AZƏRBAYCAN RESPUBLİKASI TƏHSİL NAZİRLİYİ AZƏRBAYCAN TEXNİKİ UNİVERSİTETİ

Əlyazması hüququnda

Əliyeva Qənirə Müşfiq qızı

BANKLARDA MÜŞTƏRİ XİDMƏTİ FƏALİYYƏTİNİN QİYMƏTLƏNDİRİLMƏSİ

mövzusunda

MAGİSTRLİK DİSSERTASİYASI

İxtisas: 060509 – Kompüter elmləri
İxtisaslaşma: Kompüterli modelləşdirmə

Elmi rəhbər: r.ü.f.d., baş müəllimi İsmayılova Rəna Əşrəf qızı

BAKI-2023

MÜNDƏRİCAT

GİRİŞ

Mövzunun aktuallığı. Hər müəssisənin ön ofis işçələri, yəni xidmət mərkəzlərinin əməkdaşları çox dəqiq və peşakar olmalıdırlar. Çünki şirkətin canlı nümayəndələridir onlar. Bizim halda bu şirkətlər bank müəssisələridir, burada işçilər böyük vəsaitlər ilə əməliyyatlat icra edirlər.

Banklar öz müştərilərinə müxtəlif üsul və vasitələr ilə xidmət göstərirlər, lakin bu xidmətlərdən ən effektiv və vacib olanı, müştəri xidmətlərinin fəaliyyətidir. Müştəri xidmətlərində çalışan əməkdaşların işi, bankın birbaşa inkişafına təsir göstərir. Burada yalnız gülərüz və mehriban davranışlar ilə məsələ bitmir, lazımdır ki əməliyyatları icra edən əməkdaşlar yüksək kvalifikasiyalı bir kadr olsun. Gərgin iş şəraitində çalışan müştəri xidmətlərinin əməkdaşı həm mehriban davranış və nəzakət göstərməlidir, həmçinin yüksək keyfiyyətli xidmət ilə öz müştərisni məmnun etməlidir.

Burada işçinin fəaliyyətinə nəzarət üçün artıq tək müşahidə kameraları bəhs etmir, bank əməkdaşının sistemdə apardığı əməliyyatlarında qeydiyyatı və yoxlanılması zəruridir. Sistem üzərində kiçik və böyük məbləqlərin dövriyyəsi icra olunur, hərf və ya rəqəm səhvi, şirkətə çox böyük ziyanlar vura bilər. Əməkdaşın işinin qiymətləndirilməsi üçün xüsusi proqram təminatına ehtiyac var, və bu növ proqram təminatları artıq istifadə olunur və geniş yayılmışdır.

Mövzunun öyrənilmə vəziyyəti. Bu istiqamətdə açıq ədəbiyyatda məlumata rast gəlmək olmur, çünki proqram təminatının seçimini bank müstəqil icra edir, müxtəlif növ vasitələr və alətlər mövcuddur. Məsələn: Kibrit, Orangeline, PerfomancePro, Terrasoft, Qualtrics, Trello, Salesforce və digərləri. Bank öz göstərdiyi xidmətlərdən asılı olaraq lazım olan proqram təminatından istifadə edir. İstifadə olunan alətlər haqqında yalnız daxili sənədlərdə qeydlər aparılır.

Tədqiqat işinin predmeti. Tədqiqat işinin predmetinin əsasında banklarda müştəri xidməti fəaliyyətinin qiymətləndirilməsi üsulunun tədqiqi dayanır.

Tədqiqat işinin obyekti. Biz işimizdə Azərbaycan banklarında ən geniş yayılmış və daha çox istifadə olunan Orangeline proqram təminatının üzərində işləyəcəyik.

Tədqiqat işinin metodu. Uzun illərdir Orangeline platforması üzərindən müxtəlif xidmətlər təqdim olunur. Orangeline Java proqramlaşdırma dilində yazılmışdır və geniş Java Spring alətlər dəsti mövcuddur. Lakin buna baxmayaraq, araşdırmalarımız nəticəsində sistemdə çatışmayan və müştəri xidmətlərinin əməkdaşının işinin qiymətləndirilməsində müsbət təsir göstərə biləcək catışmamazlıqlar aşkarladıq. Məhs bu çatışmayan məqamların aradan qaldırılması istiqamətində işlər apardıq, Hibernate, Apache Struts və Apache Tiles vasitələrin inteqrasiyası ilə yeni metod təklif etdik, Java proqramlaşdırma dilində addımlar yazdıq. Java proqramlaşdırma dilində işçinin fəaliyyətinin çoxfunksiyalı qiymətləndirilməsi üçün xüsusi kod hazırlanmışdır.

Tədqiqat işinin məqsədi və vəzifələri. Ümumilikdə banklarda xidmət mərkəzi əməkdaşının fəaliyyətin çox parametrli qiymətləndirilməsi üçün Orangeline mühitinin təkminləşdirilməsi təklif olunmuşdur.

I FƏSİL. BANKLARIN MÜŞTƏRİ XİDMƏTİ FƏALİYYƏTİ

1.1. Banklarda müştəri xidməti fəaliyyətinin əsas prinsipləri və funksiyaları

Bank xidməti kommersiya bankı ilə müştəriləri arasında əlaqədir. Bank xidməti əhali üçün vacibdir. Beləliklə, getdikcə daha çox vətəndaş müəyyən xidmətlər almaq üçün banklara müraciət edir. Müştəri bankın xidmətlərindən istifadə edən və ya məhsullarını satın alan bir şəxsdir. İstənilən şəxs bankın müştərisi ola bilər: başqa bank, hüquqi şəxs (təşkilat, müəssisə və s.), fiziki şəxs.

Bank müştərilərinin təsnifatı.

Bank müştərilərinin hansı kriteriyalara görə neçə təsnifata ayırır, onları meyarlarını nəzərdən keçirək:

1. Hüquqi status. Bu meyara görə müştərilər hüquqi və fiziki şəxslərə bölünür. hüquqi şəxslər iqtisadiyyatın, böyük, orta və kiçik müəssisələrin sənaye və sektorlarının nümayəndələridir. Fiziki şəxslər vətəndaşlardır.

2. Reallıq. Bu meyar bankın bütün müştərilərini real (bankın müştəri bazasında olanlar) və potensial müştərilərə bölür. Real müştərilərlə bank artıq iqtisadi münasibətdədir və potensial müştərilər hələ də real ola bilərlər.

3. Ölçü. Bankın müştəriləri də ölçülərinə görə bölünürlər, yəni, böyük, orta və kiçik ola bilərlər.

4. Bank xidmətinin başlanğıcı. Burada müştərilər iki növə bölünür – köhnə və yeni. Köhnə müştərilər uzun müddətdir bankla əməkdaşlıq edənlərdir, yeniləri isə əməkdaşlığa yeni başlayanlardır.

5. Kredit qabiliyyətinin səviyyəsi. Kommersiya bankları müştərilərinin kredit səviyyəsini ölçmək üçün beş sinifli miqyasdan istifadə edirlər.

Bunlar bankların müştərilərini təsnif etdiyi əsas meyarlardır.

Bank-müştəri münasibətlərinin prinsipləri.

Bank xidməti bankla müştəri arasındakı münasibətdir, bununla əlaqədar olaraq kommersiya bankı ilə müştərilər arasında münasibətlərin əsas prinsipləri formalaşmış, müştəri xidməti üzərində qurulmuşdur:

1. Qarşılıqlı maraq prinsipi. Bu prinsip o deməkdir ki, kompromislər, güzəştlər sayəsində yaxşı münasibətlər qorunub saxlanıla bilər.

2. Rasional fəaliyyət prinsipi. Müştəri üçün banka müraciət etmək, fəaliyyətlərini rasional təşkil etməyin yollarından biridir.

3. Likvidliyin təmin edilməsi prinsipi. Bu vəziyyətdə həm bank, həm də müştəri likvidliyini qorumağa ümid edirlər.

4. Öhdəlik prinsipi. Qarşılıqlı razılaşmaların məcburi icrası.

5. Məsuliyyət prinsipi. Hər iki tərəf bir-birinə qarşı birgə məsuliyyət daşıyır.

6. Müdaxilə etməmək prinsipi. Bu prinsip hər iki tərəfin bir-birindən müqavilələrində, müqavilələrində göstərilməyənləri tələb edə bilməməsini təmin edir.

7. Tərəfdaşlıq prinsipi. Həm bank, həm də müştəri bir-birinə tərəfdaş kimi davranmalıdır.

8. Fərqləndirmə prinsipi. Bu prinsip müştərinin fərdi xüsusiyyətlərinə yönəldilmişdir.

Xüsusi proqramlar sayəsində bank xidmətləri birbaşa bankın ofisində və ya uzaqdan həyata keçirilə bilər.

Müştəri xidməti prinsipləri.

Keyfiyyətli müştəri xidməti aşağıdakı prinsiplərə əsaslanır:

1. Baxım rahatlığı. Müştəri xidməti üçün otaq müəyyən zonalara bölünməlidir. Hər bir zona öz əməliyyatını həyata keçirməklə yanaşı, həmçinin müəyyən bir müştəri növünə xidmət etmək üçün məsuliyyət daşıyır. Məsələn, hüquqi və fiziki şəxslərə ayrıca xidmət, kredit əməliyyatları və ya kassa xidmətləri. Məsləhətləşmələr üçün ayrı bir sahə olmalıdır.

2. Uyğun ixtisas müştərilərə xidmət edən menecer. Menecer müştəri ilə bank arasında vasitəçidir. Müştəridə təkcə özü və xidmət haqqında deyil, həm də bütövlükdə bank haqqında təəssürat yaradan menecerdir.

3. Bank işçilərinin işində peşəkarlıq. Menecer tərəfindən bankın xidmət göstərilməsi, əməliyyatların aparılması və s. ilə bağlı göstərişlərinə ciddi riayət edilməsi.

4. İşdə səmərəlilik. Bankdakı növbələr həmişə müştəriləri uzaqlaşdırır. Buna görə menecerlərin işi yaxşı əlaqələndirilmiş, operativ və keyfiyyətli olmalıdır.

1.2. Müştəri xidməti fəaliyyətinin müəyyənləşdirilməsi üçün parametrlər

Müştəri xidmətinin keyfiyyətini effektiv idarə etmək üçün, xidmətin korporativ standartlar baxımından qiymətləndirilməsinə imkan verən gizli alıcı metodu (Mystery Shopping) istisna olmaqla, real müştərilər arasında sorğu aparmaq eyni dərəcədə vacibdir. Bu, müxtəlif yollarla edilə bilər, məsələn, veb anketlərdən, poçt sorğularından istifadə etməklə, satış sahəsindəki müştəriləri sorğulayan müsahibləri cəlb etməklə. Son zamanlarda bir sorğu texnologiyası aktiv şəkildə inkişaf edir, bu da xidmətin alınması və ya alınması nöqtəsində birbaşa quraşdırılmış xüsusi düymə uzaqdan idarəetmə və ya terminalların istifadəsinə əsaslanır. Bu texnologiyanın əsas üstünlüyü ondan ibarətdir ki, müştərinin hisslərini unutmağa vaxtı yoxdur, buna görə müştərilərin xidmət keyfiyyətini qavraması barədə məlumatlar çox dəqiqdir. Bundan əlavə, müştərini alış-veriş zamanı bir suala cavab verməyə həvəsləndirmək, alış sevinci (və ya əksinə, inciklik) artıq o qədər də parlaq olmadığı bir-iki gündə bir sıra suallara cavab verməkdən daha asandır. Əlbəttə ki, bu, müştərinin mükafat aldığı panel araşdırması deyilsə. Buna görə nümunənin nümayəndəliyi və nəticədə nəticələrin etibarlılığı daha yüksəkdir.

Bununla belə, düyməli pultlardan istifadə etməklə xidmətin keyfiyyətinin qiymətləndirilməsi texnologiyası “bəyənmək və ya bəyənməmək” düymələrinin qurulması və müvafiq statistikanın toplanması ilə məhdudlaşmır. Birincisi, müştərilərin çoxu nəyisə tıklamağa həvəsli deyillər. Müştəriləri həvəsləndirmək üçün xüsusi tədbirlər görülməzsə, müştərilərin 30%-dən çoxu xidmət keyfiyyətini və ümumi əhali ilə müqayisədə bir sıra parametrlərdə (cins, təhsil, gəlir, xarakter xüsusiyyətləri və s.) yerdəyişməni qiymətləndirəcək. çox böyük olacaq. Nəticə etibarilə, əldə edilən statistika real müştəri məmnuniyyətinə uyğun gəlməyəcək. İkincisi, keyfiyyətsiz xidmətdən yaranan müştəri narazılığını heyətin nəzarətindən kənar amillərin (nəsə pozulub, uzun növbə və s.) yaratdığı narazılıqdan ayırmaq mümkün deyilsə, onda yaranan statistikanı

motivasiya sistemi ilə əlaqələndirmək olmaz, bu texnologiyanın əhatə dairəsini əhəmiyyətli dərəcədə daraldır.

Buna görə texniki vasitələrdən (düymələr, proqram təminatı, video nəzarət sistemləri və s.) əlavə olaraq xidmət keyfiyyətinin qiymətləndirilməsi texnologiyası onların tətbiqi metodologiyasını da əhatə etməlidir. Bu məqalədə bu texnikanın praktik tətbiqi müzakirə olunur, əsas diqqət müştəri xidməti keyfiyyətinin əsas göstəricilərinə, xüsusən də bu göstəricilərin necə ölçülməsinə, etibarlılığına necə nəzarət edilməsinə və necə istifadə oluna biləcəyinə yönəldilmişdir.

Müştəri xidmətinin keyfiyyətini üç komponentə bölmək olar:

Ön xətt işçilərinin iş keyfiyyəti: dostluq, peşəkarlıq, görünüş, ünsiyyət tərzi və s.;

Ofis işinin təşkili keyfiyyəti və iş proseslərinin keyfiyyəti;

Məhsul və xidmətlərin keyfiyyəti və dəyəri.

Müştəri narazılığının səbəbləri müəyyən edildikdən sonra düzəliş tədbirləri görülür. Onların məqsədi narazılığın əsas səbəblərini aradan qaldırmaqdır. Aparılan diaqnostikanın düzgünlüyü və düzəldici tədbirlərin effektivliyi xidmətin keyfiyyətinin monitorinqi mərhələsində yoxlanılır. Beləliklə, idarəetmə dövrəsi bağlanır.

1.3. Xidmət mərkəzlərinin iş prosesinə təsir göstərən amillər

İşçilərə və onların işinə təsir edən əsas amillər.

Əsasən motivasiya amilləri işçilərin işinə təsir göstərir. Motivasiya amilləri işçilərin məhsuldarlığını və buna görə də bütövlükdə biznesin uğurunu birbaşa müəyyən edən amillərdir. Bir insanın davranışı və axtardığı hədəflər bu amillərdən asılıdır, buna görə də onları öyrənmək iş kollektivini başa düşmək üçün vacibdir. Motivasiya faktorları insanların müxtəlif vəziyyətlərdə niyə bu və ya digər şəkildə davrandıqlarını izah etməyə kömək edir. Bunlar çox fərdiləşdirilmişdir və bir sıra şərtlərdən asılıdır: xüsusiyyətlər, mədəni arxa plan, dəyərlər və inanclar, təcrübə və tərbiyə, fizioloji xüsusiyyətlər və vəziyyətin konteksti. Əsas amilləri və işəgötürənin onlardan praktikada necə səmərəli istifadə edə biləcəyini düşünün.

Xidmət mərkəzlərinin işinə təsir edən xarici amillər.

Xarici amillər işçini tapşırıqları yerinə yetirməyə və ya hədəflərə çatmağa təşviq edən motivatorlardır. Xarici motivasiya mükafat axtarmaqla və ya mənfi nəticələrdən qaçınmaqla işləyir. Birbaşa insandan asılı olmayan əsas motivasiya amilləri: pul, məşğulluq zəmanəti, iş şəraiti, karyera yüksəlişi, status, tanınma, prestij, şirkətin nüfuzu.

İk idarəçiliyində işəgötürənlər cəlbedici bir iş mühiti yaratmaq üçün rəqabətli əmək haqqı ilə qeyri-maddi mükafatlar arasında bir tarazlıq tapmalıdırlar.

Daxili amillər.

Bunlar bir insanı şəxsi məmnuniyyət və ya maraqdan fəaliyyət göstərməyə təşviq edən amillərdir. Daxili motivasiya xarici mükafatlardan və nəticələrdən deyil, fəaliyyətdən, Şəxsi böyümədən və ya məqsəd hissindən qaynaqlanır. Əməyin daxili motivasiya amilləri: özünü dərk etmə, yaradıcılıq, özünü təsdiqləmə, inam, maraq, sağlamlıq, öz əhəmiyyəti hissi, şəxsi böyümə, ünsiyyətə ehtiyac.

Menecerlər bu amilləri müəyyənləşdirməli və inkişaf etdirməlidirlər. Müsbət münasibəti təşviq etmək və böyümə imkanları təklif etməklə işəgötürənlər işçilərin sədaqətini və məhsuldarlığını artıra bilərlər.

Sosial sığorta amilləri.

