Міністерство освіти і науки України  
Одеський національний політехнічний університет  
Навчально-науковий інститут комп’ютерних систем  
Кафедра системного програмного забезпечення

Чебан Данило Вітальович

студент групи АС-151

**КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА**

Програмна система для підбору кандидатів на вакансії у крюінговій агенції

Спеціальність:  
121 – Інженерія програмного забезпечення

Спеціалізація:  
Інженерія програмного забезпечення

Керівник:

Зіноватна Світлана Леонідівна,

кандидат технічних наук, доцент

Одеса – 2020

**ЗМІСТ**

[ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ 4](#_heading=h.gjdgxs)

[ЗАВДАННЯ на розробку розділу «Охорона праці та безпека в надзвичайних ситуаціях» 6](#_heading=h.1fob9te)

[ВСТУП 7](#_heading=h.2et92p0)

[1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ 11](#_heading=h.tyjcwt)

[1.1 Опис предметної області 11](#_heading=h.3dy6vkm)

[1.2 Технології відбору персоналу 14](#_heading=h.1t3h5sf)

[1.3 Аналіз існуючих аналогів 16](#_heading=h.4d34og8)

[1.4 Пошук за багатьма критеріями 19](#_heading=h.2s8eyo1)

[1.5 Висновки до розділу 22](#_heading=h.17dp8vu)

[2 МОДЕЛЬ ПРЕДМЕТНОЇ ОБЛАСТІ 24](#_heading=h.3rdcrjn)

[2.1 Опис даних 24](#_heading=h.26in1rg)

[2.2 Опис алгоритму отримання результуючої множини кандидатів 26](#_heading=h.lnxbz9)

[2.3 Висновки до розділу 29](#_heading=h.35nkun2)

[3 ВИМОГИ ДО ПРОГРАМНОГО ПРОДУКТУ SELECTCREW 30](#_heading=h.1ksv4uv)

[3.1 Підстава для розробки 30](#_heading=h.44sinio)

[3.2 Функціональні вимоги до програми для підбору кандидатів на вакансії у крюінговій агенції 30](#_heading=h.2jxsxqh)

[3.3 Умови експлуатації 41](#_heading=h.z337ya)

[3.4 Стадії й етапи розробки 42](#_heading=h.1y810tw)

[3.5 Висновки до розділу 42](#_heading=h.4i7ojhp)

[4 ПРОЕКТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ SELECTCREW 43](#_heading=h.2xcytpi)

[4.1 Структура даних 43](#_heading=h.1ci93xb)

[4.2 Архітектура програмної системи 55](#_heading=h.3whwml4)

[4.3 Висновки до розділу 60](#_heading=h.2bn6wsx)

[5 РЕАЛІЗАЦІЯ ПРОГРАМИ ДЛЯ КРЮІНГОВОЇ АГЕНЦІЇ 61](#_heading=h.qsh70q)

[5.1 Класи програми 61](#_heading=h.3as4poj)

[5.2 Робота користувача з програмою 68](#_heading=h.1pxezwc)

[5.3 Висновки до розділу 74](#_heading=h.49x2ik5)

[6 ТЕСТУВАННЯ ПРОГРАМНОГО ПРОДУКТУ 75](#_heading=h.2p2csry)

[6.1 Функціональне тестування 76](#_heading=h.147n2zr)

[6.2 Нефункціональне тестування 80](#_heading=h.3o7alnk)

[6.3 Експериментальне дослідження досягнення мети проекту 80](#_heading=h.23ckvvd)

[6.4 Висновки до розділу 81](#_heading=h.ihv636)

[7 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА У НАДЗВИЧАЙНИХ СИТУАЦІЯХ 82](#_heading=h.32hioqz)

[7.1 Опис розглянутих питань охорони праці 82](#_heading=h.1hmsyys)

[7.2 Висновки за розділом 83](#_heading=h.41mghml)

[ВИСНОВКИ 84](#_heading=h.2grqrue)

[СПИСОК ЛІТЕРАТУРИ 85](#_heading=h.vx1227)

[ДОДАТОК А КОД ПРОГРАМИ 88](#_heading=h.1v1yuxt)

[ДОДАТОК Б ПИТАННЯ З ОХОРОНИ ПРАЦІ ТА БЕЗПЕКИ У НАДЗВИЧАЙНИХ СИТУАЦІЯХ 106](#_heading=h.4f1mdlm)

**Міністерство освіти і науки України  
Одеський національний політехнічний університет  
Навчально-науковий інститут комп’ютерних систем  
Кафедра системного програмного забезпечення**

Рівень вищої освіти: другий (магістерський)  
Спеціальність: 121 – Інженерія програмного забезпечення  
Спеціалізація: Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ

Завідувач кафедри

проф. Крісілов В.А.

" " 2020 р.

