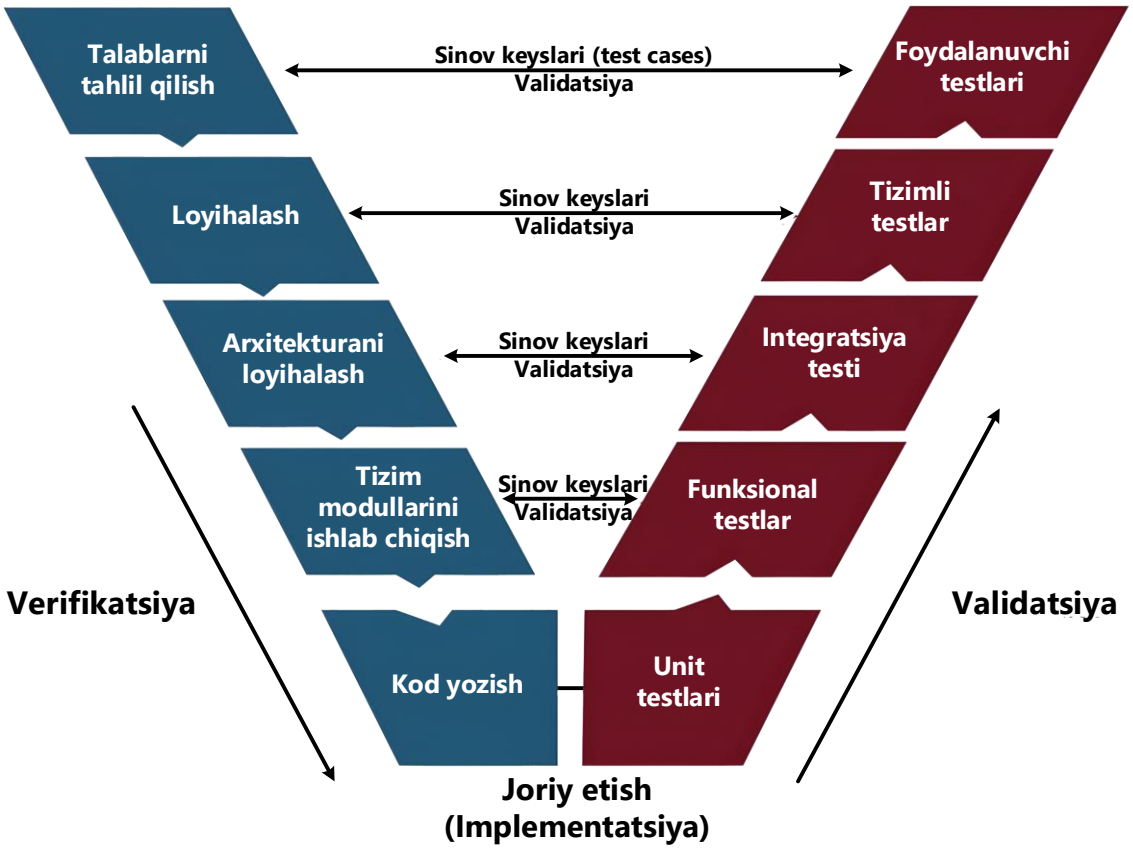
Dasturiy ta’minotni ishlab chiqishning hayotiy sikli **(Software Development Life Cycle, SDLC)** – bu dasturiy mahsulotni konsepsiyadan to to‘liq ishga tushirishgacha bo‘lgan bosqichma-bosqich rivojlantirish jarayonidir. SDLC dasturiy ta’minot sifati va ishlab chiqish samaradorligini oshirishga qaratilgan bo‘lib, u orqali loyiha bosqichlari aniq belgilab chiqiladi. Natijada, yuqori sifatli va foydalanuvchi talablariga javob beruvchi mahsulotni belgilangan muddat va byudjet doirasida yaratish imkoniyati ortadi. SDLC, xuddi binoni qurishdagi me’moriy reja singari, dasturiy loyihani oldindan puxta rejalash va boshqarish imkonini beradi. Bu jarayon axborot tizimlari, tabiiy tilni qayta ishlash (NLP) kabi murakkab loyihalarda ayniqsa muhim. NLP tizimini ishlab chiqishda dastlab til korpusini tayyorlash va talablarni aniqlash, keyin esa model arxitekturasini loyihalash va bosqichma-bosqich sinovdan o‘tkazish zarur bo‘ladi. SDLC 7 asosiy bosqichdan iborat:

1) Rejalashtirish (Planning)
2) Talablarni tahlil qilish (Analysis)
3) Loyihalash (Design)
4) Dasturlash (Development)
5) Sinovdan o‘tkazish (Testing)
6) Joriy etish (Deployment)
7) Texnik **qo‘llab-quvvatlash (Maintenance)**

Quyida ushbu bosqichlarning mazmun-mohiyati va ularning axborot tizimlarini yaratishdagi o‘rni bosqichma-bosqich tahlil qilinadi.

Akademik nuqtai nazardan, SDLC – bu tizimli metodologiya bo‘lib, har bir bosqichda bajariladigan jarayonlar va yaratiladigan hujjatlar jamlanmasini o‘z ichiga oladi.

Har qanday axborot tizimi kabi, sentiment tahlil tizimini yaratish ham bir necha bosqichli jarayondan iborat bo‘lib, har bosqichda turli vazifalar hal etiladi. Jahon amaliyotida keng qo‘llaniladigan dasturiy ta’minotni ishlab chiqish hayotiy davri modeliga ko‘ra, tizim yaratish jarayoni quyidagi asosiy bosqichlarni o‘z ichiga oladi:



2-rasm. O‘zbek matnlari sentiment tahlil tizimi ishlab chiqish bosqichlari

1. **Rejalashtirish bosqichi.** Rejalashtirish – SDLC ning birinchi va eng muhim bosqichi bo‘lib, loyiha muvaffaqiyatining poydevorini yaratadi. Ushbu bosqichda bajariladigan asosiy ishlar:

– **Muammoni aniqlash va maqsadni belgilash:** Dasturiy ta’minot yechmoqchi bo‘lgan asosiy muammo va undan kutilayotgan foyda aniqlanadi. Sentiment tahlil tizimi yaratishdan maqsad – matnlardagi sentiment hissiyot skori (bahosi)ni aniqlash.

– **Loyiha doirasini va cheklovlarini belgilash:** Kelajakdagi dasturiy ta’minotning funksional imkoniyatlari, ko‘lam (scope) va chegaralari belgilanadi. Bu jarayonda loyiha hajmi va murakkabligi baholanadi.

– **Texnik va iqtisodiy imkoniyatlarni o‘rganish (feasibility study):** Loyihani amalga oshirish uchun zarur texnologiyalar va resurslar, dasturchilar jamoasi tajribasi va byudjet taxminan baholanadi. Shuningdek, loyiha oldidagi xavf-xatarlar (risklar) ro‘yxati tuziladi va ularning ehtimoli hamda ta’siri tahlil qilinadi.

– **Ish rejasi va jadval tuzish:** Loyiha bosqichlari bo‘yicha ish reja (projekt rejasi) hamda taxminiy muddatlar grafigi tuziladi. Unda muhim bosqichiy vazifalar va milestonelar belgilanadi. Bu reja loyiha davomida bajarilayotgan ishlarni nazorat qilish imkonini beradi.

Rejalashtirish bosqichining yakuniy natijasi sifatida odatda **loyiha konseptsiyasi (Project Charter)** va **Dasturiy talablarlarni aniqlash hujjati (Software Requirement Specification, SRS)** ishlab chiqiladi. Shu bilan birga dastlabki iqtisodiy modellash amalga oshiriladi. COCOMO

(Constructive Cost Model) modeli yordamida dastur hajmi va kod qatorlari soniga qarab mehnat hajmi (Effort E) baholanadi. Bunday hisob-kitoblar rejalashtirish bosqichida loyiha narxi va davomiyligini prognoz qilish, byudjet ajratish hamda resurslarni rejalashtirishda qo‘l keladi.

2. Talablarni tahlil qilish bosqichi. Talablarni yig‘ish va tahlil qilish – dasturiy mahsulotdan foydalanuvchilar va buyurtmachi kutayotgan funksional va funksional bo`lmagan talablarni aniqlash bosqichidir. Bu bosqichda, sentiment tahlil tizimi qaysi turdagi matnlar bilan ishlashi, til modeli qamrovi, natijalarni ko‘rinish shakli kabi jihatlar belgilanadi. Ushbu bosqichda loyiha jamoasi quyidagilarni amalga oshiradi:

– **Manfaatdor tomonlar bilan muloqot:** Tizimdan foydalanuvchilari, buyurtmachilar, tizim analitiklari va boshqa manfaatdor shaxslar (stakeholders) bilan intervyu, so‘rovnomalar va fokus-guruhlar orqali aloqa o‘rnatiladi. Masalan, NLP asosida ishlovchi matnni tahlil qilish tizimi yaratayotgan bo‘lsak, lingvist-mutaxassislar va potensial foydalanuvchilarning istaklari o‘rganiladi.

– **Talablarni hujjatlashtirish:** Olingan barcha ehtiyoj va istaklar tahlil qilinib, zarur va ikkinchi darajali funksiyalar ajratiladi. Bu jarayonda *“zarur” (must-have)* va *“yaxshi bo‘lardi” (nice-to-have)* funksiyalar farqlanadi. Foydalanuvchi talablarining to‘liq ro‘yxati va ularning ustuvorlik darajasi aniqlanadi.