Təhlükəsizlik və rifah ehtiyaclarını ödəməklə motivasiyanı təsir edən bir qrup amil: tibbi sığorta, pensiya qənaət planları, ödənişli məzuniyyət, əlillik sığortası, həyat sığortası siyasətləri, işçilərə yardım proqramları, sağlamlıq proqramları.

İşəgötürənlər ən yaxşı kadrları cəlb etmək və saxlamaq və iş mühitinin ümumi keyfiyyətini yaxşılaşdırmaq üçün hərtərəfli sosial sığorta paketləri təqdim etmək imkanlarını nəzərdən keçirməlidirlər. Bu amillərlə səmərəli işləmək üçün şirkətlər işçilərinin ehtiyaclarını və gözləntilərini anlamalı və mənsubiyyət və sabitlik hissi yaratmalıdırlar.

II FƏSİL. ÖN OFİS FƏALİYYƏTİNİN QİYMƏTLƏNDİRİLMƏSİ ZAMANI İSTİFADƏ OLUNAN ALƏTƏLƏR

2.1. Müştəri xidməti keyfiyyətinin qiymətləndirilməsi üçün parametrlər və metodlar

Bankalarda müştəri xidməti keyfiyyətinin qiymətləndirilməsi dedikdə, əsas üstünlük verilən məqam ön-ofis əməkdaşıın işinin qiymətləndirilməsidir. Burada ön-ofis işçisinin fiziki, psixoloji və emosional davranışı, həmçinin sistemdə icra olunan əməliyyatlarda nəzərə alınır.

Bildiyimiz kimi, işçinin qiymətləndirlməsi üçün, audio qeydlər, video müşahidə və sistemdə proseslərin qeydləri nəzərə alınır.

Bankın ön ofisi (front-office) müştəri ilə şirkət arasında vasitəçi rolunu oynayan bir bölmədir. Sualları, ehtiyacları, iddiaları ilə banka şəxsən gələn bir müştəriyə xidmət göstərməklə əlaqəli proqram təminatı, işçi heyəti və iş proseslərini birləşdirir.

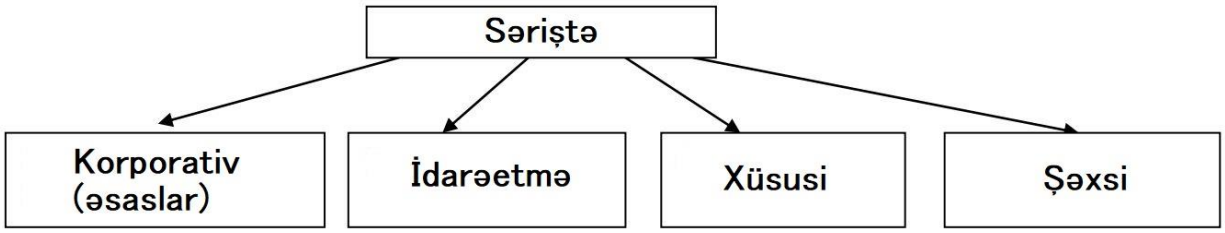
Müştəri xidməti işçilərinin bank ziyarətçiləri ilə üz-üzə işləyərkən yerinə yetirdikləri əsas vəzifələr:

* standart əməliyyatların aparılması: hesabların açılması və bağlanması, əmanətlərlə iş, bank kartlarının verilməsi, kreditlərin verilməsi və s.

* daxil olan müraciətlərin işlənməsi: konkret bank məhsulları və xidmətləri (kreditlər, yenidən maliyyələşdirmə, əmanətlər, kartlar, valyuta əməliyyatları) üzrə məsləhətləşmələr, habelə tələblərlə İş.

* satış və çarpaz satış: əsas və əlavə xidmətlərin təşviqi, əməliyyatların bağlanması.

Davranış göstəriciləri müəyyən bir səriştəyə sahib bir insanın hərəkətlərində müşahidə olunan davranış standartlarıdır. Adətən aşağıdakı səlahiyyət növləri fərqləndirirlər (Şək. 2.1).



Şəkil 2.1 İşçinin səlahiyyət növləri

Belə bir səlahiyyətlər toplusunun inkişafı şirkətdə bütün kadr idarəetmə sisteminin əsasına çevrilməli olan bir sərişdə modelinin yaradılması istiqamətində əsas addımdır. Bunun sayəsində şirkət yüksəliş, mükafatlandırma, sertifikatlaşdırma və təlim üçün alınan qiymətləndirmələrin bütün maraqlı tərəflər üçün tamamilə aydın olacağı şəffaf kadr idarəetmə sisteminə keçə biləcək.

Bir bank mütəxəssisi üçün sərişdələr modelinin hazırlanması təklif olunur, çünki bu, bankda ən çox yayılmış və tələb olunur bir məqamdır.

Bankdakı operator fiziki və hüquqi şəxslərə birbaşa xidmət göstərir və aşağıdakı funksiyaları yerinə yetirir:

- əhalinin əmanətlərinin qeydiyyatı və verilməsi;
- əmanətlər üzrə faizlərin hesablanması;
- pul köçürmələrinin qəbulu və verilməsi;
- fiziki şəxslərə cari hesabların açılması;
- bank kartları ilə iş;
- bank tərəfindən fiziki şəxslərlə aparılan müxtəlif ödənişlərin və digər əməliyyatların qeydiyyatı;
- hesablaşma sənədlərinin qəbulu və yoxlanılması

Bu və ya digər sərişdəyə olan ehtiyacı yoxlamaq üçün, Bank of Baku bankında təcrübədə olduğum zamanı, bank əməkdaşlarına növbəti sualları verdim: “yüksək uğur qazanmaq üçün operator vəzifəsinə potensial namizədin hansı keyfiyyətləri olmalıdır?”, “Bu sərişdələr işçinin davranışında necə özünü göstərəcəkdir?”, “Bu sərişdələr həqiqətən

operator vəzifəsi üçün vacibdirmi?", "İşçini müəyyən edilmiş səlahiyyətlərə görə qiymətləndirmək üçün kifayət qədər məlumat toplamaq mümkündürmü?"

Təklif olunan bütün səlahiyyət siyahılarının təhlili nəticəsində iyirmi ən vacib qabiliyyət seçildi:

1.) peşəkarlıq;

2.) müştərilərlə işləmək bacarığı;

3.) ünsiyyət bacarıqları;

4.) çalışqanlıq;

5.) analitik düşüncə;

6.) təşəbbüs, fikir yaratmaq;

7.) stresə davamlılıq;

8.) məsuliyyət;

9.) özünə inam;

10.) əzmkarlıq;

11.) yüksək öyrənmə qabiliyyəti;

12.) düşüncə çevikliyi;

13.) işin planlaşdırılması;

14.) insanlarla ünsiyyət qurmaq və ortaq bir dil tapmaq, münasibətləri idarə etmək bacarığı;

15.) öz inkişafı və həmkarlarının inkişafına kömək;

16.) Komandada işləmək bacarığı;

17.) tez qərar vermək bacarığı;

18.) məlumatların toplanması və təhlili;

19.) yüksək fəaliyyət nəticələrinə diqqət;

20.) liderlik keyfiyyətləri.

Yaradılan siyahı və qabiliyyətlərin hər biri bir daha bankın yarımbölmə rəhbərləri və qrupu rəhbərləri tərəfindən aşağıdakı məsələlər üzrə qiymətləndirilib: "bu səlahiyyət nə dərəcədə vacibdir", "bu səriştəyə məhəl qoymasanız, hansı nəticələri gözləmək olar?", "bu

sərişẗnin daimi istifadəsi hansı nəticələrə gətirib çıxaracaq?". Daha sonra, məlumatın emal ola bilməsi üçün, seçilən parametrləri 4 boyük qrupa bölürük (Cəd. 2.1).

Cədvəl 2.1 Bank mütəxəssisi üçün səlahiyyət qrupları

Sərişẗlər çoxluğu	Sərişẗlər
Təşkilatın inkişafı	Peşəkarlıq
	Müştərilərlə iş
	Təşəbbüs
İşdə yüksək nəticələr əldə etmək	Fəaliyyətin planlaşdırılması
	Qərarların qəbulu
	Nəticə yönümlü
	Analitik düşüncə
	Məlumatların toplanması və təhlili
İnsanlarla işləmək	Komanda işi
	Münasibətlərin idarə edilməsi

Beləliklə biz artıq bankın müştəri xidməti əməkdaşının keyfiyyətinin qiymətləndirilməsi üçün hansı parametrlər nəzərə alınmalı olduğunu müəyyən etdik. Qeyd edək ki, hazırda mövcud tətbiqlər belə geniş monitorinq aparıb işçini qiymətləndirmir. Sadaladığmız parametrlərin bir çoxu müxtəlif platformalarda nəzərə alınır, lakin cəm şəkildə deyil. Biz parametrlər toplusunu təyin etdik etdik.

Kadrların qiymətləndirilməsinə çoxlu yanaşmalar var, lakin onlar adətən öz diqqət mərkəzinə uyğun olaraq üç qrupa bölünürlər.

1. Keyfiyyət metodları

Metodun ikinci adı təsviridir, çünki işçiləri ciddi kəmiyyət məlumatları istifadə etmədən xarakterizə edirlər. Keyfiyyət qiymətləndirməsinə aşağıdakılar daxildir:

- Matris metodu-müəyyən bir şəxsin keyfiyyətlərini müəyyən bir vəzifə üçün ideal işçi modeli ilə müqayisə etmək.

- İxtiyari xüsusiyyətlər sistemi metodu-kadr xidməti və ya menecer işdəki ən böyük nailiyyətləri və ən ciddi pozuntuları müəyyənləşdirir və müqayisə əsasında nəticə çıxarır.
- Tapşırıqların icrasının qiymətləndirilməsi-işçinin işinin bütövlükdə qiymətləndirildiyi ən asan üsuldur.
- "360 dərəcə" metodu-işçinin həmkarları, rəhbərləri, tabeliyində olanlar, müştərilər və özləri tərəfindən qiymətləndirilməsidir.
- Qrup müzakirəsi-işçinin bu fəaliyyət sahəsindəki lider və ya mütəxəssislərlə işinin nəticələri və perspektivləri barədə söhbəti.

2. Kəmiyyət metodları

Bu metodlar ən obyektiv hesab olunur, çünki onların həyata keçirilməsinin bütün nəticələri rəqəmlərlə ifadə olunur:

- Bal qiymətləndirmə metodu. Hər bir peşəkar nailiyyət üçün işçilər müəyyən bir dövrün — ayın, rübün və ya ilin nəticələrinə görə yekunlaşdırılan müəyyən, əvvəlcədən razılaşdırılmış bal sayını alırlar.
- Dərəcə metodu. Bir qrup menecer işçilərin reytinqi kimi bir cədvəl tərtib edir, sonra bütün reytinqlər bir-biri ilə yoxlanılır və ən aşağı mövqedə olan işçilər işdən çıxarılır və ya daha az məsuliyyətli bir vəzifəyə köçürülür.
- Pulsuz bal metodu. Bu vəziyyətdə işçinin hər keyfiyyəti mütəxəssislər tərəfindən müəyyən sayda bal ilə qiymətləndirilir və nəticələr ümumiləşdirilir. Alınan faktlar əsasında reytinq tərtib edilir.

3. Kombinə edilmiş metodlar

Həm təsviri, həm də kəmiyyət aspektlərindən istifadə etdikləri üçün ən effektiv qiymətləndirmə yanaşmaları hesab olunurlar:

- Qiymətləndirmələrin cəmi metodu. Bir işçinin hər bir xarakteristikası müəyyən bir miqyasda qiymətləndirilir və sonra müəyyən bir mövqe üçün ideal ilə müqayisədə müəyyən bir ortalama göstərilir.
- Qruplaşdırma sistemi. Bütün işçilər bir neçə qrupa bölünür - qeyri-qənaətbəxş işləyənlərdən tutmuş işi praktik olaraq qüsursuz olanlara qədər.

Müəyyən bir model qurulduqda bu metodlara əsaslanmaq, işi və modelin səmərəliyini daha çox keyfiyyətli edə bilər.

2.2. Bankların müştəri xidməti fəaliyyətinin qiymətləndirilməsi üçün tətbiqlər və platformalar

Bankların avtomatlaşdırılması üçün proqramlar və onlayn xidmətlər arasında ABS (avtomatlaşdırılmış bank sistemləri), CRM, rabitə, analitik həllər, kredit avtomatlaşdırma sistemləri və s. Banklar üçün proqram nümunələri aşağıda verilmişdir.

Oracle CRM

Oracle CRM, bugünkü bazarda ən məşhur və etibarlı CRM vasitələrindən biridir. Oracle CRM Sizə müasir müştəri xidməti üçün tam, inteqrasiya olunmuş və genişləndirilə bilən proqramlar dəsti təqdim edir.

Oracle CRM sizə marketinq, satış, ticarət, sosial platformalar, xidmət və CPQ üçün etibarlı həllər təqdim edir. Etibarlıdır və bir çox yerləşdirmə modelinə malikdir. Müştərilərlə yaxşı və sağlam münasibətlər qurmağa kömək edir.

Zoho CRM

Zoho CRM 15 ildən artıq bazarda olan və ölçüsündən asılı olmayaraq kiçik və orta biznes, korporativ müştərilər və müxtəlif növ bizneslər üçün nəzərdə tutulmuş bulud əsaslı müştəri münasibətlərinin idarə edilməsi sistemidir. Bu, sadə satış boru kəməri və ya aparıcı idarəetmə alətindən kənara çıxan əməliyyat biznes platformasıdır.

Zoho-ya 180 ölkədə 250.000-dən çox şirkət etibar edir. Bu, mobil cihazları dəstəkləyən və 500-dən çox məşhur biznes proqramı ilə inteqrasiya edən 40-dan çox yerli biznes proqramı ilə sənayedə yeganə provayderdir.

Zoho CRM 2020 PCMag Redaktorun Seçimi Mükafatının və 2019 Biznes Seçimi Mükafatının (müsbət NPS reytinqinə malik yeganə satıcı olduğuna görə) laureatıdır və onu bütün dünyada istifadəçilər və tənqidçilər tərəfindən ən çox tövsiyə edilən CRM edir.

İnvestGlass.com

Bu yaxınlarda CAPGEMİNİ mükafat qazanan , InvestGlass.com -bu, daha sürətli və daha çox aktuallıqla sövdələşməyə üstünlük verən bank komandaları üçün yaradılmış CRM-dir. Invest Glass, bulud aktı asılılığı olmayan yeganə İsveçrə CRM-dir. Rəqəmsal qeydiyyat, KYC avtomatlaşdırılması, marketinq avtomatlaşdırma vasitəsi, portfel idarəetməsi və iş axını avtomatlaşdırılması kimi əsaslardan faydalanan açıq ekosistem əsasında CRM növüdür.

Burada, müştərilərlə qarşılıqlı əlaqə qurarkən, təşviq etmək istədiyiniz son təkliflərin uyğunluğu nəzərə alınmaqla prioritetlər qoyulur. Marketinq alətləri hər bir müraciəti fərdiləşdirməyə kömək edir.

Hazır şablonlar bankın müştəri münasibətlərinə xüsusi olaraq uyğunlaşdırılmışdır və marketinq avtomatlaşdırması müştərilərinizlə doğru zamanda, düzgün məlumatla və düzgün şəkildə əlaqə qurmağınızdan əmin olmaq üçün İsveçrə dəqiqliyi ilə təkmilləşdirilmişdir. Bu sadə, lakin kifayət qədər mürəkkəb müştəri münasibətləri idarəetmə proqramını API ilə birləşdirən +30 Fintech tərəfdaşının ekosistemidir.

Monday.com

CRM - Monday. com, komandaların imkanları idarə etməsinə kömək edəcək bir vasitədir. CRM əvvəlcədən hazırlanmış şablonlar təklif edir və bir çox xüsusiyyətləri olan bazarlar üçün əsl inteqrasiya olunmuş bir vasitədir.

Monday.com-dan istifadə edərək siz dərhal müştəri məlumatlarını və satışları avtomatlaşdırmaq üçün lazım olan hər şeylə bulud əsaslı CRM-i tapacaqsınız. Monday.com həmçinin mobil proqramlar və əlbəttə ki, sənayedə ən yaxşı layihə idarəetmə alətləri axtaran kiçik bizneslər üçün mükəmməldir.

Salesforce Sales Cloud

Sales Cloud müştəri davranışını təhlil edəcək, potensial müştəriləri idarə edəcək və qiymətləndirəcək, ən yaxşı imkanları təqdim edən müəyyən fəaliyyətlərə üstünlük verəcək və yüksək səviyyəli müştəri ünsiyyətini təmin etmək üçün nümayəndələriniz

üçün fərdi iş axınları yaradacaq. Bu, maliyyə xidmətləri və sığorta üçün mürəkkəb şablonlara malik ən müasir bankçılıq CRM proqramıdır.

Bir sözlə, Sales Cloud AI nümayəndələrinizə nəyi və nə vaxt edəcəyinizi söyləyir və ən yaxşı fürsəti təmsil etdiyinə əsaslanaraq müştərilərlə qarşılıqlı əlaqəni təşkil edir.