**ЗАВДАННЯ  
НА КВАЛІФІКАЦІЙНУ РОБОТУ**

Чебана Даніли Вітальовича, АС-151

(прізвище, ім'я та по батькові)

1. Тема проекту (роботи) Програмна система для підбору кандидатів на вакансії у крюінговій агенції

Керівник проекту Зіноватна Світлана Леонидівна, канд. техн. наук, доцент затверджені наказом ректора від "29"жовтня 2020 р.№412-в

2. Строк подання студентом проекту (роботи) 10.12.2020

3. Вихідні дані по проекту (роботі) Технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належіть розробити) Вступ. Аналіз предметної області. Модель предметної області. Вимоги до програмного продукту.Проектування програмної системи. Реалізація програми.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

Мета та задачі роботи. Порівняння функціональних характеристик продуктів-аналогів.

Опис даних. Алгоритм отримання результуючої множини кандидатів. Діаграма варіантів використання. Модель «Сутність – Зв’язок» для предметної області. Архитектура ПЗ. Діаграма програмних класів. Приклади віконних форм програми. Результати експериментального дослідження. Висновки

6. Консультанти розділів проекту (роботи)

|  |  |  |  |
| --- | --- | --- | --- |
| Розділ | Консультант | Підпис, дата | |
| Завдання видав | Результат прийняв |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

7. Дата видачі завдання 28.08.2020

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Назва етапів дипломного проекту  (роботи) | Строк виконання етапів проекту  (роботи) | Примітка |
| 1. | Вивчення предметної області | 10.09.2020 | виконано |
| 2. | Аналіз вимог | 25.09. 2020 | виконано |
| 4. | Розробка структури даних | 04.10. 2020 | виконано |
| 5. | Розробка алгоритмів | 21.10. 2020 | виконано |
| 6. | Розробка програмного забезпечення | 15.11. 2020 | виконано |
| 7. | Дослідна експлуатація | 25.11. 2020 | виконано |
| 8. | Оформлення пояснювальної записки | 05.12. 2020 | виконано |

**Здобувач вищої освіти**  *Д. В. Чебан*

**Керівник роботи**  *С. Л. Зіноватна*

**ЗАВДАННЯ  
на розробку розділу «Охорона праці та безпека в надзвичайних ситуаціях»**

здобувач Чебан Данило Вітальович

**(прізвище, ім’я, по батькові)**

Інститут, факультет Інститут комп’ютерних систем

**(**повне найменування інституту, факультету**)**

Кафедра Системного програмного забезпечення

**(**повне найменування кафедри**)**

Тема роботи Програмна система для підбору кандидатів на вакансії у крюінговій агенції

Зміст розділу:

1. Аналіз умов праці і вибір заходів і засобів захисту від небезпечних і шкідливих виробничих факторів.

2. Аналіз техногенних небезпек і вибір заходів і засобів забезпечення безпеки у надзвичайних ситуаціях.

3. Індивідуальне завдання

|  |  |  |  |
| --- | --- | --- | --- |
| Керівник роботи | | Консультант з охорони праці та безпеки в надзвичайних ситуаціях | |
| \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_Зіноватна С.Л.\_\_  ( підпис ) (прізвище та ініціали)  « \_\_ » \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2020 р. | \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_Москалюк А.Ю.\_\_  ( підпис ) (прізвище та ініціали)  « \_\_ » \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2020 р. | |

**ВСТУП**

Правильне використання робочої сили відповідно до її підготовки, оцінка цієї відповідності - один з резервів підвищення ефективності виробництва, росту продуктивності праці. Головним завданням у кадровій політиці є оцінка властивостей і якостей особистості, ступеня відповідності їх вимогам займаної посади. Відбір персоналу - частина процесу наймання персоналу, пов'язана з вибором одного або декількох кандидатів на вакантну посаду серед загального числа людей, що претендують на дане робоче місце (загальноприйняте трактування терміну). Працівники кадрових служб давно відчували потребу в більш обґрунтованих і надійних процедурах оцінки й відбору. Підвищення ефективності й надійності відбору припускає послідовну перевірку ділових і особистісних якостей кандидата, по взаємодоповнюючих методах і технологіям. Звичайно здійснюється поетапний відбір кандидатур. Щораз відсівають тих, хто виявив явну невідповідність пропонованим вимогам. Одночасно застосовують, по можливості, об'єктивну оцінку фактичних знань і ступеня оволодіння кандидатом необхідних виробничих навичок. У такий спосіб формується складна багатоступінчаста система проведення відбору людських ресурсів [1].

Ефективність роботи служби керування персоналом у значній мірі залежить від ефективності процедури підбора персоналу. Для компенсації або усунення помилок, допущених на цьому етапі, організації доводиться нести значні додаткові витрати, такі, наприклад, як витрати на повторний підбор персоналу, витрати на адаптацію, навчання й т. д. [2]

На думку М. Армстронга, «кінцева мета процесу підбора персоналу повинна полягати в тому, щоб з мінімальними витратами найняти певну кількість працівників певної якості для задоволення потреб компанії в людських ресурсах» [3].