– **Talablar spetsifikatsiyasi (SRS)ni tuzish:** Barcha yig‘ilgan talablar asosida Talablar spetsifikatsiyasi hujjati ishlab chiqiladi. Ushbu hujjatda dasturiy ta’minotning maqsadi, vazifalari, funksional imkoniyatlari, ishlash unumdorligi, xavfsizlik talableri, interfeys talablarigacha batafsil bayon etiladi. Talablar hujjati loyiha jamoasi uchun kelgusida yo‘l xaritasi bo‘lib xizmat qiladi va loyiha miqyosini aniq belgilaydi.

– **Talablarni verifikatsiya va tasdiqlash:** Yozib chiqilgan talablar buyurtmachi va foydalanuvchilar tomonidan qayta ko‘rib chiqilib, tasdiqlanadi. Bu bosqich talablarning to‘liq va to‘g‘ri tushunilganini tekshirish uchun muhim, aks holda keyingi bosqichlarda noaniqliklar yuzaga kelishi mumkin.

Talablarni tahlil qilish bosqichining muvaffaqiyati butun loyiha muvaffaqiyatini belgilaydi. To‘g‘ri va aniq aniqlangan talablar natijasida yaratilgan dasturiy mahsulot foydalanuvchilarning ehtiyojlarini qondiradi yoki hatto ularning kutilmalaridan oshadi. Aksincha, noto‘g‘ri yoki noaniq talablar loyiha davomida jiddiy muammolarga va qayta ishlashga olib kelishi mumkin. Shu sababli, bu bosqichda manbalarga boy tahlil va muvofiqlashtirilgan muloqot katta ahamiyatga ega.

3. Loyihalash bosqichi. Loyihalash bosqichida dasturiy tizimning umumiy tuzilishi va yechimning *“qurilishi”* rejalashtiriladi. Agar talablar bosqichida nima qilish kerakligi belgilangan bo‘lsa, loyihalash bosqichida qanday qilish kerakligi aniqlashtiriladi. Belgilangan talablar asosida tizimning arxitekturasi, ma’lumotlar bazasi strukturalari, modullar va ularning o‘zaro aloqasi loyihalashtiriladi. Bu bosqichda yuqorida ta’kidlanganidek UML diagrammalari, konseptual sxemalar tuziladi va kelajakdagi dasturiy strukturaning *“qurilishi”* rejalashtiriladi. Bu bosqichda quyidagi masalalar hal etiladi:

– **Arxitektura tuzilmasi:** Tizimning yuqori darajadagi arxitekturasi tuziladi. Tizimning asosiy komponentlar (modullar)i va ularning o‘zaro aloqasi aniqlanadi. Masalan, veb-ilova uchun *“mijoz-server”* arxitekturasi yoki NLP tizimi uchun modullar (matnni oldindan ishlov berish moduli, ML modeli, foydalanuvchi interfeysi va h.k.) o‘rtasidagi bog‘liqliklar chizib chiqiladi.

– **Ma’lumotlar bazasi va ma’lumotlar modeli tuzilmasi:** Sentiment tahlil tizimi ma’lumotlar tuzilmasi va obyektlar o‘rtasidagi bog‘lanishlar (ER-diagramma) loyihalashtiriladi. Bu

yerda muhim ma’lumot obyektlari va ularning atributlari, o‘zaro munosabatlari ko‘rsatiladi. Shuningdek, ma’lumotlar oqimi diagrammalari (DFD) tuzilib, tizim ichida ma’lumotlar qanday oqib o‘tishi tasvirlanadi.

– **Interfeys dizayni va foydalanish qulayligi (UX/UI):** Foydalanuvchi interfeysi elementlari, navigatsiya sxemalari, dasturiy ta’minotning ko‘rinishi va funksional imkoniyatlarining foydalanuvchi uchun qulayligi rejalashtiriladi. Masalan, veb-ilovaning sahifalari uchun *wireframe* va *mockuplar* chiziladi, NLP tizimi uchun esa foydalanuvchidan matn kiritish va natijani ko‘rsatish jarayonining soddaligi ta’minlanadi.

– **Texnik yechimlar va integratsiya nuqtalarini aniqlash:** Tizim qaysi dasturlash tilida yozilishi, qanday kutubxona va freymvorklardan foydalanilishi, agar mavjud tizimlar bilan integratsiya qilinishi lozim bo‘lsa, bunday bog‘lanish nuqtalari aniqlanadi. Masalan, kompyuter lingvistikasi tizimida mavjud til resurslari (lug‘atlar, korpuslar) bilan ishlash uchun tegishli API’lar belgilanadi.

Loyihalash bosqichining yakunida **Dasturiy tuzilma hujjati (Software Design Document, SDD)** tayyorlanadi. Ushbu hujjatda tizim arxitekturasi, ma’lumotlar bazasi sxemasi, modul va komponentlarning vazifalari, ularning o‘zaro aloqasi va interfeyslari batafsil bayon etiladi. Tuzilma hujjati keyingi kod yozish bosqichida dasturchilar uchun yo‘l xaritasi vazifasini o‘taydi. Loyihalash bosqichi – bu talablar bilan ijro (kodlash) o‘rtasidagi ko‘prikdir: to‘g‘ri loyihalangan tizim nafaqat foydalanuvchi talablariga mos keladi, balki kelgusida oson kengaytiriladigan va xizmat ko‘rsatishga qulay tizim bo‘lishini ham ta’minlaydi.

4. Kodlash (ishlab chiqish) bosqichi. Kodlash bosqichi (ba’zan ishlab chiqish bosqichi deb ham yuritiladi) – bevosita dasturiy ta’minotni yaratish, ya’ni kod yozish jarayonidir. Ushbu bosqichda dasturchilar jamoasi loyihalashda belgilab olingan reja va blueprintlarga amal qilgan holda, tizim funksiyalarini dasturlashni boshlaydilar. Agar tizim murakkab bo‘lsa, ayrim qismi yoki algoritmlarini matematik modellashtirish, prototip dasturlar yaratib sinab ko‘riladi. Ilmiy ishlarda, xususan, o‘zbek tilida sentiment tahlil qilish tizimini yaratishda tadqiqotchilar avvalo matematik va lingvistik modellarni ishlab chiqqan holda, ularning samaradorligini kichik tajriba dasturlari orqali sinab ko‘rishgan. Bu bosqich kelgusida to‘liq tizimni ishlab chiqishda xatolarni barvaqt ko‘rib chiqish va yechimini topishga imkon beradi. Bunda oldingi bosqichlarda ishlab chiqilgan loyiha asosida alohida modullar (masalan, matnni tozalash, so‘zlarni belgilash, klassifikatsiya modeli, natija chiqarish, foydalanuvchi interfeysi) alohida dasturlanadi va so‘ngra yagona tizimga integratsiya qilinadi. Bir necha mahalliy loyiha va ilmiy ishlar tajribasida dasturlash uchun Python tili va uning sun’iy intellekt kutubxonalari tanlangan, natijada prototip tizim tayyor bo‘lgach, u Django veb-dasturi bilan integratsiya qilinadi. Kodlash bosqichining asosiy xususiyatlari:

– **Standartga amal qilgan holda kod yozish:** Dasturchilar oldindan kelishilgan kod yozish standartlari (koding standartlari) va nomlash konvensiyalariga amal qilgan holda kod yozadilar. Bu *source* kodning yagona uslubda bo‘lishini, turli jamoa a’zolari yozgan kodning o‘zaro mos va tushunarli chiqishini ta’minlaydi. Masalan, barcha dasturchilar funksiyalar va o‘zgaruvchilarni nomlashda bir xil konvensiyaga rioya qilishadi, kodning tuzilishi yakdil qoidalarga bo‘ysunadi.

– **Modullar bo‘yicha ishlab chiqish:** Katta dasturiy tizim bo‘limlarga (modullarga) bo‘linib, har bir dasturchi yoki kichik guruh aniq bir modul ustida ishlaydi. Bu modul tamoyili kodni tartibli va boshqariladigan holda yozishga yordam beradi.

– **Versiya nazorati va integratsiya:** Dasturchilar versiya nazorati tizimlari (masalan, Git) orqali o‘zgarishlarni boshqarib boradilar. Har bir yozilgan kod bo‘lagi (commit) tizimga integratsiya qilinadi va dasturiy ta’minotning ishchi talqini muntazam yangilanib boradi. Bu *CI (Continuous Integration)* amaliyotiga mos keladi va jamoa kodni birlashtirib borar ekan, integratsion muammolar erta bosqichda aniqlanadi.

– **Kod tekshiruvi (Code Review):** Jamoa ichida yozilgan kod o‘zaro tekshirib boriladi va bir dasturchi yozgan kodni boshqa hamkasbi ko‘rib chiqadi, potensial xatolar, kamchiliklar yoki yaxshilash imkoniyatlari yuzasidan fikr bildiradi. Bunday kod revyu jarayoni dasturiy ta’minot sifatini oshiradi, xatolarni erta bosqichda bartaraf etish va eng yaxshi amaliyotlarni joriy etishga xizmat qiladi.