Sales Cloud CRM hesabat vermək üçün çox əlverişlidir və müasir mobil təcrübə təklif edir.

Creatio

Creatio CRM platforması həm CRM, həm də biznes proseslərinin idarə edilməsi (BPM) üçün aşağı kodlu platformadır.

Biznes proseslərinin idarə edilməsi (BPM) biznes proseslərinin modelləşdirilməsi, təhlili, təkmilləşdirilməsi, optimallaşdırılması və avtomatlaşdırılmasını əhatə edən əməliyyatların idarə edilməsi vasitəsidir.

Qeyd olunan platformalar, geniş yayılmışdır, lakin hərəsinin çatışmayan cəhətləri vardır.

2.3. CRM-in iş prinsipi və xüsusiyyətləri

Müştərilərlə qarşılıqlı münasibətlərin idarəolunması sistemi (CRM - CRM system - Customer Relationship Management) şirkətin hazırkı və potensial gələcək müştərilər ilə əlaqələrinin idarə edilməsi üçün bir yanaşmadır. Bu yanaşmada şirkət ilə müştərilərin əlaqələri tarixi haqqında verilənlər bazası təhlil edilməsinə və müştərilər ilə biznes əlaqələrini inkişaf etdirilməsinə çalışılır və əsasən müştərilərin əldə saxlanılmasına və şirkətin gəlirlərinin artırılmasına önəm verilir.

CRM yanaşmasının mühüm bir aspekti şirkətin internet saytı, telefonu, email ünvanı, canlı yayım, marketinq materialları və son zamanlarda çox istifadə olunan sosial media vasitələri daxil olmaqla müxtəlif rabitə kanalları ilə məlumatları tərtib edən CRM sistemləridir. CRM yanaşması və onu asanlaşdırmaq üçün istifadə olunan sistemlər vasitəsilə biznes müəsssisələri öz hədəf auditoriyaları və onların ehtiyaclarını ən ayxşı yolla necə qarşılayacağı barəsində məlumat alırlar. Lakin

CRM yanaşmasını qəbul etmək həmçinin istehlakçı auditoriyasında favoritizmə də gətirib çıxara bilər ki, bu da müştərilər arasında narazılığa səbəb olacaqdır və CRM-in məqsədlərini məğlub edəcəkdir.

Əməliyyat sistemi

Müştəri ilə Münasibətlərin İdarə Edilməsi sisteminin ən əsas məqsədi satış, marketinq və müştərilərə dəstəyin inteqrasiyası və avtomatlaşdırılmasıdır. Buna görə də, bu sistemlər adətən şirkətin sahib olduğu hər bir müştərisi üçün yaradılan bir səhifədə üç funksiyanın ümumi məlumatlarını təqdim edən alətlər panelinə malik olur. Alətlər paneli müştəri və şirkət arasında münasibətləri ümumiləşdirən müştəri məlumatları, keçmiş satışlar, əvvəlki marketinq səyləri və s. kimi məlumatları təmin edir. Operativ CRM əsasən 3 əsas komponentdən ibarətdir: satış gücü avtomatlaşdırılması, marketinq avtomatlaşdırılması və servis avtomatlaşdırılması.

- Satış gücünün avtomatlaşdırılması ilkin olaraq əlaqə məlumatlarının daxil edilməsindən tutmuş potensial müştəriləri faktiki müştərilərə dönüştürməyə kimi bütün satış mərhələlərində fəaliyyət göstərir. Məsələn, 2000-ci ilin avqust ayında OracleCRM poqram paketini təqdim etdi. OracleSalesOnline.com adlı bu paket əlaqələri, cədvəlləri və performans izləmələrini onlayn olaraq əlçatan edir ki, beləcə müştəri məlumatları ofisdə və ya kənarda fəaliyyət göstərən bütün işçilər üçün asanlıqla əldə edilən edir. Satış gücünün avtomatlaşdırılması təkrar olunan və ya gələcək potensial satışlar üçün müştərinin hesab tarixçəsinin izlənilməsini avtomatlaşdıran və satışları, marketinqi, zəng mərkəzlərini və pərakəndə satışı əlaqələndirən satış promosyon təhlilini həyata keçirir. Bu satıcı ilə müştəri arasında qarşılıqlı səylərin qarşısını alır və avtomatik olaraq hər iki tərəf arasında büütn əlaqələri və izləmələri nəzarət altına alır.
- Marketinqin avtomatlaşdırılması daha effektiv və səmərəli etmək üçün ümumi amrketinq prosesinin asanlaşdırılmasına yönləndirilmişdir.

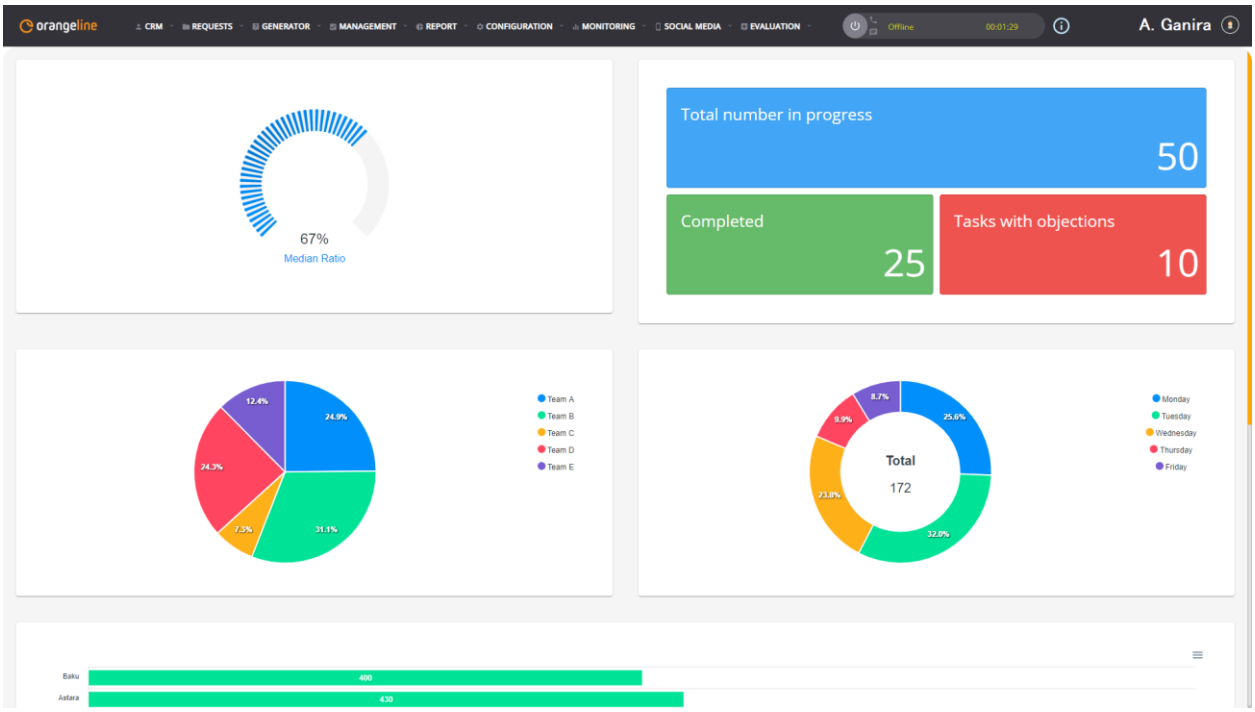
CRM alətləri bütün şirkətlər tərəfindən təşkilat iş axınlarını və sağlam və davamlı müştəri münasibətlərini qorumaq üçün fəal şəkildə istifadə olunur.

2.3.1. Orangeline, PerfomancePro, Kibrit, Qualtrics, Trello, Salesforce və Terrasoft alətlər haqqında məlumatlar

İstənilən əməkdaşın işi qiymətləndirmək üçün, iş prosesində icra olunan əməiyyatların toplusu ilə tanış olmaq lazımdır. İşçinin müştəri ilə əməliyyat prosesinə nəzər saldıqda artıq qiymətləndirmin bir hissəsini aparmaq mümkün olar. Bu proseləri izləmək üçün CRM platformalarında istifadə mütləqdir. CRM-də şirkət işçiləri mövcud, potensial və keçmiş müştərilər haqqında məlumat toplayırlar. Sistemdəki məlumatlar təhlil və sonrakı istifadə üçün əlverişli bir şəkildə saxlanılır.

Orangeline

Orange analitik sistemi, geniş tədqiqat funksiyaları ilə maşın öyrənmə və məlumatların vizuallaşdırılması üçün açıq mənbəli bir proqramdır. Orange proqram məhsulu, məlumatların çıxarılması, statistik tədqiqatlar və məlumatların vizuallaşdırılması üçün nəzərdə tutulmuşdur. Analitik platformanın komponentlərinə vidjetlər deyilir və onlar minimalist məlumatların vizuallaşdırılması, alt dəstlərin seçilməsi və əvvəlcədən işlənmədən tutmuş öyrənmə alqoritmlərinin empirik qiymətləndirilməsinə və proqnozlaşdırıcı modelləşdirilməsinə qədər dəyişir. Sistem məlumat analitiki, tədqiqatçı və alimin əlində təsirli bir vasitə olacaqdır. Daxili interfeysin nümunəsi şəkil 2.2-də göstərilmişdir.



Şəkil 2.2 Orangeline tətbiqinin interfeysi

Platformada çevik panellərin və widget-lərin yaradılması, müxtəlif vizuallaşdırma alətləri, datanın analizi və vizuallaşdırılması mümkündür.

OrangeLine proqram təminatının təyinatı:

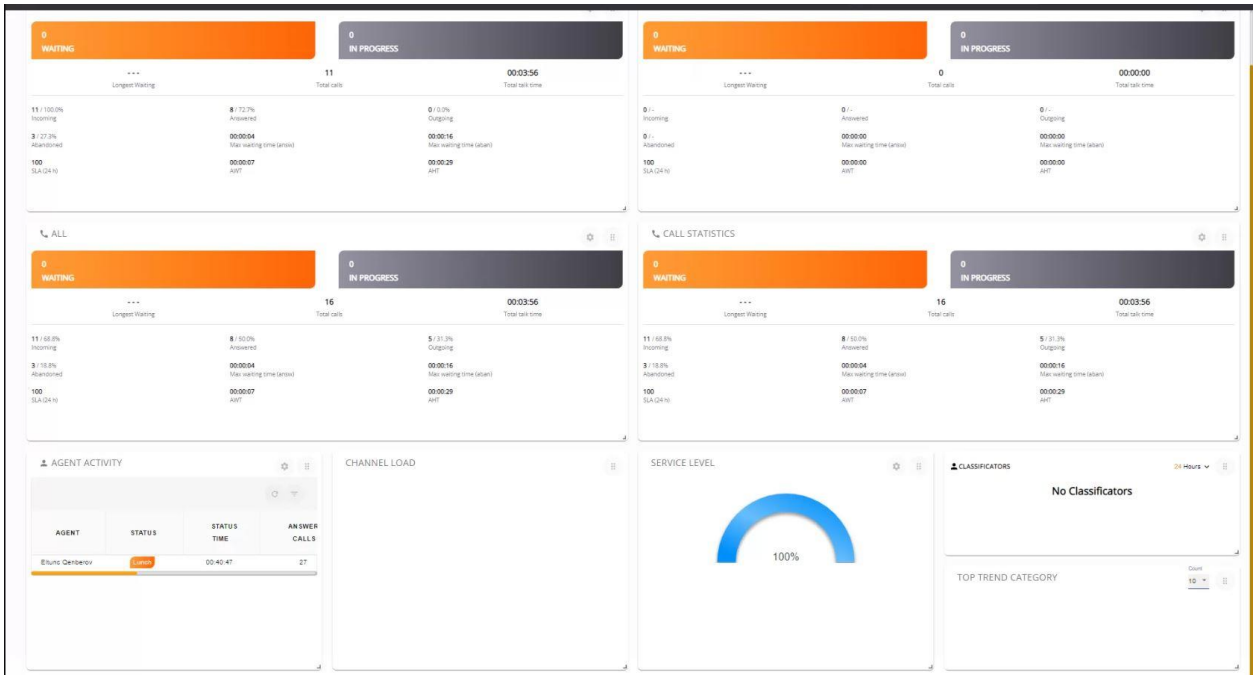
- Kiçik, orta və iri müəssisələrdə asanlıqla uyğunlaşdırılma
- Müştəri xidmətləri bölmələri, Çağrı Mərkəzlərinin tam avtomatlaşdırılması
- Müxtəlif bölmələr arasında koordinasiyanın təmini
- Müxtəlif modulları özündə birləşdirən vahid platforma
- Biznes üçün məlumatların toplanması, analizi və proqnozlaşdırılması

Vahid kanal xüsusiyyəti – bütün müraciət kanallarından (zəng, çat, e-mail və s.) inteqrasiya olunmuş və fasiləsiz şəkildə istifadə edilərək mükəmməl müştəri xidmətinin təmin edilməsidir.

Üstünlükləri: vahid platforma, ağıllı bölüşdürmə sistemi, cavablama sürəti və dəqiqliyi, avto cavablama, çevik izləmə və nəzarət, dəqiq statistika və hesabat.

Müştərinin öncədən identifikasiyası, müştəri müraciətləri barədə geniş hesabatlar, müştərinin bütün kanallar üzrə müraciətlərinin bir profayl üzərindən izlənilməsi imkanı, müştəri yönümlü müxtəlif kampaniyaların həyata keçirilməsi imkanı, üçüncü CRM sistemlərlə inteqrasiya imkanı, müştəri müraciətlərinin çevik və dəqiq cavablandırılması üçün biliklər bazası və ya tez-tez verilən suallar, müxtəlif kanallardan daxil olan müraciətlərin klassifikasiyası, müraciətin şablon cavab elementləri əsasında qısa zamanda cavablandırılması, cavab elementlərinin müştəriyə bir klik ilə SMS vasitəsilə göndərilməsi imkanı, gündəlik TOP müraciət mövzularının monitorinqi, aylıq, rüblük, illik hesabat və statistika

CRM xidməti - müraciət edən müştərilərin bazasının yaradılması, müştəri profaylı və onun idarəedilməsi. Şəkil 2.3-də əsas menyu paneli əks olunmuşdur.



Şəkil 2.3 Orangeline tətbiqinin əsas menyusu

Orange Data Mining proqramı rahat qrafik interfeysi ilə həyata keçirilən vizual proqramlaşdırmadan istifadə edir. Vizual proqramlaşdırma daxilində analitik prosedurlar əvvəlcədən təyin edilmiş və ya istifadəçi tərəfindən hazırlanmış blokları (vidjetləri) birləşdirərək yaradılır, qabaqcıl istifadəçilər məlumatları manipulyasiya etmək və yeni

bloklar (vidjetlər) yaratmaq üçün Orange-dan Python proqram kitabxanası kimi istifadə edə bilərlər.

PerfomancePro

Performance Pro performans idarəetmə proseslərini asanlaşdırmaq istəyən bütün sənaye təşkilatları üçün uygundur.

Performance Pro təlim idarəetmə sistemi, səriştəyə əsaslanan kursların avtomatlaşdırılmış paylanması və tərəqqinin davamlı qiymətləndirilməsi yolu ilə təkmilləşməkdə maraqlı olan 50-5000 işçisi olan şirkətlər üçün proqramdır. İnteqrasiya edilmiş öyrənmə idarəetmə sistemi işə qəbuldan peşəkar inkişafa qədər öyrənmə təcrübəsini optimallaşdırarkən tələb olunan məzmunla inkişafı dəstəkləyir.

Kibrit

Sahəsi 500 m^2 çox olan bütün binalar, inzibati obyektlər və ticarət obyektləri, otellər və s. üçün AİS istifadəsi məqsədəuyğun hesab olunur. Binanın bütün həyat təminatı sistemləri ABİS (avtomatlaşdırılmış bina idarəetmə sistemləri) və ya BMS (building management systems) vasitəsilə avtomatlaşdırılır. Avtomatlaşdırılmış idarəetmə sistemləri mərkəzləşdirilmiş qaydada idarə olunması və monitorinqi məqsədilə müasir obyekt və binalarda mövcud proses və əməliyyatların avtomatlaşdırılması üçün nəzərdə tutulub.

Kibrit həm AİS, həmçinin digər müxtəlif idarəetmə və proseslərə nəzarət platformalarını özündə cəmləşdirir. Məsələn öncə vurğuladığımız OrangeLine və digər platformaların cəm xüsusiyyətlərindən istifadə edir.

Qualtrics

Yalnız sorğu tərtib etmək üçün bir vasitədən daha ətraflı və mürəkkəb bir sistem axtaran təşkilatlar üçün Qualtrics təklif oluna bilər.

Şirkət peşəkar tədqiqatlar üçün ən qabaqcıl tədqiqat məntiqini və mürəkkəb təhlili təqdim edir. Bu proqram digər alətlərlə oxşar xüsusiyyətlər təklif edir, lakin daha yüksək analitik səviyyədə — hesabat almaq, tamamilə yeni bir fərdiləşdirmə səviyyəsində təhlil etmək bacarığı. Bu xidmət xüsusilə tələbkar şirkətlər üçün uygundur.

Qualtrics onlayn sorğu həlli və hər ölçüdə şirkətlər üçün araşdırma aparmaq üçün bir yoldur. Xidmət anketlər yaratmaq və müştəri məmnuniyyətini, habelə marketinq tədqiqatları və göstəricilərinin əsas nöqtələrini müəyyənləşdirmək üçün uygundur. Xidmət, müştərinin düşündüyü və istədiyi əsasında işçiləri idarə etməyə imkan verir, bu, digər heyət üzvlərinə və rəhbərliyə, məsləhətçilərə və digərlərinə də aiddir.

Trello

Trello layihə və tapşırıqların idarə edilməsi üçün onlayn platformadır. Trello kiçik şirkətlərdə və startaplarda işləməyə nəzarət etmək üçün uygundur. Bu sistem Kanban Prinsipi — məşhur layihə idarəetmə texnikası əsasında təşkil edilmişdir.

Bundan əlavə, Trello şəxsi planlaşdırma üçün istifadə edilə bilər. Veb tətbiqi gün, həftə və s. üçün iş tapşırıqlarını səmərəli şəkildə planlaşdırmağa və izləməyə kömək edəcəkdir. Həm də şəxsi səmərəliliyi artırmaq üçün əlverişli bir vasitədir. Trello-da işdən kənar Şəxsi tapşırıqları və hətta istirahəti planlaşdırmaq mümkündür. Bu vasitə istənilən mərhələdə işə daxil edilə bilər. Trello ilə necə işləməyi başa düşmək üçün əlavə kadr hazırlığına ehtiyac yoxdur.

Salesforce

Salesforce, satış, xidmət, marketinq, təhlil və müştəri əlaqələri tapşırıqlarını avtomatlaşdırmağa və asanlaşdırmağa kömək edən bir CRM platformasıdır.

Salesforce, ümumi bir verilənlər bazasına daxil edərək iş təcrübəsini avtomatlaşdırmağa və optimallaşdırmağa, satışları artırmağa və müştəri təcrübəsini yaxşılaşdırmağa kömək edir. Salesforce resurs bölgüsü arxitekturası (Multi-tenancy architecture) əsasında fəaliyyət göstərir.

Salesforce CRM-də onu müasir şirkətlər üçün geniş xüsusiyyətlərindən tutmuş təklif miqyasına qədər praktik seçim edən bir çox amillər var. Salesforce AppExchange hətta şirkətlərə başdan sona qədər tam fərdi sistemlər yaratmağa imkan verir.

Terrasoft

Terrasoft CRM, yeni alıcıları effektiv şəkildə cəlb etmək və mövcud olanlarla iş əlaqələrini inkişaf etdirmək üçün bir şirkətin müştəri münasibətləri proseslərini və əməliyyatlarını avtomatlaşdırmaq üçün hazırlanmış bir CRM sistemidir.

Terrasoft CRM-in əsas vəzifəsi marketinq, satış və istehlakçı xidmətlərinin idarə edilməsi vasitələri ilə müştəri bazası ilə işin keyfiyyətini optimallaşdırmaqdır.

CRM sistemindən istifadə biznesin inkişafı yolunda mühüm addımdır. Bu proqram işçilərə gündəlik işləri avtomatlaşdırmağa və həqiqətən vacib məsələlr ilə bağlı, müştərilərlə yüksək keyfiyyətli ünsiyyətə maksimum səy göstərməyə imkan verir.

2.3.2. Orangeline alətində istifadə olunan Java Spring Framework genişləndirilməsinin infrastrukturu

Spring Framework - Java platforması üçün universal açıq kodla freymvorkdur.

Java developerlərin də ən sevilən freymvorkudur, çünki mobil tətbiqlərin tərtibində böyük bir sərbəstlik verir. Quruluş üzərində işləməyə imkan yaradır.

Bundan əlavə, korporativ miqyaslı tətbiqlər yaratdıqda yaranan problemlərin asan vasitələrlə həllini təmin edir.

Spring-in xüsusiyyətləri hər hansı bir Java tətbiqetməsində istifadə olunur. Bu səbəbdən Spring developerlər tərəfindən strateji əhəmiyyətli freymvork olaraq tanınır.

Spring Java developerlər və təşkilatların qarşılaşdıqları problemlərin bir çoxunu həll edir.

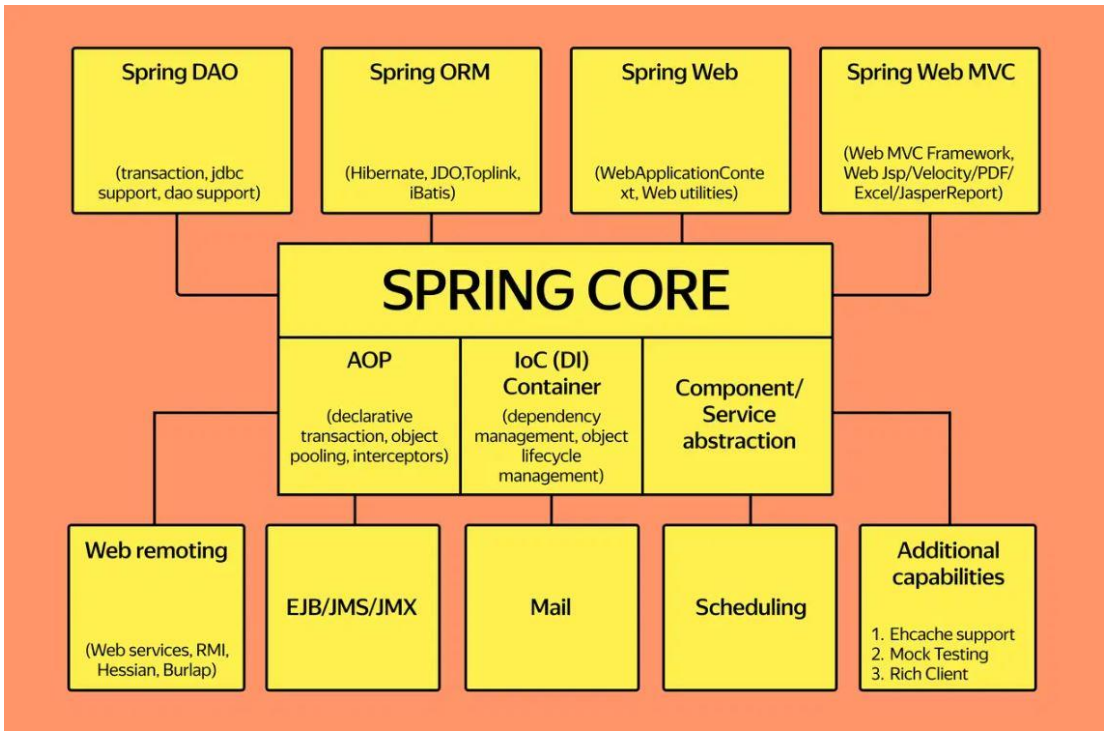
Bu freymvorkun digərlərindən fərqləndirən cəhətləri aşağıda sadalanlbdır:

- Spring yüngül proqram modelləri ilə mürəkkəb iş tətbiqetmələrini səmərəli şəkildə inkişaf etdirmək üçün tələb olunan xüsusiyyətlərin mənbəyi kimi yaxşı tanınıb.
- Ardıcıl bir tətbiq modelini yaratmağa kömək edir. Onu Java platformasında yaradılan əksər tətbiq növlərinə uyğunlaşdırır.
- Spring-in əsas məqsədi tətbiqlər üçün infrastruktur dəstəyidir: developerlər tətbiqləri quraşdırdıqları platformanın texniki detallarına deyil, yazdıqları tətbiqetmənin iş məntiqinə diqqət yetirməlidirlər.

- Spring freymvorkun əsas xüsusiyyəti Dependency Injection patternidir. DI zəif birləşdirilmiş siniflərin inkişafına kömək göstərir. Onları daha unversal edir.
- Modulların zəif əlaqəsi səbəbindən, onları sınaqdan keçirmək, developerlərin bir komanda şəklində işləməsini asanlaşdırır.

Həmçinin Spring freymvorkun bəzi digər freymvorklar və kitabxanalarla inteqrasiyası mümkündür.

Spring, Hibernate, Struts, Tapestry, JSF və s. kimi müxtəlif freymvorklar üçün dəstək təklif edən yüngül bir freymvorkdur. Java Spring freymvorku müxtəlif mini freymvorkların böyük dəstidir. Onların hər biri müəyyən tətbiqlər və ya onların hissələri üzərində işləmək üçün lazımdır (Şək. 2.4).

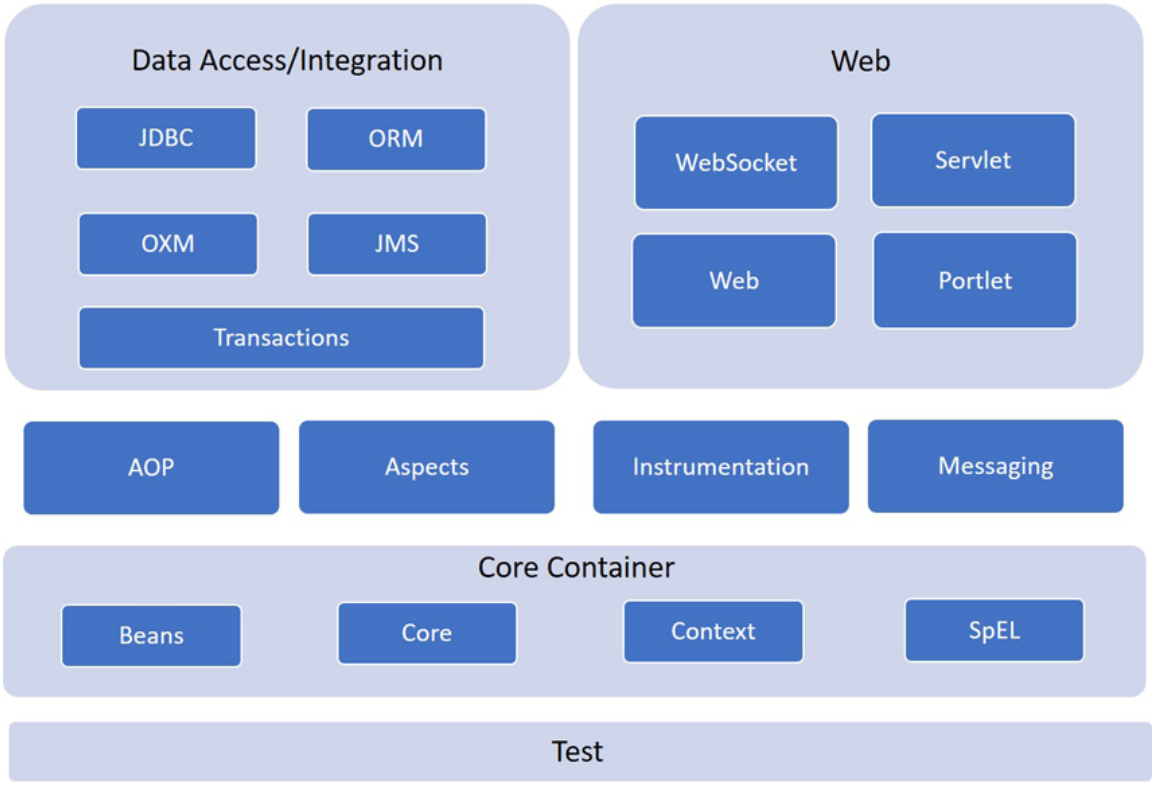


Şəkil 2.4 Spring-in əsas komponentləri

Spring, Java proqramlarını tez və asanlıqla inkişaf etdirmək üçün hərtərəfli dəstək verən açıq mənbəli, istifadəsi asan Java MVC freymvorkudur. Java Spring freymvorku tərtibatçılara biznes üçün çoxlu funksiyaları olan çox istifadəçi korporativ veb proqramları kimi mürəkkəb sistemlər yaratmaq üçün alətlər təqdim edir. Verilənlər bazası və buludlarla necə işləməyi bilən, müxtəlif modullardan ibarət olan, təhlükəsiz

kanallar vasitəsilə İnternet vasitəsilə istifadəçilərlə əlaqə quran tətbiqləri tez bir zamanda yaratmağa imkan verir. Nəzəri olaraq, bütün bunlar Java-da və əl ilə həyata keçirilə bilər, lakin bahar inkişaf etdiricilərə daha sürətli kod yazmağa və rəsmiyyətlərə deyil, proqramın unikal funksiyalarına diqqət yetirməyə imkan verən hazır alətlər verir.

Spring Framework infrastrukturunu gəldikdə, proqramın tələblərindən asılı olaraq istifadə edilə bilən 20 modul təqdim edir (Şək. 2.5).



Şəkil 2.5 Spring Framework infrastrukturu

Core və Bean, IOC və DI daxil olmaqla platformanın əsas hissəsini təmin edir.

Əsas konteyner

Əsas konteyner əlavə olaraq modullar kimi Alt komponentlərə bölünür Core, Band, Context və Expression Language.

Spring Core:

IoC (Inversion Of Control) və Inpension Injection funksiyaları daxildir. Modul fabrik şablonunun mürəkkəb tətbiqi olan BeanFactory təklif edir.

Verilənlərin əldə olunması və inteqrasiya:

Verilənlərin əldə etmə və inteqrasiya səviyyəsi JDBC, ORM, JDBC, XML, JMS və digər modullarından ibarətdir.

ORM modulu, məlumat əldə etmə texnologiyasından asılı olmayaraq kod tutarlılığını/daşınmasını təmin edir. O, obyekt yönümlü ekran konsepsiyasına əsaslanacaqdır.

JDBC modulu JDBC abstraksiya səviyyəsindən ibarətdir. Bu, JDBC ilə əlaqəli kodlaşdırma ehtiyacını anlamağa kömək edir.

OXM - Object XML Mapping (COM) obyektləri XML formatına və əksinə çevirməyə kömək edir.

Java Mesajlaşma Xidməti modulu mesajların yaradılması və istehlakı kimi funksiyaları təklif edir.

Tranzaksiya - bu modul unikal interfeyslərin həyata keçirilməsi və bütün növ POJO (Plain Old Java Object) üçün deklarativ və proqramatik idarəetmə metodunu təklif edir.

Spring AOP (Aspect Oriented Programming)

Spring-in əsas komponentlərindən biri AOP freymvorkudur. Aop Spring-də aşağıdaki hallarda istifadə olunur:

1. Xüsusilə EJB deklarativ xidmətlərini əvəz edən bəyannamə müəssisəsi xidmətlərini təmin etmək üçün. Bu cür xidmətlərdən ən vacibi Spring əməliyyat abstraksiyasına əsaslanan deklarativ əməliyyatların idarə edilməsidir.

2. İstifadəçilərə AOP ilə OOP-dan istifadəni tamamlayaraq fərdi aspektləri həyata keçirməyə icazə vermək üçün.

Spring ORM

ORM paketi verilənlər bazasına girişlə bağlıdır. JDO, Hibernate və iBatis daxil olmaqla, məşhur obyektlə əlaqəli xəritəçəkmə API-ləri üçün inteqrasiya qatlarını təmin edir.

Spring Web

Spring Web modulu Spring MVC-ni özündə cəmləşdirən Spring-in web proqram inkişaf toplusunun bir hissəsidir.

Spring DAO

Spring-də dao (Data Access Object) dəstəyi ilk növbədə JDBC, Hibernate və ya JDO kimi texnologiyalardan istifadə edərək məlumat əldə etmə işini standartlaşdırmaq üçün nəzərdə tutulmuşdur.

Spring Context

Bu paket mesaj mənbələri və Observer dizayn nümunəsi üçün dəstək əlavə etmək üçün beans paketinə əsaslanır və tətbiq obyektlərinin razılaşdırılmış API-dən istifadə edərək resurslar əldə etmək imkanı verir.

Spring-in üstünlükləri və mənfi cəhətləri

Üstünlüklər:

* Universallıq. Spring-də quraşdırılmış bir çox vasitə var, buna görə də əksər inkişaf problemlərini həll etmək üçün kifayətdir.
* İşin sürətləndirilməsi. İnkişaf üçün lazım olan bir neçə ayın işini freymvork bir neçə gün ərzində etməyə imkan verəcəkdir.
* Rutindən qurtulmaq. Freymvork eyni tipli, gündəlik vəzifələri avtomatik həll etməyə imkan verir. Bu, proqramçılara yaradıcı işlərə diqqət yetirməyə imkan verir.
* Böyük icma. Spring uzun müddətdir mövcuddur və inkişaf etdiricilər arasında çox populyardır. Bir hissəsi azərvaycan dilinə tərcümə edilmiş ətraflı sənədlər var. İxtisaslaşmış qruplarda istənilən suala cavab tapa bilərsiniz. Freymvork açıq mənbəlidir və istəyənlər onu daim inkişaf etdirirlər.
* Əlçatanlıq. Ümumiyyətlə, Java ilə əlaqəli texnologiyalar pullu və qapalıdır. Spring pulsuzdur, buna görə şəxsi məqsədlər üçün də istifadə etmək asandır.