– **Ichki sinov va prototiplash:** Kodlash jarayonida har bir modul yozilgach, dasturchilar tomonidan dastlabki ichki testlar o‘tkaziladi. Ya’ni, modul asosiy funktsiyani bajarayotgan-bajarmayotganini tezkor tekshirish (unit-test) amalga oshiriladi. Shuningdek, agar loyiha prototiplar orqali iterativ rivojlantirilayotgan bo‘lsa, kodlash bosqichida ishlab chiqilgan dastlabki prototip foydalanuvchilarga yoki buyurtmachiga namoyish etib boriladi va bu feedback olib, kerak bo‘lsa, dizayn yoki talablarni tezda o‘zgartirish imkonini beradi.

Kodlash bosqichi yakunida **dasturiy ta’minotning ishlaydigan versiyasi** (yoki minimal ishchi mahsulot – Minimum Viable Product, MVP) hosil bo‘ladi. Bu hali to‘liq xatolardan xoli bo‘lmasligi mumkin, ammo u oldingi bosqichlarda belgilangan funksional talablarning asosiy qismini amalda bajarib turadi. Axborot tizimlari va NLP loyihalarida kodlash bosqichi ayniqsa ijodiy jarayon bo‘lib, algoritmlar va modellarning bevosita hayotga tatbiq etilishidir. Masalan, NLP tizimi uchun bu bosqichda matnni tozalash va pars qilish algoritmlari, mashina o‘rganish modeli arxitekturasi kodda yoziladi, va dastlabki natijalar olinadi. Kodlash bosqichi – bu oldingi bosqichlardagi reja va loyihalarning “jonlanishi”, ya’ni real dasturiy mahsulotga aylanish jarayonidir.

5. Sinovdan o‘tkazish bosqichi. Sinov bosqichi – yozilgan dasturiy ta’minotni sifat nazoratidan o‘tkazish, xatolarni aniqlash va tuzatish bosqichidir. Bu bosqichni ishlab chiqarish tarmoqli zavodidagi mahsulot sifat tekshiruvi bilan qiyoslash mumkin. Tizimning barcha komponentlari tayyor bo‘lgach, ularni funksional va yuklama bo‘yicha sinovdan o‘tkazish bosqichi keladi. Bu yerda tizimga turli test ma’lumotlar kiritilib, uning to‘g‘ri ishlashiga ishonch hosil qilinadi; xatolar aniqlansa, tegishli tuzatishlar kiritiladi. Sentiment tahlil tizimi uchun sinov bosqichida har xil ko‘rinishdagi matnlar (ijobiy, salbiy, neytral, turli uzunlik va uslubdagi) sinab ko‘riladi va modelning aniqlik, to‘g‘rilik kabi ko‘rsatkichlari baholanadi. Ayrim manbalarda o‘zbek tilidagi model sinovlari natijasida 92–93% aniqlikka erishilgani qayd etilgan bo‘lib, sinovlar jarayonida aniqlangan zaif joylar asosida modelga tuzatishlar kiritilgan. Sinov bosqichida quyidagi jarayonlar amalga oshiriladi:

– **Test rejasi va test holatlari (test case) ishlab chiqish:** Dasturiy ta’minotning talablariga mos holda sinov strategiyasi belgilanadi. Qaysi funksiyalarni qanday sharoitlarda tekshirish kerakligi, kutilayotgan natijalar nimalardan iborat bo‘lishi lozimligi aniqlanadi. Buning uchun test holatlari yoziladi va har bir test holatida dasturga beriladigan kiruvchi ma’lumotlar va kutilayotgan chiqishlar ko‘rsatiladi. Masalan, matn tarjimasi tizimi uchun bir necha turdagi jumlalar (oddiy, murakkab, idiomatik) kiritilib, tarjima natijasi inson tarjimasi bilan solishtiriladi.

– Turli xil test turlarini o‘tkazish: Dasturiy ta’minotning barcha jihatlarini qamrab olish uchun bir necha bosqichli test sinovlari amalga oshiriladi. Jumladan:

– *Birlik testi (unit testing):* Dasturiy ta’minotni tashkil etuvchi alohida modullar va funksiyalar izolyatsiya holda test qilinadi. Har bir modul o‘z vazifasini to‘g‘ri bajarayotgani tekshiriladi.

– *Integratsiya testi:* Modullar birgalikda ishlaganda yuzaga keluvchi muammolarni aniqlash uchun o‘zaro bog‘liq qismlar birlashtirilib sinovdan o‘tkaziladi.

– *Tizim testi:* Tizim to‘liq majmua holida ishga tushirilib, yakuniy foydalanuvchi nuqtai nazaridan barcha funksiyalarining to‘g‘ri ishlashi tekshiriladi.

– *Qabul testi (acceptance testing):* Buyurtmachi yoki oxirgi foydalanuvchilar ishtirokida dasturiy mahsulot real foydalanish sharoitiga yaqin muhitda sinovdan o‘tadi. Agar bu testdan muvaffaqiyatli o‘tsa, dasturiy ta’minot qabul qilinishga tayyor hisoblanadi.

– *Xavfsizlik testi:* Tizimning xavfsizlik talablariga javob berishi, ma’lumotlar himoyalanganligi, ruxsatsiz kirishlardan holi ekanligi sinovdan o‘tadi (agar dasturiy ta’minot uchun bu dolzarb bo‘lsa).

– *Yuklama va stress testi:* Tizim ko‘p foydalanuvchilar, katta ma’lumotlar oqimi yoki chegaraviya holatlar (masalan, server resurslari kamayganda)da barqaror ishlash-qilmasligi tekshiriladi.

– Xatolarni qayd etish va tuzatish: Har bir muvaffaqiyatsiz test holati natijasida aniqlangan xato va kamchiliklar batafsil hujjatlashtiriladi. Xatolikning alomati, qanday sharoitda sodir bo‘lishi, dasturiy ta’minotga ta’siri kabi ma’lumotlar qayd qilinadi. Ushbu xatolar dasturchilarga yetkazilib, kodni tuzatish ishlari amalga oshiriladi. Tuzatish kiritilgach, tegishli testlar takrorlanadi va xato bartaraf etilgunga qadar bu jarayon davom etadi.

– Sinov natijalarini tahlil qilish: Barcha testlar bajarilgach, umumiy natijalar ko‘rib chiqiladi. Agar jiddiy xatolar qolmagan bo‘lsa va dastur belgilangan talablarga muvofiq ish bersa, loyiha keyingi bosqichga o‘tadi. Aks holda, kerak bo‘lsa, ayrim talablar qayta ko‘rib chiqilishi yoki orqaga kodlash bosqichiga qaytilishi ham mumkin.

Sinov bosqichi dasturiy ta’minot sifatini ta’minlashda hal qiluvchi ahamiyatga ega. Ayniqsa, axborot tizimlari uchun, tizimning ishonchliligi va barqaror ishlashi muhim. Shu bois puxta test qilingan tizimgina iste’molchi qo‘liga topshiriladi. NLP tizimlari misolida, sinov bosqichida modelning **aniqlik, to‘g‘rilik (precision/recall)** kabi ko‘rsatkichlari baholanadi, turli sinov korpuslarida model ishlashi tekshiriladi va zarurat tug‘ilsa, model qayta o‘qitilib, natijalar takomillashtiriladi.

6. Joriy etish va ishga tushirish bosqichi. Joriy etish (deployment) bosqichi – dasturiy mahsulotni ishchi muhitga (production muhitiga) o‘tkazish va foydalanuvchilarga yetkazish bosqichidir. Sinovlardan muvaffaqiyatli o‘tgan tizim real muhitga joriy etiladi, foydalanuvchilar uchun foydalanish yo‘riqnomalari ishlab chiqiladi. Shuningdek, ushbu bosqichda tizimning ekspluatatsiya jarayonida monitoringi, foydalanuvchi fikr-mulohazalarini yig‘ish va keyingi versiyalar uchun takomillashtirish rejalari tuziladi. Bu bosqichga kelib dasturiy ta’minot sinovlardan o‘tgan va qabulga tayyor deb topilgan bo‘ladi. Joriy etish bosqichida:

– **Ishchi muhitga o‘tkazish:** Dasturiy mahsulot ishlab chiqarish muhitiga o‘rnatiladi va ishga tushiriladi. Ko‘p hollarda kompaniyalar dasturiy ta’minotni bir yo‘la barcha foydalanuvchilarga emas, balki bosqichma-bosqich o‘tkazadilar. Masalan, dastlab *sinov muhitida*

(staging) yoki *sandbox* muhitida ishga tushirib ko‘rishadi, so‘ngra cheklangan foydalanuvchilar guruhiga beta-versiya sifatida taqdim etishadi, oxirida esa to‘liq *produktsion* muhitga joylashadi. Bunday yondashuv dasturiy ta’minotni real muhitda ham tekshirib ko‘rish va kutilmagan muammolarni keng auditoriyaga chiqmasdan oldin tuzatish imkonini beradi.

– **Joylashtirish strategiyasini tanlash:** Loyihaning xususiyatlariga qarab, joriy etishning turli strategiyalari qo‘llanadi. Masalan, “Big Bang” – mahsulotni bir zumda hamma foydalanuvchilarga ishga tushirish; “Blue-Green” – eski versiya (blue) yonida yangi versiya (green) parallel ishga tushirib, sinovdan so‘ng trafikni yangi versiyaga o‘tkazish; yoki “Kanareyka (Canary) joriy etish” – foydalanuvchilarning kichik qismida yangi versiyani sinab ko‘rib, so‘ng bosqichma-bosqich ulushini oshirish. Bunday usullar foydalanuvchi tajribasiga minimal ta’sir bilan yangilanishlarni tatbiq etish imkonini beradi.