Mənfilər:

* Uzun quraşdırma. Təmiz Spring qurmaq uzun və çətindir, proqramın xüsusi iş şəraiti üçün uyğunlaşdırılmalıdır. Spring Boot quraşdırmanı asanlaşdırır, lakin bütün məqsədlər üçün uyğun deyil.
* Çətin başlanğıc. Spring-də çoxlu alətlər və daxili freymvorklar var. Tək oturub onunla işləməyə başlaya bilməzsən-öyrənməli və anlamalısan.
* Funksiyaların çoxluğu. İnkişafda Spring-in lazım olmadığı vəzifələr var. Ancaq bəziləri hələ də ondan istifadə etməyə çalışırlar, bu da işləməyi çətinləşdirir.
* Kodun çəki artımı. Spring-də yazılmış layihələr daha çox zəruri asılılıqları ehtiva edir, daha uzun müddətə başlayır və düşünülmüş optimallaşdırma tələb edir.

Bundan əlavə, Spring bu bir çox birləşdirməyə imkan verir. Məsələn, avtomatik olaraq konfiqurasiya edilə bilən hər şeyi — Spring özü edir, burada bəzi xüsusi parametrləri göstərməlidir-Java konfiqurasiyaları əlavə olunmalıdır və əlavə olaraq xml formatında bəzi konfiqurasiyaları birləşdirəmək olar. Ümumiyyətlə, bütün bunlar kifayət qədər çevik şəkildə edilə bilər. Ümumilikdə yenə də hər şey avtomatik tənzimləmə ilə edilə bilər.

III FƏSIL. JAVA SPRİNG FRAMEWORK ALƏTLƏR DƏSTİNİN TƏKMİNLƏŞDİRİLMƏSİ

3.1. Java Spring Framework-da mövcud modulların tətbiq təhlili

Spring çox sayda moduldan ibarətdir. Bunların arasında Spring tətbiqetməsinin sadəcə başladığı başlanğıc modulları var və tətbiqə müəyyən funksionallıq əlavə edən köməkçi layihələr də var: məsələn, axın məlumatları üçün Spring Data Flow, təhlükəsizlik üçün Security və ya paylanmış sistemlər üçün Cloud. Bu quruluş, tərtibatçılara yalnız lazımi vasitələrdən istifadə edərək tətbiqləri səmərəli şəkildə qurmağa və saxlamağa imkan verir.

Java Spring freymvorkun funksionallığını artıran modullar şəkil 3.1-da göstərilmişdir.

Şəkil 3.1. Java Spring freymvorkun modulları

Spring Boot

Spring Boot, minimal səy və parametrlərlə tətbiqetmələr yaratmaq və işlətmək üçün hərtərəfli bir freymvorkdur. Bu modul iki yığına bölünür: API-yə əsaslanan Spring MVC və reaktiv Spring Webflow.

Spring WebFlux, Müasir çox nüvəli prosessorlardan maksimum yararlanmaq və eyni vaxtda çox sayda əlaqəni idarə etmək üçün qurulmuş bir web platformadır.

Spring MVC, API-lər üzərində qurulub və "bir məlumat axınına 1 sorğu"modeli ilə sinxron bloklama arxitekturasından istifadə edir.

Spring Boot-da JVM-də reaktiv sistemlər yaratmaq üçün istəyə bağlı olaraq Reactor kitabxanasını da qoşmaq olar.

Spring Data

Modul tətbiqlərə relyasiyalı və qeyri- relyasiyalı verilənlər bazası (DB), map-reduce freymvorklar və bulud xidmətləri vasitəsilə məlumatlara çıxış imkanı verir. Spring Data xüsusi VBİS-lərə (Verilənlər Bazası İdarəetmə Sistemi) həsr olunmuş bir çox alt layihələri ehtiva edir. Onların arasında, məsələn, MySQL, MongoDB, Redis və bir çox başqaları var. Həmçinin ArangoDB, Google Datastore, Microsoft Azure Cosmos DB və başqaları kimi daha spesifik verilənlər bazaları üçün Spring qrupu tərəfindən hazırlanmış alt modullardan istifadə etmək olar.

Spring Data-da həyata keçirilən əsas mexanizm depodur. Bu depo məlumatlarla qarşılıqlı əlaqədə olmaq üçün JPA Entity istifadə edən interfeyslər dəstidir.

Spring Cloud

Spring Cloud ilə paylanmış sistemlərdə şablonları asanlıqla və tez bir zamanda yaradıla bilər. Bu cür şablonlara misal olaraq: konfiqurasiya idarəetməsi, xidmət aşkarlanması, ağıllı marşrutlaşdırma, mikro proxylər, birdəfəlik tokenlər və daha çox.

Spring Cloud ilə yaradılan şablonlar, öz dizüstü kompüteriniz, məlumat mərkəzləriniz və Cloud Foundry kimi PaaS platformaları da daxil olmaqla hər hansı bir paylanmış mühitdə yaxşı işləyəcəkdir.

Spring Cloud da müxtəlif məqsədlər üçün bir çox alt layihədən ibarətdir. Belə ki, Spring Cloud Azure Spring-i Azure xidmətləri ilə birləşdirir, Spring Cloud Stream idarə olunan mikroservislər (event-driven microservices) və s.yaratmaq üçün istifadə olunur.

Spring Cloud Data Flow

Spring Cloud Data Flow Toplu məlumatların emalında və ötürülməsində istifadə edən tətbiqlərə Spring Cloud Data Flow ehtiyacı var.

Freymvork, ETL, hadisə axını və proqnozlaşdırıcı analitik daxil olmaqla bir sıra əvvəlcədən hazırlanmış məlumat işlərini dəstəkləyir.

Spring Security

Spring Security-autentifikasiya, avtorizasiya və girişə nəzarət mühitidir. Spring əsaslı tətbiqləri qorumaq üçün istifadə olunan standart bir freymvorkdur.

Spring Security öz ehtiyaclarınız üçün asanlıqla genişləndirilə bilən əsas təhlükəsizlik xüsusiyyətlərini təmin edir.

Spring Integration

Spring Integration, Spring əsaslı tətbiqlərdə mesajlaşmanı asanlaşdırır, xarici sistemlərlə inteqrasiyanı dəstəkləyir və müxtəlif mənbələrdən məlumatların işlənməsi üçün alətlər verir. Spring Cloud-un alt layihələrindən biri olan Spring Cloud Stream, Spring Integration-dan hadisə ilə idarə olunan mikroservislər üçün mühərrik kimi istifadə edir.

Spring Batch

Spring Batch, toplu tətbiqetmə inkişaf platformasıdır. Spring Batch həm sadə, həm də daha mürəkkəb layihələr üçün işləyəcək - platforma asanlıqla miqyaslanır və çox miqdarda məlumatı idarə edə bilir.

Modullar alət dəstləridir. Geniş istifadə olunan modulların siyahısı cədvəl 3.1-də göstərilmişdir.

Cədvəl 3.1 Java Spring modulları

Modul	Təsviri
aop	Bu modul, Spring-dən aspekt yönümlü proqramlaşdırma vasitələrini (AOP) tətbiq etmək üçün lazım olan bütün sinifləri ehtiva edir.
aspects	Bu modul AspectJ-də Aop kitabxanası ilə genişləndirilmiş inteqrasiya üçün nəzərdə tutulmuş bütün sinifləri ehtiva edir. AspectJ üslubunda annotasiyalardan istifadə edərək əməliyyat idarəçiliyinə ehtiyacınız varsa, bu modula tələbat olacaq.
beans	Bu modul, Spring binlərinin manipulyasiyasını dəstəkləmək üçün hazırlanmış bütün sinifləri ehtiva edir. Bu modul XML Spring konfiqurasiya faylı və Java annotasiya emal üçün tələb dəstləri ilə doludur.
context	Bu modul, sinif dəstəyi ilə əlaqəli Spring core üçün genişlənmələri təmin edən sinifləri ehtiva edir.
context-support	Bu modul spring-context modulu üçün əlavə genişləndirmələr ehtiva edir. Burada müxtəlif tapşırıq icrası və planlaşdırma kitabxanaları (CommonJ Quartz) ilə inteqrasiya üçün siniflər vardır.
core	Bu, hər Spring tətbiqi üçün tələb olunan əsas moduldur.
expression	Bu modul SpEL dili üçün bütün dəstək siniflərini ehtiva edir (Spring Expression Language - Spring ifadə dili).
instrument	Bu modul Java virtual maşını yükləmək üçün Spring cihaz agentini ehtiva edir.
instrument-tomcat	Bu modula Tomcat serverində Java virtual maşınının yüklənməsi üçün Spring cihaz agenti daxildir.
jdbc	Modul JDBC-ni dəstəkləmək üçün nəzərdə tutulmuş bütün sinifləri əhatə edir. Bu modul verilənlər bazasına giriş tələb edən bütün tətbiqlər üçün lazımdır.
orm	Bu modul məşhur ORM alətləri (Hibernate, JDO, JPA) dəstəyi ilə JDBC platforma Spring alətləri inkişaf etmiş dəsti daxildir.
oxm	Bu modul oxm (object/XML Mapping - XML-də obyektlərin göstərilməsi) üçün dəstək verir. Bu modul XML sıralaması və sıradan çıxarılması üçün nəzərdə tutulmuş siniflərlə daxildir.
test	Bu modul tətbiqlərin test edilməsinə kömək etmək üçün bir sıra toplular təqdim edir.
tx	Bu modul Spring əməliyyatlarını dəstəkləmək üçün hazırlanmış bütün sinifləri təmin edir.

web	Bu modul Spring-in veb tətbiqetmələrində istifadəsi üçün əsas növləri təmin edir.
webmvc	Bu modul, Spring platformasının öz MVC quruluşu üçün bütün qrupları ehtiva edir.
websocket	Bu modul WebSocket üçün Java API dəstəyini təmin edir.

Tərtibatçı, xüsusi proqram üçün asılılıq kimi idxal etmək istədiyi əsas dəstdən başqa, digər modullar dəstini müəyyən edə bilər.

3.2. Hibernate, Apache Struts və Apache Tiles vasitələrin inteqrasiyası üsulu ilə təklif olunan metodlar

Hibernate obyektlə əlaqəli xəritələşdirmə problemlərini həll etmək üçün Java kitabxanasıdır. 2001-ci ilin əvvəlində ilk buraxılışında Hibernate, Criteria API kimi bir alət tətbiq etmədən ORM freymvork problemlərini həll etdi. Bununla belə, Hibernate JPA spesifikasiyasının tətbiqidir və bu da öz növbəsində Java EE spesifikasiya dəstinin bir hissəsidir (müasir Jakarta EE üslubunda).

Hazırda Hibernate nəinki obyektlə əlaqəli xəritələşdirmə (eləcə də SQL-dən Java növlərinə və əksinə) tapşırıqlarını həll edir, həm də bir çox digər xüsusiyyətləri genişləndirir.

Spring və Hibernate-i inteqrasiya etməyin iki yolu var: hibernate.cfg.xml faylını qurmaq və Spring konfiqurasiya faylında məlumat mənbəyini qurmaq;

Birinci növ: Spring tərəfindən birbaşa istinad edilən məlumat mənbəyini konfiqurasiya etmək üçün hibernate.cfg.xml faylı var.

hibernate.cfg.xml-də məlumat mənbəyi quraşdırırsınızsa, hibernate.connection.provider_class atributunu göstərməlisiniz (Şək. 3.2):

```
1        <property name="hibernate.connection.provider_class">
2                org.hibernate.service.jdbc.connections.internal.C3P0ConnectionProvider
3        </property>
```

Şəkil 3.2 Hibernate class atributu

Əks halda, xəta bildiriləcək: Tələb olunan növü [javax.sql.DataSource] yerləşdirmək mümkün deyil. Sonra hibernat-c3p0 ilə əlaqəli jar paketini idxal edin:

c3p0-0.9.2.1.jar, hibernate-c3p0-4.1.7.final.jar, mchange-commons-java-0.2.3.4.jar

pom.xml-də konfiqurasiya belə görünür (çünki hibernate artıq dbcp dəstəyləmədiyi üçün, ona görə də burada c3p0 istifadə olunur (Şək.3.3)):

```
<! - Hibernate-də göstərilən əlaqə kodunda istifadə edilməli olan jar paketini idxal edin.
Budur c3p0 əlaqə Kodu. Spring və hibernate-ə inteqrasiya üçün, hibernate.cfg.xml inteqrasiyasında,
hibernate.connection.provider_class atributunun dəyəri göstərilməlidir.
        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-c3p0</artifactId>
            <version>4.1.7.Final</version>
             <! - Asılı c3p0 paketini istisna edin, əks halda daha sonra idxal olunan jar paketi ilə ziddiyyət təşkil edəcəkdir->
            <exclusions>
                <exclusion>
                    <groupId>c3p0</groupId>
                    <artifactId>c3p0</artifactId>
                </exclusion>
            </exclusions>
        </dependency>
        <dependency>
             <! - Əgər groupId burada c3p0, o halda hibernate-c3p0 yuxarıda axtarmaq lazım deyil->
            <groupId>com.mchange</groupId>
            <artifactId>c3p0</artifactId>
            <version>0.9.2.1</version>
        </dependency>
        <dependency>
            <groupId>com.mchange</groupId>
            <artifactId>mchange-commons-java</artifactId>
            <version>0.2.3.4</version>
        </dependency>
         <! - Hibernate - də göstərilən əlaqə kodunda istifadə ediləcək jar paketini idxal edin,
         burada c3p0 əlaqə kodunun sonu->
```

Şəkil 3.3 c3p0 istifadəsi

Aşağıdakı konfiqurasiya Spring konfiqurasiya faylındadır (Şək 3.4):

```
<! - Quraşdırmanın birinci üsulu: hibernate.cfg.xml faylına birbaşa yönləndirmə->
    <bean class="org.springframework.orm.hibernate4.LocalSessionFactoryBean"
        id="sessionFactory">
        <property name="configLocation" value="hibernate.cfg.xml"></property>
    </bean>
```

Şəkil 3.4 Spring konfiqurasiya faylı

İkinci üsul: Spring-də məlumat mənbəyinin qurulması

Spring-dəki bağlantılar kodu eyni dbcp-dən istifadə edir, ona görə də onu pom.xml-də konfiqurasiya etmək lazımdır (yuxarıda qeyd olunan c3p0-dan da istifadə etmək olar(Şək. 3.5)):

```
<! - Spring tərəfindən tələb olunan dbcp əlaqə kodunu konfiqurasiya edin->
        <dependency>
            <groupId>commons-dbcp</groupId>
            <artifactId>commons-dbcp</artifactId>
            <version>1.2.2</version>
        </dependency>
```

Şəkil 3.5 c3p0 Spring əlavəsində istifadəsi

Spring konfiqurasiya faylında verilənlər mənbəyi konfiqurasiya faylı altında yerləşir (Şək. 3.6):

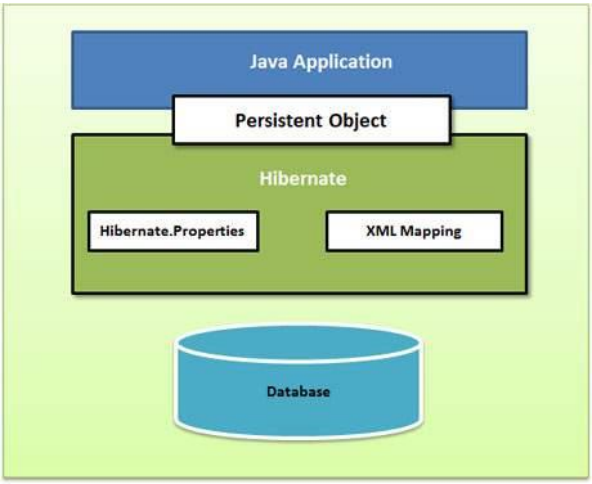
```
<! - Quraşdırmanın ikinci üsulu->
    <bean class="org.springframework.orm.hibernate4.LocalSessionFactoryBean"
        id="sessionFactory">
        <property name="dataSource" ref="dataSource"></property>
        <property name="hibernateProperties">
            <props>
                <prop key="hibernate.dialect">org.hibernate.dialect.MySQLDialect</prop>
                <prop key="hibernate.show_sql">true</prop>
                <prop key="hibernate.format_sql">true</prop>
                <prop key="hibernate.hbm2ddl.auto">create</prop>
            </props>
        </property>
        <property name="mappingLocations">
            <list>
                 <value> XML-ünvan faylı, </ value> mahiyyətinə uyğundur
            </list>
        </property>
    </bean>
```

Şəkil 3.6 Spring konfiqurasiya faylında yerləşməsi

Yuxarıda Spring və Hibernate inteqrasiyasının iki yolu göstərilibdir.