– **Foydalanuvchi qo‘llanmasi va o‘qitish:** Yangi tizim foydalanuvchilarga taqdim etilayotganda, ularga yo‘riqnoma va qo‘llanmalar beriladi. Zarur hollarda trening va seminarlar o‘tkazilib, foydalanuvchilarga tizimdan foydalanish qoidalari o‘rgatiladi. Ayniqsa, korporativ axborot tizimlarini joriy etishda bu bosqich muhim va xodimlar yangi tizimga moslashishi va uning imkoniyatlaridan to‘liq foydalana olishi lozim.

– **Texnik infratuzilmani sozlash:** Mahsulotni ishga tushirishda serverlar, ma’lumotlar bazasi, tarmoq sozlamalari kabi infratuzilma komponentlari tayyorlanadi. Agar bu bulutli servis bo‘lsa, DevOps jarayonlari (CI/CD pipeline) orqali avtomatlashtirilgan deploy qilinadi.

– **Foydalanuvchi fikr-mulohazalarini yig‘ish:** Mahsulot ishga tushirilgach, dastlabki foydalanuvchilardan fikr va mulohazalar olinadi. Bu ma’lumotlar keyingi yangilanishlarda hisobga olinishi uchun jamlanadi.

Joriy etish bosqichi mahsulotning loyiha rejimidan ishchi rejimga o‘tishini anglatadi. Ushbu bosqich yakunida dasturiy ta’minot real foydalanuvchilar qo‘lida o‘z vazifasini bajara boshlaydi. Bu bosqich yakuniy nuqta emas, balki yangi bosqich, ya’ni texnik xizmat va qo‘llab-quvvatlash bosqichining boshlanish nuqtasidir. Misol tariqasida, NLP modeli asosidagi xizmatni ishga tushirganimizda, real foydalanuvchi matnlarini qayta ishlash jarayonida model qanday natija berayotgani kuzatiladi, foydalanuvchilarning qoniqish darajasi tahlil qilinadi.

7. Texnik xizmat ko‘rsatish (qo‘llab-quvvatlash) bosqichi. Texnik xizmat va qo‘llab-quvvatlash – dasturiy ta’minotni ishlab chiqishning yakuniy va doimiy davom etuvchi bosqichi bo‘lib, mahsulot ishga tushirilgandan so‘ng uning barqaror ishlashi, yangilanishi va takomillashtirib borilishini nazarda tutadi. Bu bosqichda quyidagilar amalga oshiriladi:

– **Xatolarni tuzatish:** Foydalanuvchilar tizimdan foydalana boshlagach, avval topilmagan ba’zi xatolar (baglar) aniqlanishi mumkin. Texnik xizmat ko‘rsatish jarayonida ishlab chiquvchilar bunday nosozliklarni bartaraf etishadi va dasturiy yamalar (patch) tarzida tizimga joriy qilishadi. Bu doimiy monitoring va tezkor reaksiyani talab qiladi. Xususan, agar yangi versiya jiddiy xatoga ega bo‘lsa, hot-fix tarzida zudlik bilan tuzatish chiqarilishi zarur.

– **Funktsional imkoniyatlarni kengaytirish:** Vaqt o‘tishi bilan foydalanuvchilarning yangi talab va ehtiyojlari paydo bo‘lishi mumkin, yoki bozor talabiga ko‘ra dasturga yangi funksiyalar qo‘shish zaruriyati tug‘iladi. Qo‘llab-quvvatlash bosqichida tizimga yangilanishlar va versiyalar orqali yangi funksiyalar qo‘shib boriladi. Bu aslida SDLC jarayonining yana dastlabki bosqichlariga ma’lum ma’noda qaytishni bildiradi. Ya’ni, yangi qo‘shimchalar uchun mini-rejalashtirish, dizayn, kodlash va testdan o‘tish jarayonlari takrorlanadi (spiral tarzda rivojlantirish).

– **Ishlash sifatini oshirish va optimizatsiya:** Mahsulot ishlayotgan davrida uning ishlash ko‘rsatkichlari (performance) kuzatib boriladi. Masalan, veb-tizim bo‘lsa, uning javob berish vaqti, server yuklamasi, foydalanuvchi kirishlari soniga qarab ko‘rsatkichlari monitoring qilinadi. Zaruriyat tug‘ilganda, kod optimizatsiya qilinadi yoki server resurslari kengaytiriladi. NLP tizimi uchun, vaqt o‘tishi bilan modelni yangi ma’lumotlarda qayta o‘qitish (retraining) ham optimizatsiya turiga kiradi. Chunki til ma’lumotlari yangilanib boradi yoki dastlabki modelda yetarlicha hisobga olinmagan holatlar paydo bo‘lishi mumkin.

– **Xavfsizlik yangilanishlari:** Dasturiy ta’minot uchun xavfsizlik juda muhim, ayniqsa axborot tizimlarida. Qo‘llab-quvvatlash bosqichida yangi aniqlangan kiberxavfsizlik xatarlari bo‘yicha tizim doimiy yangilanadi, xavfsizlik yamalari chiqariladi va foydalanuvchilarga o‘z dasturlarini yangilab borish tavsiya etiladi.

– **Foydalanuvchilarni qo‘llab-quvvatlash:** Texnik xizmat tarkibiga foydalanuvchilarga yordam berish, ularning savollariga javob berish ham kiradi. Ko‘pincha maxsus texnik support jamoasi 24/7 rejimida ishlaydi, foydalanuvchilardan murojaatlar tushsa, muammoni hal qilishda ko‘mak beradi. Bu dasturiy mahsulotga bo‘lgan ishonch va qoniqishning muhim qismi sanaladi.

– **Mahsulotning hayotiy siklini yakunlash (sunsetting):** Vaqt o‘tib tizim eskirganda yoki yangiroq platformaga o‘tish rejalashtirilganda, mahsulotni qo‘llab-quvvatlashni to‘xtatish haqida qaror qabul qilinishi mumkin. Bu bosqichda foydalanuvchilarga oldindan e’lon qilinadi, muqobil yechimlar taklif etiladi va tizim bosqichma-bosqich yopiladi. Misol uchun, eski versiyadagi dastur uchun oxirgi yangilanish va xavfsizlik yamasi chiqarilib, foydalanuvchilarga ma’lum muddatdan keyin ushbu versiya endi qo‘llab-quvvatlanmasligi haqida xabar beriladi.

Qo‘llab-quvvatlash bosqichi eng uzoq davom etadigan bosqich bo‘lib, dasturiy ta’minotning butun foydalanish davomiyligini qamrab oladi. To‘g‘ri yo‘lga qo‘yilgan texnik xizmat dasturiy mahsulotning umrini uzaytiradi, foydalanuvchilarning qoniqishini yuqori darajada ushlab turadi va mahsulotning bozordagi raqobatbardoshligini ta’minlaydi.

Yuqoridagi bosqichlar ketma-ket sanalgan bo‘lsa-da, zamonaviy metodologiyalarda ular iterativ tarzda takrorlanishi ham mumkin. Ya’ni kichik qism ishlab chiqilib sinovdan o‘tgach, navbatdagisiga o‘tiladi va tizim bosqichma-bosqich kengaytiriladi. Adabiyotlarda sentiment tahlil tizimini ishlab chiqishda ham xuddi shunday takrorlanuvchi yondashuv samarali ekani ta’kidlanadi, chunki dastlabki model natijalari va foydalanuvchi fikri asosida tizimni tezkor takomillashtirish imkoniyati bo‘ladi. Muhimi, rivojlanish bosqichlarida puxta rejalashtirish va har bir jarayonni hujjatlashtirish yakunda barqaror va sifatli axborot tizimini yaratishga zamin bo‘ladi.

Axborot tizimlarini ishlab chiqishning hayotiy sikli – SDLC konsepsiyasi dasturiy mahsulot yaratish jarayonini nazariy va amaliy jihatdan asoslashga xizmat qiladi. SDLC doirasida ishlash:

1. *Loyihani puxta rejalash va risklarni boshqarish imkonini beradi.*
2. *Talablar va dizaynda aniqlikni ta’minlaydi.*
3. *Ishlab chiqish jarayonida sifat nazoratini kuchaytiradi.*
4. *Yakuniy mahsulotni o‘z vaqtida va sifatli topshirishga zamin yaratadi.*

SDLC turli rollardagi mutaxassislarni (analitiklar, dasturchilar, testerlar, foydalanuvchilar) yagona maqsad sari birlashtiradi va ularning har bir bosqichdagi ishtirokini aniqlab beradi. SDLCning bir qator modellari mavjud bo‘lib, ularning har biri ma’lum sharoitda maqsadga muvofiqdir – bu haqda munozara qismida sharhlab o‘tildi. SDLC bosqichlarini ongli ravishda qo‘llash va mos modelni tanlash dasturiy mahsulot yaratuvchilariga ijodiy yondashuv va tizimli

nazoratni birlashtirishga imkon beradi. Bu ayniqsa bugungi kunda tez o‘zgaruvchan IT sohasida juda muhim: yangi texnologiyalar (sun’iy intellekt, katta ma’lumotlar) shiddat bilan rivojlanar ekan, ular bilan bog‘liq loyihalarni muvaffaqiyatli amalga oshirish intizomli yondashuvni talab qiladi. SDLC esa aynan shunday intizom va izchillik uchun nazariy va amaliy poydevor bo‘lib xizmat qiladi. Shunday ekan, SDLC tamoyillariga rioya qilgan holda, dasturiy injiniring va axborot tizimlari sohasida yanada ishonchli, samarali va foydalanuvchi talablari darajasidagi yechimlarni yaratish lozim.