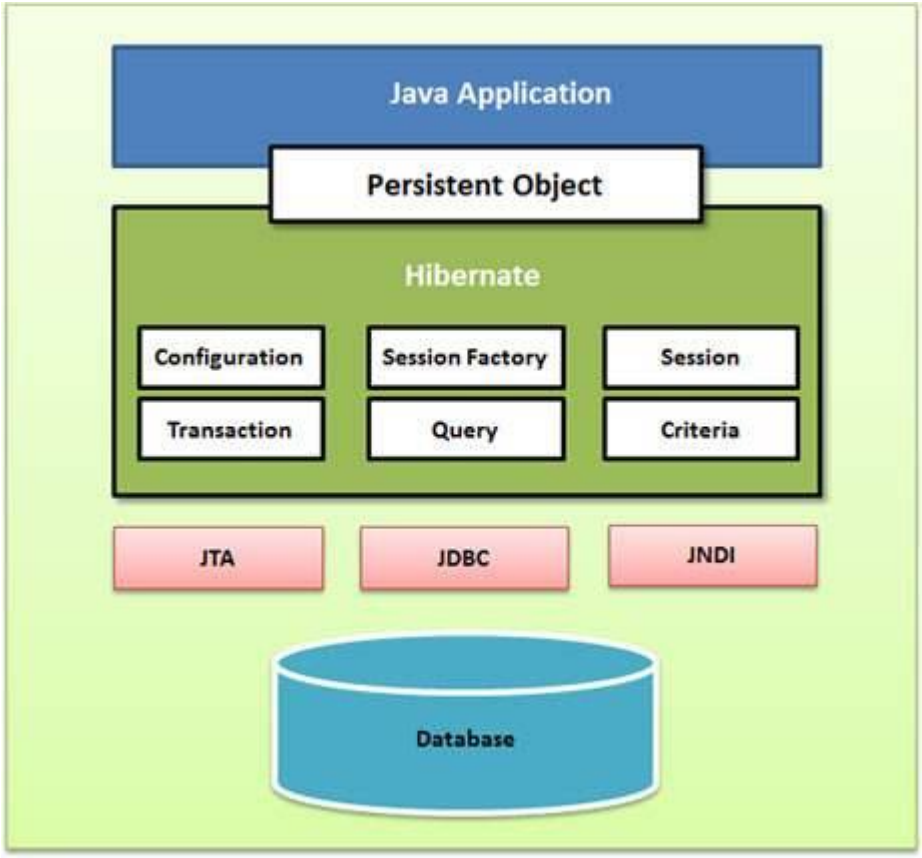
Hibernate, istifadəçinin əsas API-ləri bilmədən işləməsinə kömək edən çox səviyyəli bir arxitekturaya malikdir. Hibernate, tətbiqlərə daimi xidmətlər (və daimi obyektlər) təqdim etmək üçün verilənlər bazası və konfiqurasiya məlumatlarından istifadə edir.

Aşağıda (Şək. 3.7) Hibernate tətbiq arxitekturasının çox yüksək səviyyədə təqdimatı verilmişdir.



Şəkil 3.7 Hibernate arxitekturası

Aşağıda (Şək. 3.8) Hibernate-in vacib əsas sinifləri ilə tətbiq arxitekturasının ətraflı təsviri verilmişdir.



Şəkil 3.8 Hibernate-in əsas sinifləri ilə arxitekturası

Hibernate vasitəsi ilə Java Spring üzərində müxtəlif əmr və müvafiq bölmələr əlavə oluna bilir, buda istifadə olunan platformada dəyişklik etmək üçün lazım olan bir məqamdır.

Apache Tiles, veb tətbiq istifadəçi interfeyslərinin inkişafını asanlaşdırmaq üçün yaradılan bir şablon freymvorkdur.

Əvvəlcə jar sənədlərini Apache Tiles veb saytından yükləmək lazımdır . Layihə sinifləri qovluğuna aşağıdakı jar sənədlərini əlavə etmək lazımdır.ə etmək la.

api-xyzjar - paketi

Compat-xyzjar- paketi

əsas-xyzjar- paketi

xyzjar-JSP- paketi

Servlet-xyz jar- paketi

Yuxarıda göstərilənlərə əlavə olaraq, aşağıdakı jar sənədlərini Struts2 yükləməsindən veb-İNF / lib-ə kopyalamalıyıq .

BeanUtils-xyzjar plaqini

Ümumi -xyjar plaqini

Struts2-plugin-xyzjar dəsti

İndi aşağıda göstərildiyi kimi Struts-Tiles inteqrasiyası üçün web.xml faylını quraşdıraq lazımdır. Burada iki mühüm məqamı qeyd etmək lazımdır. Birincisi, biz tile.xml konfiqurasiya faylını haradan tapacağımızı qeyd etməliyik. Bizim nümunədə bu /WEB-INF qovluğu olacaq. Sonra, Struts2 yükləməsi ilə gələn Tiles dinləyicisini işə salmalıyıq (Şək 3.9).

```
<?xml version = "1.0" Encoding = "UTF-8"?>
<web-app xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
   xmlns = "http://java.sun.com/xml/ns/javaee"
   xmlns:web = "http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
   xsi:schemaLocation = "http://java.sun.com/xml/ns/javaee
   http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
   id = "WebApp_ID" version = "2.5">
   <display-name>Struts2Example15</display-name>

   <context-param>
      <param-name>
         org.apache.tiles.impl.BasicTilesContainer.DEFINITIONS_CONFIG
      </param-name>

      <param-value>
         /WEB-INF/tiles.xml
      </param-value>
   </context-param>

   <listener>
      <listener-class>
         org.apache.struts2.tiles.StrutsTilesListener
      </listener-class>
   </listener>

   <filter>
      <filter-name>struts2</filter-name>
      <filter-class>
         org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter
      </filter-class>
   </filter>

   <filter-mapping>
      <filter-name>struts2</filter-name>
      <url-pattern>/*</url-pattern>
   </filter-mapping>

   <welcome-file-list>
      <welcome-file>index.jsp</welcome-file>
   </welcome-file-list>
</web-app>
```

Şəkil 3.9 Struts2 tətbiqi

Qeyd olunan alətlərin tətbiqi ilə, biz işləyəcəyimiz platforma və ya mühitə lazım olan funksionallığı əlavə edə bilərik. Bunlar vasitəsi ilə, platfroma üzərində ölçülən kəmiyyətlərin və ya parametrlərin sayını artıra billərik, həmçinin onların idarə olunmasını təmin edə bilərik. Biz istifadə edəcəyimiz platforma OrangeLine tətbiqidir və orada verilən parametrlərin sayı və işləmə ardıcıllığı bizi qane etmədiyi üçün, həmin tətbiqə müəyyən əlavələr tətbiq edəcəyik.

3.3. Java proqramlaşdırma dilində işçinin fəaliyyətinin çoxfunksiyalı qiymətləndirilməsi üçün kodun hazırlanması

Öncəki fəsillərdə təyin etdiyimiz 20 parametr üzrə işçinin qiymətləndirilməsi aparalıcaqdır. Lakin qeyd olunan parametrlər 4 böyük qrupa bölünür və verilənlər bazasına əlavə olunur. Hər qrup özündə 5 parametr ehtiva edir, parametrlər məqsədə

uyğun olaraq seçilib və paylaşdırılırb qruplar arasında. Ətraflı təsviri cədvəl 3.2-də görə bilərik.

Cədvəl 3.2 Qiymətləndiirlmə üçün seçilmiş parametrlər və qruplar

Sərişlələr çoxluğu	ünsiyyət bacarıqları; stresə davamlılıq; məsuliyyət; özünə inam; yüksək öyrənmə qabiliyyəti;
Təşkilatın inkişafı	peşəkarlıq; müştərilərlə işləmək bacarığı; təşəbbüs, fikir yaratmaq; əzmkarlıq; liderlik keyfiyyətləri;
İşdə yüksək nəticələr əldə etmək	analitik düşüncə; işin planlaşdırılması; tez qərar vermək bacarığı; məlumatların toplanması və təhlili; yüksək fəaliyyət nəticələrinə diqqət;
İnsanlarla işləmək	çalışqanlıq; düşüncə çevikliyi; insanlarla ünsiyyət qurmaq və ortaq bir dil tapmaq, münasibətləri idarə etmək bacarığı; öz inkişafı və həmkarlarının inkişafına kömək; komandada işləmək bacarığı;

Qeyd olunan parametrləri biz verilənlər bazasına əlavə etməliyik ki, növbəti mərhələlərdə, həmin məlumatları emal edə bilək. Göstərilən parametrləri ehtiva edən cədvələ uyğun olaraq yaradılmış verilənlər bazası üçün Java kodu şəkil 3.10-də göstərilmişdir:

```java
import java.sql.*;

public class DatabaseExample {
    public static void main(String[] args) {
        // Verilənlər bazası qoşulma parametrləri
        String url = "jdbc:mysql://localhost:3306/mydatabase";
        String username = "root";
        String password = "password";

        // Verilənlər bazası ilə bağlantının yaradılması
        try (Connection connection = DriverManager.getConnection(url, username, password)) {
            // Cədvəlin yaradılması
            String createTableQuery = "CREATE TABLE parameters (" +
                    "id INT PRIMARY KEY AUTO_INCREMENT," +
                    "parameter_name VARCHAR(100) NOT NULL" +
                    ")";
            try (Statement statement = connection.createStatement()) {
                statement.executeUpdate(createTableQuery);
                System.out.println("Cədvəl 'parameters' uğurla yaradılmışdır.");
            }

            // Məlumatların cədvələ daxil edilməsi
            String insertQuery = "INSERT INTO parameters (parameter_name) VALUES (?)";
            try (PreparedStatement preparedStatement = connection.prepareStatement(insertQuery)) {
                preparedStatement.setString(1, "Səriştələr çoxluğu");
                preparedStatement.executeUpdate();

                preparedStatement.setString(1, "Təşkilatın inkişafı");
                preparedStatement.executeUpdate();

                preparedStatement.setString(1, "İşdə yüksək nəticələr əldə etmək");
                preparedStatement.executeUpdate();

                preparedStatement.setString(1, "İnsanlarla işləmək");
                preparedStatement.executeUpdate();

                System.out.println("Məlumatlar cədvələ uğurla əlavə edildi.");
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

Şəkil 3.10 Java üzərində qruplar üçün VB yaradılması

Verilənlər bazasına uyğun olaraq əlaqə parametrlərini (url, istifadəçi adı və parol) dəyişdirməl lazımdır. Java-da parametr verilənlər bazası yaratmaq üçün JDBC (Java Database Connectivity) və SQL istifadə etmək olar. Cədvəli tamalamaq və parametrlərlə qeydlər əlavə etməyi üçündə kod yazılmalıdır (şək. 3.11):

```java
import java.sql.*;

public class ParameterDatabase {
    public static void main(String[] args) {
        try {
            // Verilənlər bazası ilə əlaqə qurmaq
            Connection connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/mydatabase", "username", "password");

            // Cədvəlin yaradılması
            createTable(connection);

            // Parametrlərlə qeydlərin əlavə edilməsi
            addParameter(connection, "Səriştələr çoxluğu", 5);
            addParameter(connection, "Təşkilatın inkişafı", 5);
            addParameter(connection, "İşdə yüksək nəticələr əldə etmək", 5);
            addParameter(connection, "İnsanlarla işləmək", 5);

            // Verilənlər bazası əlaqəsinin bağlanması
            connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    // Cədvəl yaratmaq üçün üsul
    private static void createTable(Connection connection) throws SQLException {
        String createTableQuery = "CREATE TABLE IF NOT EXISTS parameters (" +
                "id INT AUTO_INCREMENT PRIMARY KEY," +
                "name VARCHAR(100) NOT NULL," +
                "rating INT NOT NULL" +
                ")";
        Statement statement = connection.createStatement();
        statement.execute(createTableQuery);
        statement.close();
    }

    // Parametrlər ilə qeydlərin əlavə olunma üsulu
    private static void addParameter(Connection connection, String name, int rating) throws SQLException {
        String addParameterQuery = "INSERT INTO parameters (name, rating) VALUES (?, ?)";
        PreparedStatement preparedStatement = connection.prepareStatement(addParameterQuery);
        preparedStatement.setString(1, name);
        preparedStatement.setInt(2, rating);
        preparedStatement.executeUpdate();
        preparedStatement.close();
    }
}
```

Şəkil 3.11 VB parametrlərlə qeydlər əlavə etmək üçündə kod

Qeydlər:

Verilən kod MySQL verilənlər bazasından istifadə edir. Layihədə müvafiq JDBC drayverinin əlavə olunmasına əmin olmaq lazımdır.

"jdbc:mysql://localhost:3306/mydatabase" sətrini verilənlər bazası ünvanına, "istifadəçi adı" və "parol"u verilənlər bazasına daxil olmaq üçün hesab məlumatları ilə əvəz edin.

CreateTable metodu id (avtomatik artım identifikatoru), ad (parametr adı) və reytinq (parametr reytinqi) sahələri ilə parametrlər cədvəlini yaradır.

AddParameter metodu, göstərilən parametr adı və qiymətləndirməsi ilə cədvələ yeni bir giriş əlavə edir.

Qeyd edək ki, kod verilənlər bazasında səhv emalını ehtiva etmir. Tətbiqdə müvafiq istisna işləyiciləri və səhv nəzarəti əlavə etmək lazım olacaqdır.

Seçilmiş 20 parametr üzrə, müştəri xidmətləri əməkdaşının göstəricilərinin hesablanması üçün yazılmış kod və həmin məlumatlara əsasən maksimal, minimal və orta balın hesablanması məntiqi şəkil 3.12 a və şəkil 3.12 b göstərilmişdir.

```java
import java.util.HashMap;
import java.util.Map;

public class BankEmployeeEvaluator {
    public static void main(String[] args) {
        // İşçilərin parametrlərini saxlamaq üçün lüğətin yaradılması
        Map<String, int[]> employeeParameters = new HashMap<>();

        // Gurban işçisi üçün parametrlər
        int[] gurbanParameters = {8, 7, 9, 6, 8, 7, 9, 8, 7, 8, 9, 6, 7, 8, 9, 7, 8, 9, 7, 8};
        employeeParameters.put("Gurban", gurbanParameters);

        // Vusal işçisi üçün parametrlər
        int[] vusalParameters = {9, 8, 7, 8, 9, 7, 8, 9, 6, 7, 9, 8, 7, 9, 7, 8, 9, 7, 8, 9};
        employeeParameters.put("Vusal", vusalParameters);

        // Javid işçisi üçün parametrlər
        int[] javidParameters = {7, 9, 8, 7, 8, 9, 7, 8, 9, 6, 7, 8, 9, 8, 7, 9, 6, 7, 8, 9};
        employeeParameters.put("Javid", javidParameters);

        // Hər bir işçi üçün parametrlərin qiymətləndirilməsi
        for (Map.Entry<String, int[]> entry : employeeParameters.entrySet()) {
            String employeeName = entry.getKey();
            int[] parameters = entry.getValue();

            double averageScore = calculateAverageScore(parameters);
            int maxScore = calculateMaxScore(parameters);
            int minScore = calculateMinScore(parameters);

            System.out.println("İşçi üçün qiymətləndirmə " + employeeName + ":");
            System.out.println("Ortalama bal: " + averageScore);
            System.out.println("Maksimal bal: " + maxScore);
            System.out.println("Minimal Bal: " + minScore);
            System.out.println();
        }
    }
```

Şəkil 3.12 (a) Göstəricilərinin hesablanması

Bu kod hər bir işçinin parametrlərini saxlamaq üçün Map funksiyasından istifadə edir. Hər bir işçi bir sıra parametrləri ilə müqayisə olunur.