SDLC nuqtai nazaridan, texnik xizmat bosqichi eng uzoq davom etadigan va ko‘p resurs talab qiladigan bosqich. Tadqiqotlar ko‘rsatadiki, dasturiy ta'minotga sarflanadigan umumiy resurs va xarajatlarning katta qismi aynan uning texnik xizmat ko‘rsatish davriga to‘g‘ri keladi. Shu sabab, dasturiy mahsulotni loyihalashda kelgusidagi texnik xizmatni yengillashtiradigan yechimlarni tanlash (masalan, kodning o‘qilishi oson bo‘lishi, hujjatlarning to‘liq yozilishi, modulning mustaqilligini saqlash) juda muhim. Axborot tizimlari va NLP mahsulotlari uchun muntazam yangilanish chiqarmasdan iloji yo‘q. NLP modellari yangi ma’lumotlar bilan qayta o‘qitilishi yoki til qoidalariga mos ravishda yangilanib turishi kerak. Texnik xizmat bosqichi mohiyatiga ko‘ra *“loyiha boshidan boshlangan takomillashtirish jarayoni”*ning davomidir. Ya’ni, dasturiy mahsulot hayotga kelgach ham, uni doimiy ravishda parvarishlash zarur. Aks holda, dasturiy mahsulot tez orada eskirolishi, foydalanuvchilarni qoniqtirmay qo‘yishi va natijada foydalanilmay qolishi mumkin. Yuqorida bayon etilgan SDLC bosqichlarini sarhisob qilar ekanmiz, quyidagi jadval ularning har birida bajariladigan asosiy vazifalar, ishlab chiqiladigan hujjatlar va bosqich yakunidagi natijani umumlashtirib ko‘rsatadi (1-jadval):

1-jadval. SDLC bosqichlari bo‘yicha bajariladigan asosiy ishlar va natijalar

Bosqich	Asosiy ishlar (vazifalar)	Hujjat va natijalar
1. Rejalashtirish	Loyiha maqsadini aniqlash; texnik-iqtisodiy asoslash; risklarni baholash; jadval tuzish; resurslarni rejalashtirish; loyiha doirasini belgilash.	Loyiha kontsepsiyasi; Loyiha rejasi; Dastlabki SRS (talablar ro‘yxati).
2. Talablar tahlili	Foydalanuvchi va buyurtmachi talablarini yig‘ish; ularni tahlil qilish va ustuvorliklarini belgilash; aniq va tekshiriladigan talablar shakllantirish; talablarni hujjatlashtirish va tasdiqlatish.	Talablar spetsifikatsiyasi (SRS) hujjati – dasturga qo‘yiladigan barcha talablar ro‘yxati.
3. Loyihalash	Tizim arxitekturasi va komponentlarini aniqlash; ma’lumotlar modeli va interfeys dizaynini ishlab chiqish; DFD, ER diagrammalar tuzish; texnik yechimlarni (algoritmlar, platformalar) tanlash; integratsiya nuqtalarini belgilash.	Dasturiy dizayn hujjati (SDD) – arxitektura tavsifi, komponentlar va ularning o‘zaro aloqa sxemasi, UI maketlar.
4. Dasturlash	Kod yozish (modullarni dasturlash); versiya nazorati ostida hamkorlikda ishlab chiqish; kod standartlariga rioya qilish;	Ishlovchi dasturiy ta’minotning dastlabki versiyasi (prototip yoki

	kutubxonalar va freymvorklardan foydalanish; ichki birlik testlarini o‘tkazish; kodni ko‘rib chiqish (review).	beta-versiya); yaratilgan dastur kodi (repository).
5. Sinov	Test strategiyasini belgilash; test-case larini tayyorlash; birlik, integratsion, tizim va qabul testlarini o‘tkazish; xatolarni aniqlash va qayd etish; xatolarni tuzatish va qayta test qilish (iterativ jarayon).	Sinov hisobotlari; xato (bug)lar ro‘yxati va ularning holati; dastur sifati tasdiqlangani haqida xulosa (tasdiqlangan talablarga muvofiqlik).
6. Joriy etish	Dasturiy ta’minotni ishlab chiqarish muhitiga o‘tkazish; serverlarda o‘rnatish va konfiguratsiya qilish; foydalanuvchilarga dasturdan foydalanish imkonini yaratish; foydalanuvchi qo‘llanmalarini taqdim etish; dastlabki monitoring.	Ishlab chiqarish muhitida ishga tushirilgan dasturiy mahsulot; foydalanuvchilarga e’lon qilingan yangi mahsulot (versiya); sozlangan tizim muhiti.
7. Texnik xizmat	Dastur faoliyatini doimiy kuzatish (monitoring); yuzaga kelgan xatolarni tuzatish (patch releases); o‘zgaruvchan talablar va muhitga mos ravishda dasturga o‘zgartirishlar kiritish; foydalanuvchi takliflariga ko‘ra yangi funksiyalar qo‘shish; tizimni optimallashtirish va yangilab borish; foydalanuvchilarga texnik ko‘mak berish.	Dasturiy mahsulotning yangilangan versiyalari (updatelar); texnik qo‘llab-quvvatlash hujjatlari; foydalanuvchi muammo/takliflari jurnali; (Mahsulot hayoti yakunida: tizimni eskirgan deb topish va dekomission qilish hujjatlari).

Yuqoridagi 1-jadvaldan ko‘rinib turibdiki, SDLC doirasidagi har bir bosqichning aniq vazifalari va natijalari mavjud. Har bir bosqich o‘zidan keyingi bosqich uchun asos yaratadi, shu sababli biror bosqichdagi kamchilik yoki e’tiborsizlik butun loyiha sifatiga salbiy ta’sir ko‘rsatishi mumkin. Shu nuqtai nazardan, SDLC modeli dasturiy ta'minot ishlab chiqish jarayonini izchil boshqarish va nazorat qilishga imkon beradi.

Sentiment tahlil axborot tizimining dasturiy kod tuzilmasi

Sentiment tahlil axborot tizimi kodi aniq va tartibli tuzilishda tashkil etiladi. Kataloglar iyerarxiyasi shunday rejalashtiriladiki, har bir modul va vazifa alohida joylashuvi bilan ajralib tursin. Quyida tizimning namunaviy papkalar strukturasi va ularning vazifalari keltirilgan:

1. **preprocessing/** – Matnlarni oldindan qayta ishlash (tozalash, normalizatsiya, tokenizatsiya)ga oid funksiyalar va skriptlar joylashadi. Masalan, bu yerda matndagi imlo xatolarini to‘g‘rilash, stop-so‘zlarni olib tashlash kabi kodlar bo‘ladi.

2. **classifier/** – Sentiment klassifikatsiya modeli bilan ishlovchi modul. Bu katalogda modelning o‘zi (o‘qitilgan Transformer modelini yuklash yoki sklearn modelini **.pkl** ko‘rinishida

saqlash), matnlardan xususiyatlar ajratish (**feature_extraction.py**) va baholash (**predict.py**) kodlari turadi. Agar modelni o‘qitish ham tizim tarkibida ko‘zda tutilgan bo‘lsa, u holda o‘qitish skripti (**train.py**) va tegishli konfiguratsiyalar ham shu yerda bo‘ladi.

3. **api/** – Flask web-servisi uchun manba kodlari joylashgan katalog. Unda marshrutlar (routes) va endpointlar aniqlanadi (**app.py** yoki **routes.py** ichida **/analyze** API endpointi). Bu yerda foydalanuvchidan kelgan so‘rovni qabul qilish, classifier modulini chaqirish va natijani qaytarish funksiyalari yoziladi. Shuningdek, bu katalogda autentifikatsiya yoki foydalanuvchi sessiyasini boshqarish kabi veb-xizmatga aloqador kodlar ham bo‘lishi mumkin.

4. **visualization/** – Natijalarni vizual tarzda chiqara olish uchun zarur modul. Masalan, agar tizimda biror statistik ko‘rsatkichlarni grafik ko‘rinishda taqdim etish (diagramma chizish) funksionali bo‘lsa, uning kodi shu papkada bo‘ladi. Web interfeysda grafikalar ko‘rsatish yoki frontendga diagramma uchun ma’lumot tayyorlash funksiyalari ham shu yerga joylanadi.

5. **config/** – Loyihaning turli konfiguratsion fayllari va sozlamalari. Masalan, ma’lumotlar bazasi ulanish sozlamalari (**db_config.py** yoki **.env** faylida), model parametrlarini o‘rnatish, dastur ishlash muhitiga tegishli o‘zgaruvchilar bu yerda saqlanadi. Bu modul yordamida kod ichida turli sozlamalarni alohida jamlab, ularni markazlashtirilgan holda boshqarish osonlashadi.

6. **logs/** – Tizim ishlashi jarayonida hosil bo‘ladigan log-fayllar saqlanadigan katalog. Flask server loglari, xatoliklar stack trace’lari yoki foydalanuvchi so‘rovlari tarixi kabi ma’lumotlar bu yerga yozilishi mumkin. Loglar asosida keyinchalik monitoring va xatolarni debug qilish amaliyoti olib boriladi.

7. **tests/** – Test skriptlari va test ma’lumotlari joylashgan bo‘lim. Har bir modul uchun alohida test fayllari (masalan, **test_preprocessing.py, test_api.py**) tuziladi. Unittest yoki PyTest yordamida yozilgan avtomatlashtirilgan testlar tizim funksiyalarini tekshiradi. Bu katalogda shuningdek test uchun kerakli sun’iy ma’lumotlar yoki fixture-fayllar saqlanishi mumkin.

Yuqoridagi tuzilma *modullarning bir-biridan mustaqilligini ta’minlaydi* va kodni tushunishni osonlashtiradi. Masalan, agar kelgusida yangi model sinab ko‘rilmoqchi bo‘lsa, **classifier/** katalogida o‘zgartirishlar qilinadi, frontend qismi tegilmagan holda qolaveradi. Shu tarzda **MVC (Model-View-Controller)** prinsipiiga o‘xshash qatlamlarga bo‘lingan strukturaga erishiladi:

1) **classifier** – *model (Model) qismi,*

2) **api** – *controller (xizmat) qismi, visualization,*

3) **UI kodi** – *view (taqdimot) qismi vazifasini bajaradi.*

Bunday kataloglar strukturasini qo‘llash jamoaviy ishlashda ham qulay bo‘lib, har bir dasturchi tegishli modul ustida mustaqil ishlashi va versiyalarni boshqarish tizimida to‘qnashuvlar minimal bo‘lishini ta’minlaydi.

Sentiment tahlil axborot tizimining dasturiy ta’minot tarkibi modular tarkibda tashkil etilishi zarur. Bu degani, tizimni amalga oshiruvchi dasturiy kod aniq vazifalar bo‘yicha bo‘lingan bo‘lib, har bir modul mustaqil ravishda o‘z funksiyasini bajaradi va kerak bo‘lganda boshqa modullar bilan o‘zaro aloqaga kirishadi. Ilmiy ishlar tahliliga ko‘ra, bunday modul tuzilmasi nafaqat kodni tushunishni osonlashtiradi, balki uni qo‘llab-quvvatlash va kengaytirishni ham yengillashtiradi. Masalan, matnlar bazasini yuklash, matnni oldindan qayta ishlash, klassifikatsiya modelini tayyorlash va yangi matnlar uchun bashorat qilish kabi bosqichlar har biri alohida kod moduli sifatida tashkil etilishi tavsiya etiladi.

Sentiment tahlil tizimi to‘rtta asosiy modulga ajratilgan:

1) Sharhlar datasetsini yuklash va saqlash;

2) Matnlarni tozalash va tayyorlash (kichik harflarga o‘tkazish, ortiqcha belgi va so‘zlarni olib tashlash, lemmalash);

3) Mashina o‘qitish modelini tayyorlash;

4) Sinov ma’lumotlarini modelga kiritib, natijalarni bashorat qilish.

Modular yondashuvning afzalligi shundaki, agar tizimning ma’lum qismida o‘zgarish qilish kerak bo‘lsa (klassifikator turini o‘zgartirish yoki yangi oldindan qayta ishlash qoidalarini qo‘shish), bu to‘g‘ridan-to‘g‘ri tegishli modul ichida amalga oshiriladi va bu boshqa qismlarga minimal ta’sir ko‘rsatadi.

O‘zbek tilidagi sentiment tahlil tizimlarini yaratish tajribasida ham kod strukturasi masalasiga alohida e’tibor berildi. Yuqorida zikr etilgan Django asosidagi tizimda kod MVT arxitektura printsipiga muvofiq qatlamlarga ajratilgan: models.py fayllarida ma’lumotlar bazasi va til modellari bilan ishlovchi sinflar, views.py da biznes mantiq (matnni qayta ishlash, modelni chaqirish va h.k.), templates/ jildida esa foydalanuvchi interfeysi uchun HTML shakllari joylashtirilgan. Shuningdek, mustaqil NLP funksiyalarini (masalan, to‘plamni tozalash, so‘zlarni ajratish) alohida kutubxona yoki yordamchi modullarga joylashtirish amaliyoti kuzatiladi. Bu esa kodning qayta ishlatiluvchanligini oshiradi va bir xil funksiyani tizimning turli joyida takror yozish o‘rniga, bitta modulni chaqirish kifoya bo‘ladi. Tizimning dasturiy strukturasini lozim darajada modular tashkil etish uning sifatini belgilovchi muhim omillardandir. Zamonaviy dasturlashda keng tarqalgan MVC/MVT arxitektura tamoyillari va obyektga yo‘naltirilgan dasturlash uslubi bunday kod tuzilmasini yaratishda asosiy vosita bo‘lib xizmat qiladi.

Sentiment tahlil axborot tizimining texnik tavsifi (texnologiyalar va arxitektura tanlovi)

Ushbu sentiment tahlil tizimini yaratishda zamonaviy va ochiq kodli texnologiyalar majmuasi tanlandi. Asosiy dasturlash tili – Python, chunki Python tabiiy tilni qayta ishlash (NLP) va mashinaviy o‘rganish uchun boy kutubxonalarga ega. ML model sifatida Transformers kutubxonasidan foydalanildi va bu Hugging Face kompaniyasi tomonidan taklif qilinadigan, hozirgi zamon NLP muammolari uchun eng ilg‘or oldindan o‘qitilgan modellar to‘plamini oson qo‘llash imkonini beruvchi kutubxonadir. Transformers kutubxonasi yordamida biz o‘zbek matnlari uchun tayyorlangan BERT modelini yoki shunga o‘xshash ko‘p tilli modelni ishlatish mumkin. Oldindan tayyor modeldan foydalanish hisoblash xarajatlarini kamaytiradi va modelni noldan o‘qitish uchun ketadigan vaqt va resurslarni tejalishiga olib keladi [13]. Shu tariqa, juda kam ma’lumot bilan ham yuqori sifatli sentiment aniqlovchi modelni integratsiya qilish imkoni bo‘ldi.

Backend qismi uchun Flask ishlatilgani bejiz emas. Flask Python’da yozilgan veb-freymvork bo‘lib, u RESTful API yaratishda juda qulay va tez natija beruvchi vositadir. Flask yordamida biz tezkorlik bilan zarur endpointlarni yo‘lga qo‘yib, ularni ML modeli bilan bog‘landi. Flask’ning soddaligi va minimalizmi loyihani kichik mikroxizmat tarzida tuzishda qo‘l keldi. Shuningdek, Flask’ga kerak bo‘lsa boshqa kutubxonalar (masalan, Flask-CORS orqali frontend bilan cross-origin aloqani yo‘lga qo‘yish) oson ulanishi va konfiguratsiya qilinishi ham tanlov uchun sabab bo‘ldi.

Frontend qismi React yordamida ishlab chiqildi. React – bu komponentga yo‘naltirilgan JavaScript kutubxonasi bo‘lib, Single Page Application (SPA) yaratish uchun juda mos. React orqali foydalanuvchi interfeysi dinamik va tezkor bo‘ladi, chunki sahifa tarkibining o‘zgarishi faqat kerakli qismlarnigina yangilash orqali amalga oshadi. Bu esa foydalanuvchi tajribasini sezilarli

yaxshilaydi. Bundan tashqari, React yordamida loyihani mobil qurilmaga mos Progressive Web App (PWA) sifatida ham tayyorlash imkoni paydo bo‘ladi. Frontend bilan backend o‘rtasida o‘zaro aloqa AJAX/Fetch API orqali HTTP so‘rovlar yuborish orqali amalga oshiriladi. Ya’ni, React komponenti foydalanuvchi kiritgan matnni oladi va Flask API’ga JSON so‘rov yuboradi, natijani qabul qilib, foydalanuvchiga ko‘rsatadi.

Ma’lumotlar bazasi sifatida SQLServer2022 tanlandi. SQLServer kuchli MBBT bo‘lib, u ma’lumotlarning yaxlitligi va ishonchliligini ta’minlaydi, va murakkab SQL so‘rovlarini qo‘llab-quvvatlaydi. Ushbu tizimda SQLServer’dan foydalangan holda foydalanuvchilar kiritgan matnlar, ularning sentiment natijalari, va ehtimoliy statistik ma’lumotlar saqlanadi. Masalan, agar foydalanuvchi bir nechta matnlarni jo‘natgan bo‘lsa, ularning har biri uchun sentiment va sana-vaqti bilan yozuvlar bazada saqlanib, keyinchalik tahlil uchun ishlatilishi mumkin. Bazaga ulanish Flask orqali amalga oshiriladi. Tanlovimizga ko‘ra, SQLServer ko‘p foydalanuvchili muhitda ham barqaror ishlashi, tranzaksiyalarni qo‘llab-quvvatlashi va JSON kabi ma’lumot turlarini ham saqlay olishi sabab loyihamiz uchun ayni muddao bo‘ldi.