```java
// Orta balın hesablanması üsulu
private static double calculateAverageScore(int[] parameters) {
    int sum = 0;
    for (int parameter : parameters) {
        sum += parameter;
    }
    return (double) sum / parameters.length;
}

// Maksimum balın hesablanması üsulu
private static int calculateMaxScore(int[] parameters) {
    int maxScore = parameters[0];
    for (int i = 1; i < parameters.length; i++) {
        if (parameters[i] > maxScore) {
            maxScore = parameters[i];
        }
    }
    return maxScore;
}

// Minimum balın hesablanması üsulu
private static int calculateMinScore(int[] parameters) {
    int minScore = parameters[0];
    for (int i = 1; i < parameters.length; i++) {
        if (parameters[i] < minScore) {
            minScore = parameters[i];
        }
```

Şəkil 3.12 (b) Göstəricilərinin maksimal, minimal və orta dəyərinin hesablanması

Daha sonra hər bir işçi üçün parametrlər massivi götürən və orta dəyəri qaytaran, Average Score() hesablama metodunu çağırmaqla orta xal hesablanır. Nəticə hər bir işçi üçün ayrılıqda göstərilir.

3.4. Java-da hazırlanan koda SOLID, DRY, TDD təcrübələrin tətbiqi

Java-da SOLID Prinsipləri.

Proqram mühəndisliyində SOLID, proqram dizaynını daha başa düşülən, çevik, etibarlı və davamlı etmək üçün nəzərdə tutulmuş 5 dizayn prinsipinin qısaldılmasıdır. Bu prinsiplərin qəbulu həmçinin kod xətasının qarşısını almağa kömək edə bilər.

SOLID-in 5 prinsipi bunlardır:

S - Vahid Məsuliyyət Prinsipi

O - Açıqlıq-qapalılıq prinsipi

L - Liskov əvəzetmə prinsipidir

I - İnterfeyslərin ayrılması prinsipi

D - Asılılığın inversiya prinsipi

SOLID-in prinsipləri istənilən proqramlaşdırma dilinə aid olsa da, sonrakı hissədə onların hər birini xüsusi olaraq Java-da yazılmış misallarla izah edəcəyik.

Vahid məsuliyyət prinsipi

Bu prinsipdə deyilir ki, “sinifin dəyişməsi üçün yalnız bir səbəb olmalıdır”, yəni hər sinifin tək bir məsuliyyəti olmalıdır.

Vahid məsuliyyət prinsipinə çatmaq üçün yalnız bir funksiyanı yerinə yetirən ayrı bir sinif tətbiq etməliyik.

Açıqlıq-qapalılıq prinsipi

Bu prinsipdə deyilir ki, “proqram qurumları (siniflər və s.) genişlənməyə açıq olmalıdır, lakin modifikasiya üçün bağlanmalıdır”, bu o deməkdir ki, sinifdə heç nəyi dəyişmədən genişləndirilə bilən olmalıdır.

Hər kəsin funksiyanızı genişləndirərək yenidən istifadə edə biləcəyi bir kod hazırlamaq lazımdır və hər hansı bir dəyişikliyə ehtiyac olarsa, sinfi genişləndirə və funksiyasını üstünə əlavə edə biləcəklər.

Liskov əvəzetmə prinsipidir

Bu prinsip "törəmə siniflər öz əsas siniflərini əvəz edə bilməlidir" deyir. Başqa sözlə, əgər A sinfi B sinifinin kiçik sinfidirsə, onda proqramın cari prosesi dayandırmadan B-ni A ilə əvəz edə bilməliyik.

İnterfeyslərin ayrılması prinsipi

Bu prinsip interfeyslərə aiddir və vahid məsuliyyət prinsipinə bənzəyir. Orada deyilir: “müştəri heç vaxt istifadə etmədiyi bir interfeys tətbiq etməyə məcbur

edilməməli və ya Müştərilər istifadə etmədikləri metodlardan asılı olmağa məcbur edilməməlidir.”

Asılılığın inversiya prinsipi

Asılılıq inversiyası (DIP) prinsipində deyilir ki, “Varlıqlar konkret tətbiqlərdən (siniflərdən) deyil, abstraksiyalardan (mücərrəd siniflər və interfeyslərdən) asılı olmalıdırlar. Həmçinin, yüksək səviyyəli modul aşağı səviyyəli moduldan asılı olmamalıdır, lakin hər ikisi abstraksiyalardan asılı olmalıdır.”

SOLID prinsiplərinə riayət etmək layihənimi daha az mürəkkəbliklə genişləndirilə bilər, asanlıqla dəyişdirilə bilən, yaxşı sınaqdan keçirilmiş edə bilər.

Java-da DRY Prinsipləri.

DRY - Don’t repeat yourself. Kodun təkrarlanmasının qarşısını almağa dəyər.

Kodunuz təkrarlanmırsa, onda onun dəyişdirilməsi və modifikasiyası həmişə bir yerdə baş verəcək, bu da səhvlərin sayını azaldacaq, testi asanlaşdıracaq və anlayışı yaxşılaşdıracaqdır. Əks təqdirdə, test edərkən və yeni bir funksionallıq tətbiq edərkən məhsulunuzu dəhşətli xətalara məhkum edirsiniz.

DRY inkişafın əsas prinsipidir. Kod yazarkən həmişə bir və ya digər parçanı necə yenidən istifadə edə biləcəyinizi, universal bir funksiyaya və ya sinfə ayrıla biləcəyinizi, modul edə biləcəyinizi düşünün. Eyni zamanda, hər bir heterojen funksiya üçün kitabxanalar yaratmaqdan danışmırıq — bir neçə yerdə tapılan, bəlkə də funksiyaya gətirməyin mənası olan çox oxşar bir məntiqi nəzərdə tuturuq. Və eyni funksiya bir neçə yerdə müəyyən edilmişdirsə, o zaman ümumi modula daxil edilə bilər. Və nəhayət, eyni modulu tez-tez istifadə etsək, ehtimal ki, ondan bir kitabxana edilə bilər.

Bu prinsip platformadan və dildən asılı olmayaraq həmişə faydalıdır. Deyək ki, bəzi davranışları avtomatlaşdırmalısınız. Eyni məntiqi bir neçə dəfə yazmamaq və Kodu lazımsız şəkildə şişirtməmək üçün onu ümumiləşdirməyə və ayrı bir elementə gətirməyə çalışın.

Java-da TDD Prinsipləri.

TDD və ya Test Driven Development testlər vasitəsilə inkişafdır: əvvəlcə inteqrasiya testləri yazılır, burada gələcək funksionallıq ətraflı təsvir olunur. Və bu testlər əsasında yeni xüsusiyyətlər hazırlanır.

TDD prinsipinin üstün cəhətləri aşağıdakılardır:

İnkişafı sürətləndirir. Tərtibatçılar mətn yazmağa öyrəşsələr də, iş daha yavaş gedəcək. Buna öyrəşdikdən sonra inkişaf sürətlənir;

Funksional testlərin yüksək əhatəsini təmin edir. Hər bir xüsusiyyət üçün bütün mümkün istifadə hallarını əhatə etmək üçün 20-25 test yazıram;

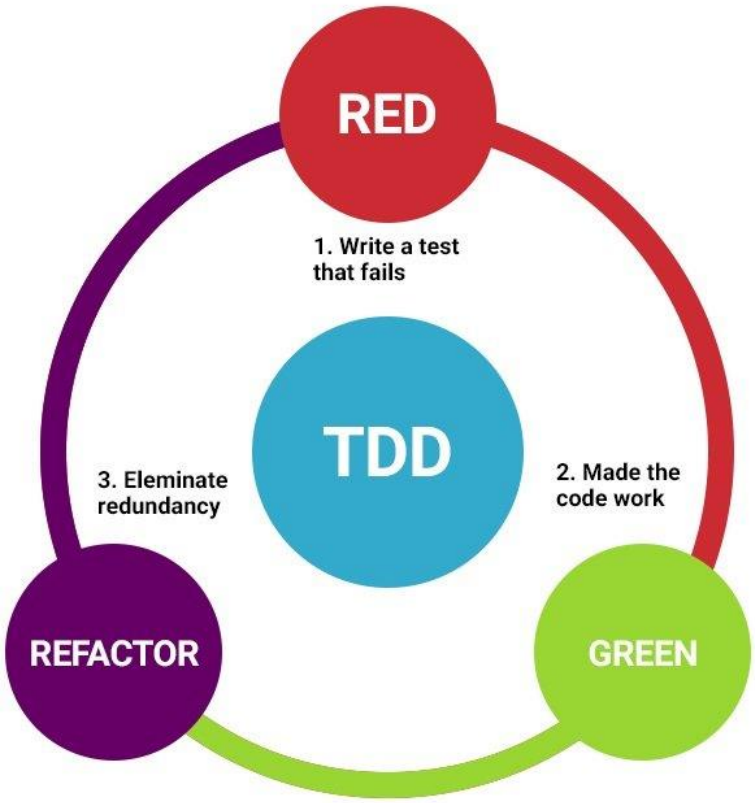
Sazlama üçün daha az vaxt tələb olunur. Adətən sazlama əsas funksionallıq artıq yerinə yetirildikdə başlayır. Burada testlər yazarkən funksionallıq anlayışını əvvəlcədən düzəldirik.

TDD-nin inkişafında istənilən kodun yolu 3 mərhələdən keçir:

1. Qırmızı mərhələ: yazı testləri. Test etdikləri interfeys hazır olmadığı üçün ilk sınaq işi həmişə xəta verir. Hər bir sınaqda ən azı bir dəfə uğursuzluq yaşanmalıdır. Ən azı yazdıqdan dərhal sonra, belə hallar baş verir.

2. Yaşıl mərhələ: biz funksionallığı inkişaf etdiririk və onu yenidən sınaqdan keçiririk. Bizim vəzifəmiz testin yaşıl rəngə çevrilməsini təmin etməkdir. Beləliklə, funksionallığın düzgün inkişaf etdirildiyini başa düşürük. Testin ən azı bir dəfə uğursuz olması və ən azı bir dəfə keçməsi vacibdir. Həmişə uğursuz və ya həmişə keçən bir test yazanda səhv etmək asandır. Amma həm xəta verən, həm də keçən test bir növ məntiqi açıq şəkildə yoxlayır. Bundan sonra, funksionallığın düzgün işlədiyini güman edə bilərik.

3. Refaktorinq. Biz konkret tapşırığı yenidən nəzərdən keçiririk, çünki biz artıq işləyən test kodumuz olduğuna əminik. Biz sadəcə layihəmiz üçün məntiqli və təmiz kod yazırıq (Şək. 3.13).



Şəkil 3.13 TDD-nin inkişafında əsas 3 mərhələ

TDD harada tətbiq edilə bilər sualnıa cavab tapmaq üçün, ilk öncə TDD seçiminin nədən asılı olduğunu bilməliyik. Bu aşağıdakılardan asılıdır:

1. Komanda və vəzifələri təyin edən və qəbul edən şəxs (müştəri ola bilər).

2. Layihədə daxili proseslər. Məsələn, komanda davamlı inteqrasiyadan istifadə etmir. Burada TDD istifadə etməyə dəyməz, çünki geliştirici Kodu depoya doldurmadan əvvəl hər dəfə testlər keçirməyəcəkdir.

TDD-nin işləyəcəyini müəyyənləşdirə biləcəyiniz meyarlar var:

- Layihə interfeyslərin quruluşunu aydın şəkildə təsvir edir, məsələn, JSON-API və ya REST-API istifadə olunur. Bizim vəzifəmiz məhsulun inkişafını sürətləndirməkdir. Aydın bir quruluş olduqda, proqramçı quruluşu özü icad etmir. Aydınlıq yoxdursa, proqramçı interfeyslərin quruluşunu düşünür və görmə qabiliyyəti Müştərinin istəyi ilə

üst-üstə düşə bilməz. Buna görə, layihədə düşünülmüş bir API quruluşu varsa, koddan əvvəl testlər yazın;

- Layihədə interfeyslərin həyata keçirilməsinə eyni yanaşma, yəni.fərqli elementlər üzərində işləmə məntiqi və qaydası Eyni və ya çox oxşardır. Testləri yazmadan əvvəl test mühitində tələb olunan məlumatları təşkil edirik. Layihədə interfeyslərin həyata keçirilməsinə müəyyən bir yanaşma varsa, bir şablonu etibarlı şəkildə tətbiq edə və testlərin yazılmasına vaxt ayıra bilərsiniz. İnkişafda vahid bir yanaşma yoxdursa, başlamazdan əvvəl hər bir test üçün öz şərtlərinizi yaratmalısınız. Bu, adi inkişafda olduğu qədər vaxt aparacaq, buna görə TDD istifadə etməyin heç bir faydası olmayacaq;

- Tapşırıqlar üslubda qoyulur: bir alt tapşırıq - bir test. Adətən bu cür vəzifələr müştəri-texnik tərəfindən qoyulur. Sistemə başlayın: tapşırıq daxilində hər alt tapşırıq bir testdir. Sonra tərtibatçılar sakitcə alt tapşırıqları təsvir edən testlər yazırlar, sonra həyata keçirirlər və testlər tədricən yaşıl olur. Hər alt tapşırıqda birdən çox test təcrid oluna bilərsə, proses adi inkişafla müqayisədə vaxt qazanmaz;

- Layihədə istifadə olunan texnologiya üçün inteqrasiya testlərini dəstəkləyən yüksək səviyyəli bir test çərçivəsi var. Alternativ seçim: başqa bir texnologiyada həyata keçirilmədən mücərrəd bir test çərçivəsi.

- Komandada ayrı-ayrı testçilər yoxdur. Komandada testçilər varsa, testlər məhz onlar tərəfindən yazılmalıdır. Nəticədə TDD - nin əsas üstünlüyü itirilir - "proqramçının funksionallıqla zehni əlaqəsi", çünki testlər başqa bir şəxs tərəfindən yazılır. TDD-nin faydalanması üçün testlər bir proqramçı tərəfindən yazılmalıdır.

Bir proqramçı JSON-u front-end qaytaran API yazırsa, TDD mütləqdir. Proqramçı API-ni TDD olmadan yazırsa, JSON-u “gözlər”lə yoxlamaq çox uzun vaxt aparır. Nə qədər uzun müddət yoxlayırsa, bir o qədər çox səhv olur və onları düzəltmək bir o qədər uzun çəkir. Testlər yazıldıqda, onlar avtomatik olaraq JSON-da səhvləri yoxlayır. Əvvəlcədən yazılmış mətnlər proqramçıya JSON çıxışını doğrulamağa kömək edəcək. Nəticədə inkişaf daha sürətli gedir.

Tərtibatçılar hesab edirlər ki, TDD aşağıdaki hallarda işləmir.:

TDD yalnız məhsul şirkətləri üçün tətbiq olunur;

TDD üçün uyğun texnologiya yığını olmadıqda;

Metodologiya işləmir;

Yalnız backend-də edilə bilər;

TDD universal bir həll deyil. Bəzi hallar üçün uyğundur, ancaq bir yerə zərər verə bilər. TDD layihə üçün uyğun deyilsə, inkişafa mane olacaq və yavaşlatacaq.

3.5. Fəaliyyətin çox parametrli qiymətləndirilməsi üçün Orangeline-nin təkminləşdirilməsi

Bundan öncəki yarımfəsildə qeyd etdiyimiz 3 prinsipi, yəni, SOLID, DRY, TDD təcrübələrini nəzər alaraq, Orangeline platformasında müəyyən dəyişikliklər tətbiqedə bilərik. SOLID, DRY və TDD prinsiplərindən istifadə edərək Java və Orangeline interfeysi yaratmaq üçün biz modul və yaxşı təşkil edilmiş kod hazırlamalıyıq. Bu prinsipləri nəzərə alan kod strukturunun nümunəsi şəkil 3.14-də göstərilibdir.

```java
// Employee.java
public class Employee {
    private String name;
    // digər sahələr, getters, setters

    public Employee(String name) {
        this.name = name;
    }
}

// EmployeeEvaluationService.java
public interface EmployeeEvaluationService {
    Map<String, Integer> evaluateEmployee(Employee employee);
}

// OrangelineEmployeeEvaluationService.java
@Service
public class OrangelineEmployeeEvaluationService implements EmployeeEvaluationService {

    @Override
    public Map<String, Integer> evaluateEmployee(Employee employee) {
        // Orangeline-da işçilərin qiymətləndirilməsi məntiqi
        // ...
        return evaluations;
    }
}
```

Şəkil 3.14 SOLID, DRY və TDD təcrübələrinin OrangeLine-da tətbiqi

Bu nümunədə:

- Employee müvafiq sahələri və metodları ilə işçi sinifini təmsil edir.
- Employee Evaluation Service işçini qiymətləndirmək üçün interfeysi müəyyənləşdirir.
- OrangelineEmployeeEvaluationService işçi qiymətləndirməsini həyata keçirmək üçün EmployeeEvaluationService tətbiq edir və Orangeline ilə qarşılıqlı əlaqə qurur.
- EmployeeEvaluationController, işçinin qiymətləndirmə tələblərini qəbul edən və qiymətləndirməni yerinə yetirmək üçün işçidən istifadə edən bir nəzarətçi təqdim edir.
- EmployeeEvaluationTest, EmployeeEvaluationService üçün vahid testlərini ehtiva edir.

Yazılan kod davam elətdirsək şəkil 3.15-da göstərilənləri əldə edərik.

```
// EmployeeEvaluationController.java
@RestController
public class EmployeeEvaluationController {

    private final EmployeeEvaluationService evaluationService;

    @Autowired
    public EmployeeEvaluationController(EmployeeEvaluationService evaluationService) {
        this.evaluationService = evaluationService;
    }

    @PostMapping("/evaluate")
    public ResponseEntity<Map<String, Integer>> evaluateEmployee(@RequestBody Employee employee) {
        Map<String, Integer> evaluations = evaluationService.evaluateEmployee(employee);
        return ResponseEntity.ok(evaluations);
    }
}

// EmployeeEvaluationTest.java
public class EmployeeEvaluationTest {

    private EmployeeEvaluationService evaluationService;

    @BeforeEach
    public void setUp() {
        evaluationService = new OrangelineEmployeeEvaluationService();
    }

    @Test
    public void testEvaluateEmployee() {
        Employee employee = new Employee("Gurban");
        Map<String, Integer> evaluations = evaluationService.evaluateEmployee(employee);
        // İşçinin qiymətləndirmələrinin yoxlanılması
        // ...
        Assertions.assertNotNull(evaluations);
        // ...
    }
}
```

Şəkil 3.15 İşçi siniflərin əlavə olunması prosesi

Orangeline Java Spring-də işlədiyindən, Java kodunu Orangeline ilə cütləşdirmək üçün Spring-in imkanlarından istifadə edə bilərik. Java Spring-in Orangeline ilə əsas inteqrasiyasını nümayiş etdirən kod 3.16 şəkilində göstərilmişdir.

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.Bean;

@SpringBootApplication
public class OrangelineIntegrationApplication {

    @Autowired
    private OrangelineService orangelineService;

    public static void main(String[] args) {
        SpringApplication.run(OrangelineIntegrationApplication.class, args);
    }

    @Bean
    public CommandLineRunner demo() {
        return args -> {
            // Xidmət vasitəsilə Orangeline istifadə nümunəsi
            orangelineService.someMethod();
        };
    }
}

@Service
public class OrangelineService {

    public void someMethod() {
        // Orangeline ilə qarşılıqlı əlaqə üçün əlavə
    }
}
```

Şəkil 3.16 Java Spring-in Orangeline ilə əsas inteqrasiyası

Bu nümunə, Orangeline ilə qarşılıqlı əlaqə üsullarını ehtiva edən bir OrangelineService xidmət sinifinə sahib olduğunuzu düşünür. Bu xidməti @Autovired annotasiyası ilə Orangelineıntegrationapplication sinfinə daxil edə və demo metodunda istifadə edə bilərsiniz.