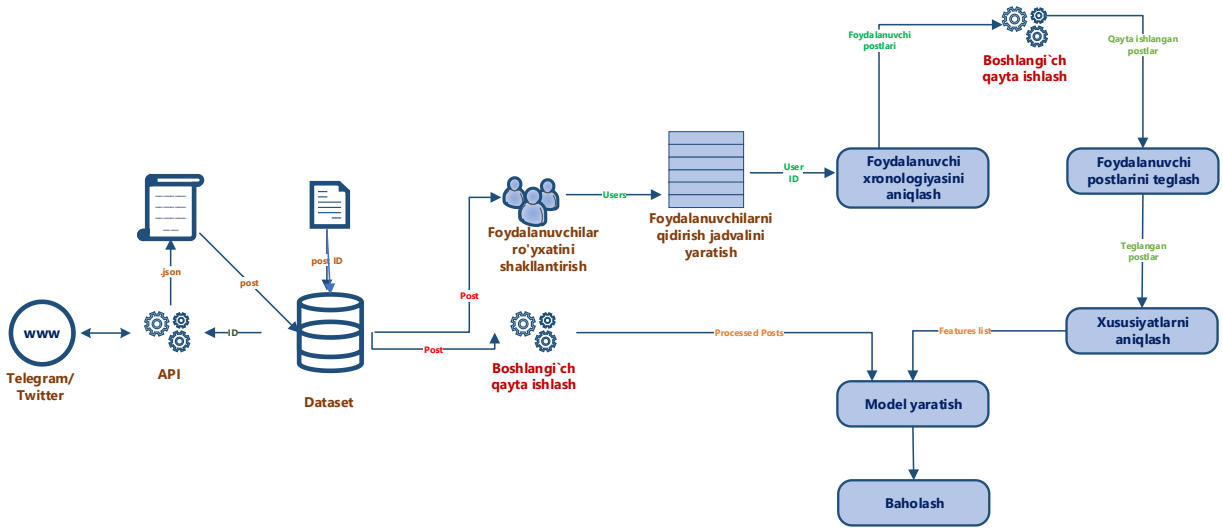
Docker konteyner texnologiyasi tizimni har xil muhitlarda bir xilda ishlashini ta’minlash va deployment (yuklash) jarayonini soddalashtirish uchun joriy qilindi. Docker yordamida bizning frontend (React) va backend (Flask) qismlarimiz alohida konteynerlarda qadoqlanib, bir xil konfiguratsiya bilan har qanday serverga o‘rnatilishi mumkin. Bu jarayon dasturchilar o‘rtasida muhitalarning uyg‘unligini ta’minlaydi. Docker konteynerlari **microservice arxitekturasi**ga o‘tish uchun ham zamin yaratadi, chunki kelgusida tizim modullarini alohida konteynerlarga ajratib, ularni orkestratsiya (Kubernetes) yordamida boshqarish mumkin bo‘ladi.

Yuqoridagi texnologik yechimlar zamonaviy tizim muhandisligi tamoyillariga mos ravishda tanlangan. Masalan, uch qavatli arxitektura (client-server-DB) tanlash orqali sentiment tahlil tizimi xavfsizligini oshirildi. Chunki mijoz (frontend) to‘g‘ridan-to‘g‘ri ma’lumotlar bazasiga ulanmaydi, faqatgina oraliq API orqali murojaat qiladi [12]. Bu esa ma’lumotlar qatlamini himoya qilgan holda, biznes mantiqni prezentatsiyadan ajratish imkonini berdi. Shuningdek, har bir komponent mustaqil bo‘lgani tufayli, masalan, yuk ko‘payganda faqat model serverini alohida kuchaytirish (scaling) yoki biror qismda nosozlik bo‘lsa, boshqa qismlarga ta’sirini minimallashtirish imkoni tug‘ildi.

Arxitekturani tanlashda loyihaning hajmi va murakkabligi ham inobatga olindi. Monolit yondashuv (barcha funksiyalarni bitta ilovaga jamlash) kichik loyihalarda qulay bo‘lsa-da, kelgusida tizimni rivojlantirish va yangilashda qiyinchilik tug‘dirishi mumkin. Shu bois biz **modullararo aniq interfeyslar (API)** bilan bo‘lingan xizmatlar arxitekturasini ma’qul ko‘rdik. Bu yondashuv hozirgi kunda mikroxizmatlar arxitekturasi tarafdorlari orasida keng qo‘llanadi va katta tizimlar uchun o‘zining samaradorligini isbotlagan. Bizning holatimizda tizim nisbatan kichik bo‘lsa-da, modular arxitektura tatbiq etildi. Natijada loyiha ustida bir nechta dasturchi parallel ishlaganda ular bir-biriga xalaqit bermaydi, har bir qismni alohida sinab, alohida joyga joriy etish mumkin bo‘ladi.

Umuman olganda, tanlangan texnologiyalar to‘plami **Python+Flask+Transformers+SQLServer+React+Docker** loyihani tez va samarali rivojlantirish, ham ilmiy jihatdan ilg‘or yondashuvlarni (masalan, Transformer modelidan foydalanish) qo‘llash, ham amaliy jihatdan mustahkam yechim qurishga xizmat qildi. Bunda har bir texnologiyaning o‘z o‘rni va asosli tanlash sabablari bor: Python – NLP va ML uchun, Flask – yengil va tezkor API xizmati uchun, Transformers – sifatli matn tahlili modeli uchun, SQLServer – ishonchli

ma’lumotlar saqlash uchun, Docker – portable (ko‘chma) va uyg‘un ishlash muhiti uchun, React – zamonaviy va foydalanuvchiga qulay interfeys uchun. Ushbu uyg‘unlik ilmiy yondashuv (modelning aniqligi, algoritmlarning to‘g‘riligi) va amaliy yondashuvni (tizimning ishlashi, arxitekturaviy yechimlar) birlashtirishga imkon berdi. Quyida ijtimoiy tarmoq foydalanuvchilari post(xabar)larini sentiment tahlil qilish tizimi arxitrkturasi keltirilgan.



3-rasm. Ijtimoiy tarmoq foydalanuvchilari postlarini sentiment tahlil qilish axborot tizimi arxitrkturasi

Quyidagi 2-jadvalda sentiment tahlil axborot tizimining texnik tavsifi keltirilgan.

2-jadval. Sentiment tahlil axborot tizimining texnik tavsifi

№	Komponent / Tavsif	Tanlangan texnologiya	Tavsif / tanlash sababi
1.	Dasturlash tili	Python	NLP va mashinaviy o‘qitish uchun kuchli ekotizim, qulay va tez ishlab chiqish
2.	NLP modeli	Transformers (BERT/XLM-RoBERTa)	Zamonaviy NLP vazifalar uchun oldindan tayyorlangan modellar, tez integratsiya qilish imkoniyati
3.	Backend freymvorki	Flask	Yengil vaznli REST API uchun optimallashtirilgan, tezkor va sodda
4.	Frontend texnologiyasi	React	Komponentlarga asoslangan, dinamik va qulay foydalanuvchi interfeysi, SPA va PWA ishlab chiqish imkoniyati
5.	Ma’lumotlar bazasi	SQLServer	Ishonchlilik, tranzaksiyalarni qo‘llash, murakkab so‘rovlarni qo‘llab-quvvatlash va JSON ma’lumotlarini saqlash
6.	Konteynerlashtirish	Docker	Tizimni istalgan muhitda oson joylashtirish va bir xil konfiguratsiya bilan ishlash imkoniyati, mikroservislar uchun asos yaratish

7.	Arxitektura	Uch qavatli va Mikroservislar	Modularlik, kengayish imkoniyatlari, yuklamani alohida boshqarish va xatolarning ta’sirini kamaytirish
8.	API protokoli	REST	Frontend va backend o‘rtasida sodda, keng tarqalgan va moslashuvchan aloqa o‘rnatish
9.	Modelni o‘qitish muhiti	PyTorch / HuggingFace	Chuqur o‘rganish modellari uchun moslashuvchan, o‘zbek tiliga mos pre-trained modellardan foydalanish imkoniyati
10.	Frontend-backend integratsiya	Axios / Fetch API	REST API orqali JSON formatida frontend-backend o‘zaro aloqa qilish imkonini beradi

Ushbu tahliliy jadval sentiment tahlil axborot tizimi uchun tanlangan texnologiyalar va arxitekturaviy yechimlarni aniq ko‘rsatib beradi va tizimning samarali ishlashini ta’minlash uchun tanlovlarning asosini tavsiflaydi.

Foydalanuvchi interfeysi (UI)

Tizimning foydalanuvchi interfeysi oddiy va intuitiv tarzda ishlab chiqilgan. Maqsad shuki, foydalanuvchi ilk bor tizimga murojaat qilganidayoq qanday amallarni bajarish kerakligini oson tushunilishi kerak. Go‘yo ilgari ham ushbu interfeysdan foydalangandek “o‘zini uydagidek his qilsin” [14]. Buning uchun interfeys dizaynida bir nechta prinsiplarga amal qilindi: **minimalizm** (keraksiz elementlarni chiqarib tashlash), **mantiqiy guruhlash** va **ko‘rinuvchi navigatsiya**. Asosiy sahifa strukturasi foydalanuvchiga darhol tanlash uchun ikki xil imkoniyatni taqdim etadi: *yakka matnni tahlil qilish* yoki *fayl (ko‘p matnlarni) tahlil qilish* rejimi. Interfeysning bosh ekrani yuqori qismida loyiha nomi va menyu (asosiy bo‘limlarga havolalar: bosh sahifa, loyiha haqida ma’lumot, aloqa) joylashgan. Markaziy qismda foydalanuvchiga tanlov beriladi: u to‘g‘ridan-to‘g‘ri matn kiritishi yoki oldindan tayyorlangan CSV faylini yuklab, bir nechta matnni birdaniga tahlil qilishi mumkin. Senrtiment axborot tizimi bir vaqtning o‘zida bir nechta matnni baholab, natijalarni jadval shaklida taqdim eta oladi. Jadvalda har bir matn uchun asl matnning o‘zi, oldindan tozalangan ko‘rinishi, aniqlangan sentiment kategoriyasi va (agar mavjud bo‘lsa) sana-vaqt kabi ma’lumotlar ko‘rsatiladi. Interfeysda natijalarni filtrlash imkoniyati ham mavjud: foydalanuvchi jadval ustidagi filtrdan foydalanib, faqat ijobiy yoki faqat salbiy yozuvlarni alohida ajratib ko‘rishi mumkin. Bu kabi funksionallik foydalanuvchiga ko‘p sonli natijalar orasida keraklisini tez topishga yordam beradi. Jadval tarzidagi chiqarishdan tashqari, yakka matn tahliliga mo‘ljallangan rejimda natija alohida blok sifatida ko‘rsatiladi.