Bunun xaricində, Orangeline tələblərinizdən və spesifikasiyalarınızdan asılı olaraq @komponent, @Repository və daha çox şərhlərdən istifadə edərək Orangelineıntegrationapplication sinifində tələb olunan binaları və komponentləri də konfiqurasiya edə bilərsiniz.

Pom.xml faylınıza (əgər biz maven istifadə ediriksə) və ya müvafiq layihə konfiqurasiya faylına əlavə edilmiş Orangeline üçün bütün tələb olunan asılılıqların qoşulu olduğundan əmin olmalıyıq.

Qeyd edək ki, kod geniş şəkildə verilmişdir və Orangeline-ın xüsusiyyətlərindən və tələblərinizdən asılı olaraq əlavə tənzimləmə tələb oluna bilər. Java Spring ilə inteqrasiya yolunun daha ətraflı olması üçün Orangeline ilə geniş şaxəli genişləndirilmə mümkündür.

Bütün bu qeyd olunanları nəzərə alaraq, öncəki yarımfəsillərdə qeyd etdiyimiz 20 parametrlərin qiymətləndirilməsini apara bilərik. Həmin parametrlər, hansıları ki 4 qrupa böldük və hər qrupda müvafiq olaraq 5 parametr qeyd etdik, banklarda müştəri xidməti fəaliyyətinin çoxfunksiyalı qiymətləndirilməsini təmin edəcəkdir.

Cədvəl 3.2 qeyd olunan verilənlərin, 1-ci qrupunun 5 parametrinin, nümünə olaraq Java proqramlaşdırma dilində Orangeline platforması üçün kodunu yazaq (Şək. 3.17). Parametrlər: ünsiyyət bacarıqları; stresə davamlılıq; məsuliyyət; özünə inam; yüksək öyrənmə qabiliyyəti. Nümunədə 3 işçi qeyd edirik, 3.3. bölməmizdəki, 3.12 (a) şəkilində qeyd olunduğu adları götürə bilərik, Gurban, Vusal, və Javid.

```java
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class EmployeeEvaluation {
    public static void main(String[] args) {
        Map<String, Map<String, Integer>> evaluations = new HashMap<>();

        // Gurban üçün qiymətləndirmə
        Map<String, Integer> gurbanEvaluations = evaluateEmployee("Gurban");
        evaluations.put("Gurban", gurbanEvaluations);

        // Vusal üçün qiymətləndirmə
        Map<String, Integer> vusalEvaluations = evaluateEmployee("Vusal");
        evaluations.put("Vusal", vusalEvaluations);

        // Javid üçün qiymətləndirmə
        Map<String, Integer> javidEvaluations = evaluateEmployee("Javid");
        evaluations.put("Javid", javidEvaluations);

        // Qiymətləndirmə nəticələrini göstərilməsi
        for (Map.Entry<String, Map<String, Integer>> entry : evaluations.entrySet()) {
            String employeeName = entry.getKey();
            Map<String, Integer> criteriaEvaluations = entry.getValue();

            System.out.println("Qiymətləndirmə " + employeeName + ":");
            for (Map.Entry<String, Integer> criteriaEntry : criteriaEvaluations.entrySet()) {
                String criterion = criteriaEntry.getKey();
                int rating = criteriaEntry.getValue();

                System.out.println("- " + criterion + ": " + rating);
            }
            System.out.println();
        }
    }
```

Şəkil 3.17 İşçi parametrlərin Orangeline inteqrasiyası

Qiymətləndirilmə 10 ballıq sistem üzərindən aparılır (Şək. 3.18), lakin müəssisədən asılı olaraq dəyişdiirlə bilər. Bəzən 100% qiymətləndirilmə cədvəlindən istifadə olunur.

```
    public static Map<String, Integer> evaluateEmployee(String employeeName) {
        Scanner scanner = new Scanner(System.in);
        Map<String, Integer> evaluations = new HashMap<>();

        System.out.println("Qiymətləndirmə " + employeeName + ":");

        System.out.print("Ünsiyyət bacarıqlarının qiymətləndirilməsi (1-10): ");
        int communicationSkills = scanner.nextInt();
        evaluations.put("ünsiyyət bacarıqları", communicationSkills);

        System.out.print("Stressə davamlılığın qiymətləndirilməsi (1-10): ");
        int stressResistance = scanner.nextInt();
        evaluations.put("stresə davamlılıq", stressResistance);

        System.out.print("Məsuliyyətin qiymətləndirilməsi (1-10): ");
        int responsibility = scanner.nextInt();
        evaluations.put("məsuliyyət", responsibility);

        System.out.print("Özünə inamın qiymətləndirilməsi (1-10): ");
        int selfConfidence = scanner.nextInt();
        evaluations.put("özünə inam", selfConfidence);

        System.out.print("Yüksək öyrənmə qabiliyyəti qiymətləndirilməsi (1-10): ");
        int learningAbility = scanner.nextInt();
        evaluations.put("yüksək öyrənmə qabiliyyəti", learningAbility);

        System.out.println();

        return evaluations;
    }
}
```

Şəkil 3.18 İşçilərin qiymətləndirilməsinin aparılması

Düşünürəm ki təklif etdiyimiz üsulların istifadəsi, Java proqramlaşdırılma dili üzərində yazılmış kodlar və digər bu işdə öz əksini tapan araşdırma və təhlillər, müəssisələrin inkişafına müsbət təsir göstərəcəkdir.

NƏTİCƏ

Bu işdə, banklarda müştəri xidməti fəaliyyətinin əsas prinsipləri və funksiyaları ilə tanış olduq, fəaliyyətin müəyyənləşdirilməsi üçün parametrlər, xidmət mərkəzlərinin iş prosesinə təsir göstərən amillər, bankların müştəri xidməti fəaliyyətinin qiymətləndirilməsinin üçün tətbiqlər və platformalar haqqında məlumatı araşdırdıq.

Bir neçə platfromanı araşdırdıqdan sonra, OrangeLine platfromasında uyğun işlər aparacağımızı təyin etdik, çünki hazırda Azərbaycaan banklarının, həmçinin digər qurumların böyük bir hissəsi bu tətbiqi istifadə edir. OrangeLine tətbiqi Java proqramlaşdırılma dilində yazılmışdır. Burada geniş yayımış Java Spring ilə əlaqə mövcuddur. Daha sonra, Hibernate, Apache Struts və Apache Tiles vasitələrin inteqrasiyası üsulu ilə təklif olunan metodlarımızı təhlil etdik.

İşçilərin qiymətləndiirlməsi üçün geniş təhlil aparıb çoxsaylı paramterlər seçdik və həmin parametrləri nəzərə alaraq, Java proqramlaşdırma dilində işçinin fəaliyyətinin çoxfunksiyalı qiymətləndirilməsi üçün kodu hazırladıq. Təklif olun üsulun daha mükəmməl çalışması üçün, yazılmış koda, SOLID, DRY, TDD təcrübələrin tətbiqini əlavə etdik.

Düşünürəm ki, banklarda müştəri xidməti fəaliyyətinin qiymətləndirilməsi məqsədi ilə təklif etdiyimiz, çox parametrli qiymətləndirilmə yolu ilə Orangeline-nin təkminləşdirilməsi üsulu çox faydalı olacaqdır. Aparılan araşdırmalar, təklif etdiyimiz üsulların istifadəsi, Java proqramlaşdırılma dili üzərində yazılmış kodlar və digər araşdırma və təhlillərimiz, müəssisələrin xidmət mərkəzlərinin inkişafına müsbət təsir göstərəcəyinə və gələcəkdə bu mövzuya yaxın və ya bu mövzu istiqamətində aparılan işlərə faydalı olacağını düşünürəm.

İstifadə olunmuş ədəbiyyatların siyahısı

1. Arnold, Ken, James Gosling, David Holmes. The Java™ Programming Language, Third Edition. Addison-Wesley, Boston, 2000. ISBN: 0201704331.
2. Beck Kent. Extreme Programming Explained: Embrace Change. Addison-Wesley, Reading, MA, 1999. ISBN: 0201616416.
3. The Java® Language Specification Java SE 8 Edition James Gosling Bill Joy Guy Steele Gilad Bracha Alex Buckley 2015-02-13// 314 - 318 s.
4. Vermeulen, Allan, Scott W. Ambler, Greg Bumgardener, Eldon Metz, Trevor Mesfeldt, Jim Shur, Patrick Thompson. The Elements of Java™ Style, Cambridge University Press, Cambridge, United Kingdom, 2001. ISBN: 0521777682.
5. Core JavaTM 2, 7th Edition, Volumes I & II, by Horstmann & Cornell (Prentice Hall, 2005). Huge, comprehensive, and the first place I go when I'm hunting for answers. The book I recommend when you've completed Thinking in Java and need to cast a bigger net.
6. Thinking in Java Fourth Edition Bruce Eckel President, MindView, Inc. 2018 y. renew, 234-257 s.
7. Java™ in a Nutshell, Fifth Edition by David Flanagan Copyright © 2005, 2002, 1999, 1997, 1996 O'Reilly Media, Inc. All rights reserved. Printed in the United States of America.
8. Boyatsis R. Səlahiyyətli menecer. Effektiv iş modeli.-M.: GİPPO, 2016. - 352 S.
9. Odegov Yu, Kotova L. insan resurslarının idarə edilməsinə yanaşmalar və onların kadrlarla işin səmərəliliyinin qiymətləndirilməsinə təsiri // kadr zabiti. – 2017. - № 2. - S.5-10.
10. Spencer M., Spencer S. işdəki səriştələr. - M.: Elm, 2005. - 384 S.
11. Gorbunova O. A., Kravchenko O. V. pərakəndə xidmətlər bazarında kommersiya bankının strateji fəaliyyət istiqamətləri / / Samara bələdiyyə İdarəetmə İnstitutunun bülleteni. – 2019. - № 2. - S. 56-66.

12. Karpova T. P. insan resurslarının idarə edilməsinin tərkib hissəsi kimi kadr ehtiyatının formalaşdırılması: inkişafın mahiyyəti və istiqamətləri //Samara bələdiyyə İdarəetmə İnstitutunun bülleteni. - 2018. - № 2. - S. 32-40.
13. Kravchenko O. V., Gorbunova O. A. kommersiya bankında xərclərin azaldılması strategiyası / / Samara bələdiyyə İdarəetmə İnstitutunun bülleteni. – 2018. - № 2. - S.58-66.
14. Tuguskina G. müəssisələrin insan kapitalının dəyərinin qiymətləndirilməsi / / kadr zabiti. - 2017. - №11. -S.10-15.
15. Fletcher K. Performance Appraisal. Qiymətləndirmə və rəy. (İş səmərəliliyinin praktik aspektləri). - M.: HİPPO PUBLİSHİNG, 2016. - 288 s.
16. ZHmaeva Y.V., Udovenko S.G., Chalaya L.E. Adaptivnoe prognozirovanie optimal'nogo kolichestva resursov IT proekta po metodologii Agile // ASU i pribory avtomatiki, 2015. – №173. – [EHlektronnyj resurs]. URL: http://cyberleninka.ru/article/n/adaptivnoe-prognozirovanie-optimalnogo-kolichestva-resursov-it-proekta-po-metodologii-agile.
17. Zhidchenko S.I., Dubovik T.M. Razrabotka programmnogo modulya monitoringa ostatochnyh znanij studentov po disciplinam kafedry specializirovannyh komp'yuternyh sistem // FMO, 2016. – №4 (10). – [EHlektronnyj resurs]. URL: http://cyberleninka.ru/article/n/rozrobka-programnogo-modulya-monitoringu-zalishkovih-znan-studentiv-z-distsiplin-kafedri-spetsializovanih-komp-yuternih-sistem.
18. Kozhombaeva A.T., Zotin A.G. Realizaciya programmnogo kompleksa provedeniya inventarizacii s ispol'zovaniem mobil'nyh ustrojstv // Aktual'nye problemy aviacii i kosmonavtiki, 2015. – №11. – [EHlektronnyj resurs]. URL: http://cyberleninka.ru/article/n/realizatsiya-programmnogo-kompleksa-provedeniya-inventarizatsii-s-ispolzovaniem-mobilnyh-ustroystv
19. Spring Framework [Electronic resource]. n.d. Date of modification: 4.10.2017. URL: https://ru.wikipedia.org/wiki/Spring_Framework

20. Spring Framework [Electronic resource]. n.d. URL: http://projects.spring.io/spring-framework/#quick-start.
21. Crimean Engineering and Pedagogical University per. Uchebniy, 8, Simferopol, Republic of Crimea, 29015.
22. Java Platform Enterprise Edition Specifications, version 5. http://java.sun.com/j2ee/5.0/index.jsp.
23. Web-sayt layihə Apache Struts http://struts.apache.org/.
24. C. Cavaness. Programming Jakarta Struts. O'Reilly, 2002.
25. Java Server Faces Specification, version 1.2. http://java.sun.com/j2ee/javaserverfaces/download.html.
26. H. Bergsten. JavaServer Faces. O'Reilly, 2004.
27. Xüsusiyyətlər J2EE 5.0. http://java.sun.com/javaee/5/javatech.html.
28. Web-sayt layihə Hibernate http://www.hibernate.org/.
29. C. Bauer, G. King. Hibernate in Action. Manning, 2004.
30. B. A. Tate, J. Gehtland. Better, Faster, Lighter Java. O'Reilly, 2004.
31. Web-sayt texnologiya JDO http://jdocentral.com/.
32. D. Jordan, C. Russell. Java Data Objects. O'Reilly, 2003.
33. Enterprise Java Beans 3.0. xüsusiyyətləri http://java.sun.com/products/ejb/docs.html.
34. Web-sayt təminat Spring http://www.springframework.org/.
35. R. Johnson. Expert One-on-One J2EE Design and Development. Wrox, 2002.
36. R. Johnson, J. Hoeller, A. Arendsen, T. Risberg, C. Sampaleanu. Professional Java Development with the Spring Framework. Wiley, 2005.
37. http://developer.mozilla.org/en/docs/Category:AJAX:Articles.
38. D. Crane, E. Pascarello, D. James. Ajax in Action. Manning, 2005.
39. G. Alonso, F. Casati, H. Kuno, V. Machiraju. Web Services. Concepts, Architectures and Applications. Springer-Verlag, 2004.
40. H. Daytel, P. Daytel, S. Santri. Java 2 proqramlaşdırma texnologiyaları. Kitab,3: korporativ sistemlər, servletlər, JSP, veb xidmətləri. M.: Binom, 2003.

41.https://abbtech.az/az/spring-framework-haqqinda
42.https://java-blog.ru/osnovy/java-spring
43.https://practicum.yandex.ru/blog/framework-spring-java/#id4
44.https://tproger.ru/articles/spring-modules-overview/
45.http://www.inf.ethz.ch/personal/alonso/WebServicesBook.
46.https://habr.com/ru/articles/490586/
47.https://javarush.com/groups/posts/spring-framework-java-1
48.https://dev.to/urunov/spring-framework-architecture-and-runtime-components-31id
49.https://www.codejava.net/frameworks/spring/understanding-the-core-of-spring-framework
50.https://docs.spring.io/spring/docs/4.3.x/spring-framework-reference/html/overview.html
51.https://dzone.com/articles/spring-framework-tutorial-for-beginners-2
52.https://github.com/Urunov/Spring-DAO-ORM-JEE-Web-AOP-Core-Boot
53.https://offlinecrm.ru/top-10-luchshih-programmnyh-instrumentov-crm-v-2022-godu-poslednie-rejtingi/#Sravnenie_lucsih_instrumentov_CRM
54.https://www.kp.ru/guide/otsenka-personala.html#kompetencij
55.https://academy.mediasoft.team/article/instrumenty-razrabotchika-na-java-spring-hibernate-mybatis-i-drugie/
56.https://coderlessons.com/tutorials/java-tekhnologii/vyuchit-hibernate/hibernate-kratkoe-rukovodstvo
57.https://nuancesprog.ru/p/9525/
58.https://coderlessons.com/articles/java/vvedenie-v-java-tdd-chast-1
59.https://fetisovvs.blogspot.com/2018/12/tdd-spring-boot-java.html
60.https://itnan.ru/post.php?c=1&p=433958
61.https://question-it.com/questions/6187164/dry-minimizatsija-povtorjajuschegosja-koda-v-java
62.https://java-blog.ru/osnovy/struts-2-java
63.https://robotdreams.cc/course/razrabotka-cherez-testirovanie-tdd-na-java