Navigatsiya va ko‘rinish jihatidan, interfeysning barcha elementlari bir-biridan aniq ajratilgan va belgilangan. Masalan, matn kiritish maydoni yonida **“Tahlil qilish”** tugmasi joylashgan bo‘lib, u bosilganda foydalanuvchi tanlagan rejimga qarab tegishli harakatni amalga oshiradi (yakka matn uchun API chaqiriladi yoki fayl yuklangan bo‘lsa, fayl serverga jo‘natiladi). Agar tizimda foydalanuvchi ro‘yxatdan o‘tishi yoki tizimga kirishi talab etilsa, interfeysning yuqori o‘ng qismida *“Kirish”* yoki *“Ro‘yxatdan o‘tish”* tugmalari ko‘zda tutiladi. Biroq, ushbu sentiment tahlil tizimi odatda ochiq foydalanish uchun mo‘ljallanganligi sababli, autentifikatsiya majburiy emas va asosiy e’tibor funksiyaga qaratiladi.

Grafik dizaynda ranglar va shriftlar foydalanuvchi ko‘zini qulay qabul qiladigan tarzda tanlandi. Asosan neytral ochiq ranglar foni va kontrastli matn ranglari ishlatildi. Natija chiqarishda ijobiy, neytral va salbiy holatlar uchun mos ravishda yashil, kulrang va qizil belgi yoki matn ranglari orqali vizual ajratish joriy qilindi. Bu foydalanuvchiga matn kayfiyatini darhol rangdan bilib olishga yordam beradi. Masalan, agar natija *“ijobiy”* bo‘lsa, *“ijobiy”* so‘zi yashil bilan yoziladi, salbiy bo‘lsa qizil va hokazo – bu intuitiv rang kodi inson ongida tezroq idrok etiladi. Interfeysni loyihalashda responsivlik ham inobatga olingan. Ya’ni, sayt turli ekran o‘lchamlarida to‘g‘ri va qulay ko‘rinishda qoladi. Buning uchun CSS Flexbox/Grid texnologiyalaridan foydalanilib, komponentlarning joylashuvi mobil qurilmalarda vertikal tartibga o‘tishi, keng ekranda esa yonma-yon joylashishi ta’minlandi. Shuningdek, agar foydalanuvchi smartfon yoki planshetdan kiritish jarayonida xatoga yo‘l qo‘ysa (matn kiritmay “Tahlil” tugmasini bosib yuborsa), interfeys buning oldini olib, mos ravishda ogohlantiruvchi xabar chiqish (validation) imkoniyatlariga ega.

Demo sahifa tuzilmasini qisqacha tasvirlaydigan bo‘lsak, u quyidagicha: yuqorida sayt sarlavhasi va menyu, markazda chapda *“CSV yuklash”* bloki va o‘ngda *“Matn kiritish”* bloki (yoki ketma-ket tartibda, ekran kengligiga qarab), pastroqda esa natija paneli joylashgan bo‘ladi. Agar foydalanuvchi biror rejimni tanlamay to‘g‘ridan-to‘g‘ri matn kiritishni boshlasa, tizim avtomatik ravishda shu rejimga o‘tadi. Interfeys foydalanuvchining odatiy xatti-harakatlarini oldindan hisobga olgan holda ishlab chiqilgan. Matn kiritish maydoniga biror matn yozilishi bilan *“Tahlil qilish”* tugmasi faol (enabled) holatga o‘tadi. Shu tariqa har bir elementning holati foydalanuvchi harakatiga javoban moslanadi. Umuman olganda, foydalanuvchi interfeysi zamonaviy veb-ilova dizayn prinsiplari asosida yaratilgan bo‘lib, yuklash tezligi, interaktivlik va soddalikka alohida e’tibor berilgan. Intuitiv interfeys ortida jiddiy dizayn qarorlari yotadi. Ya’ni, foydalanuvchi uchun *“hammasi o‘z-o‘zidan tushunarli”* bo‘lishi uchun dizayner va dasturchilar oldindan puxta mehnat qilishgan. Natijada, sentiment axborot tizimdan foydalanish jarayoni foydalanuvchi uchun iloji boricha qulay va samarali tashkil etildi: u bir necha oddiy qadam bilan o‘zbek tilidagi matnlarning sentimentini aniqlashi va natijalarni ko‘rishi mumkin. Ushbu maqolada keltirilgan yechimlar va tavsiflar ilmiy yondashuv (UML diagrammalar, algoritmlar) va amaliy tajriba (texnologik tanlov va tuzilma) uyg‘unlashgan holda, zamonaviy tizim muhandisligi tamoyillari asosida keltirildi.

Xulosa

Ushbu maqolada sentiment tahlil vazifasini amalga oshiruvchi zamonaviy axborot tizimi arxitekturasining barcha qatlamlari – frontend, backend, ma’lumotlar bazasi, ML modeli – ilmiy asosda tahlil qilindi va ularning o‘zaro integratsiyasi aniqlashtirildi. Taklif etilgan arxitektura RESTful mikroservislar asosida qurilgan bo‘lib, modulga asoslangan dizayn tizimning kengayuvchanligini va xatolarga chidamliligini ta’minlaydi. XLM-RoBERTa modeli yordamida sentiment aniqligi yuqori bo‘lishi (AUC $\approx$ 0.96) tizimning ilmiy asosliligi va amaliy samaradorligini isbotlaydi. Interfeys soddaligi, tizim tezligi va Docker yordamida konteynerlash texnologiyasi esa real vaqtda foydalanish qulayligini ta’minlaydi. Yagona ekotizim sifatida bu yechim o‘zbek tilida avtomatik matn tahlili uchun barqaror, masshtablanuvchan va ilmiy jihatdan asoslangan platforma yaratishga xizmat qiladi. Tizimning modulli arxitekturasi esa kelgusida boshqa NLP vazifalarini (masalan, named entity recognition (NER), topic modeling) integratsiyalash imkoniyatini ham beradi.

Foydalanilgan adabiyotlar

1. Suyunova, M. (2025). SENTIMENT ANALYSIS RESEARCH AND IMPORTANCE. *Journal of Applied Science and Social Science*, *1*(2), 891-897.
2. Muniraja, M., Bharadwaj, S., Marupudi, H., Santhosh, C., More, K., Valla, A. A., & Maddi, D. (2025, May). Sentiment Analysis of Customer Satisfaction Using ML Models. In *2025 Global Conference in Emerging Technology (GINOTECH)* (pp. 1-6). IEEE.
3. Das, A., Gunturi, K. S., Chandrasekhar, A., Padhi, A., & Liu, Q. (2021, December). Automated pipeline for sentiment analysis of political tweets. In *2021 international conference on data mining workshops (icdmw)* (pp. 128-135). IEEE.
4. Boltayevich, E. B., Tursunalievna, A. M., Ilxomovna, A. X., Xolmo'minovna, A. O., & Farkhodovich, K. B. (2024, February). Modeling of Models and Processes that Differentiate Semantically Polyfunctional Words in the Context of the Uzbek Language. In *International Congress on Information and Communication Technology* (pp. 421-431). Singapore: Springer Nature Singapore.
5. Garg, A., & Sharma, D. (2025). AI Insights: Navigating Education News Ethically Through Aggregation and Sentiment Analysis. *Ethical Decision-Making Using Artificial Intelligence: Challenges, Solutions and Applications*, 313-342.
6. Yadav, A., & Vishwakarma, D. K. (2020). Sentiment analysis using deep learning architectures: a review. *Artificial Intelligence Review*, *53*(6), 4335-4385.
7. Bommagani Ramesh, D. G. A STUDY ON UML DIAGRAMS FOR TOURIST PLACE REVIEW SENTIMENT ANALYSIS CLASSIFICATION USING MACHINE LEARNING.
8. Gupta, R., Sameer, S., Muppavarapu, H., Enduri, M. K., & Anamalamudi, S. (2021, September). Sentiment analysis on Zomato reviews. In *2021 13th International Conference on Computational Intelligence and Communication Networks (CICN)* (pp. 34-38). IEEE.
9. Lüftenegger, E., & Softic, S. (2020, September). Sentipromo: a sentiment analysis-enabled social business process modeling tool. In *International Conference on Business Process Management* (pp. 83-89). Cham: Springer International Publishing.
10. Söderman, O. (2025). Comparison of web application architecture styles.
11. https://amlanscloud.com/apparchitecture/
12. https://www.intellectsoft.net/blog/web-application-architecture/
13. https://huggingface.co/docs/transformers/en/index
14. https://medium.com/design-bootcamp/3-key-principles-for-creating-an-intuitive-user-interface-6189a6165134

Management and Future Technologies

journal.umft.uz