«Відбір персоналу – дуже важливий і відповідальний момент в управлінні кадрами. Завдання менеджера - знайти і вибрати кращого співробітника, це вимагає наявності високої кваліфікації і солідного професійного досвіду. Помилки менеджера при відборі персоналу можуть негативно відбитися на ефективності колективної роботи в організації» [4].

Основними витратами на підбор персоналу є витрати на оплату ЗМІ для залучення кандидатів, оплата робіт кадрових агентств по підборі персоналу, оплата роботи самої служби управління персоналу й т.д. [3]

Результати, які повинні бути досягнуті на різних етапах підбора [2]:

1. етап пошуку й залучення кандидатів - чисельність притягнутих кандидатів з характеристиками, що відповідають вимогам вакансії (чим їх більше, тим більше вибір);
2. етап оцінки й відбору притягнутих кандидатів - число що звільнилися/звільнених співробітників протягом року.

Ідеальна ефективність - це 100% економічності, тобто відсутність витрат на підбір персоналу, і 100% результативності (немає нових співробітників, що звільнилися протягом першого року роботи).

Якщо якийсь час назад економіка трималася в основному за рахунок трудових ресурсів, підготовлених і соціалізованих ще в радянський час, то в цей час питання кадрового менеджменту, у тому числі й питання наймання й підбора персоналу, актуалізуються - підприємства мають потребу в якісно новій робочій силі. Труднощі з підтвердженням кваліфікації пов'язані з появою великої кількості випускників нових бізнес-установ освіти, які не мають устояної репутації [6].

Метою роботи є зменшення кількості помилок при обранні кандидата на вільну посаду співробітниками крюінгової агенції за рахунок багатокритеріального пошуку персон з бази даних агенції.

Для досягнення мети потрібно вирішити такі задачі:

* визначити критерії, які повинні бути враховані під час обрання кандидатів на посаду;
* проаналізувати функціональність існуючих автоматизованих платформ для визначення кандидатів на вакансію;
* розробити модель предметної області підбору персоналу для роботи на судах;
* спроектувати та створити базу даних для зберігання потрібної інформації,
* визначити алгоритми виконання функції визначення списку кандидатів, що задовольняють множині критеріїв.

Об'єктом дослідження є інформаційні платформи підбору персоналу.

Предметом дослідження є засоби багатокритеріального пошуку.

Методи дослідження. Результати, отримані в кваліфікаційній роботі, базуються на теорії множин, принципах об’єктно-орієнтованого проектування.

Наукова новизна полягає в наступному: вперше надано множинне представлення предметної області пошуку кандидатів на посаду, яке формалізує структуру даних та алгоритми пошуку за основними та додатковими критеріями.

Науково-практична цінність полягає в тому, що автоматизовано пошук людей для зайняття посади з підвищеними формальними вимогами до складу наявних документів .

Апробація результатів роботи. Основні результати роботи доповідалися й обговорювалися на XIIІ міжнародній науково-практичній конференції «Інформаційні технології і автоматизація – 2020».

У першому розділі описуються технології, що застосовуються для відбору персоналу на вакантні посади, та виконаний огляд характеристик деяких програмних продуктів для роботи з кандидатами на посади, представлених на ринку IT.

У другому розділі детально надані упорядковані основні та додаткові критерії відбору кандидатів на посаду, специфічні для розглянутої предметної області, яка описана у вигляді множинної моделі. Також у розділі наведений алгоритм використання набору критеріїв для отримання множини персон, які можуть розглядатися у якості можливих кандидатів.

У третьому розділі описані вимоги до програмної системи, описані сценарії окремих варіантів використання, розглянуті основні етапи розробки.

У четвертому розділі на основі формалізованого представлення предметної області розроблена структура БД, яка містить набор нормалізованих таблиць для збереження інформації про об’єкти предметної області. Описана архітектура програмного забезпечення, обґрунтований вибір шаблону модель-представлення-контролер.

У п’ятому розділі надана структура класів, які використаються для реалізації функцій програми, детально описані методи класів. Показано, як програма взаємодіє з користувачем, за допомогою яких засобів можна переглядати та змінювати дані.

У шостому розділі описано, яким чином проведено тестування програми, а також описаний експеримент, що підтвердив досягнення мети розробки.

У сьомому розділі розглянуті питання з охорони праці та безпеки у надзвичайних ситуаціях, а саме – основні вимоги до організації охорони праці, управління охороною праці, надзвичайні ситуації та шляхи їх запобігання. В ході роботи проаналізовано робоче місце інженера-програміста, вимоги до інтервалів робочих змін, необхідний рівень освітленості при умовах роботи на моніторі високої точності, та режим відпочинку.

Впровадження розробленого програмного забезпечення дозволить зменшити кількість помилок при відборі кандидатів, що може скоротити затрати робочого часу працівників агенції на перевірку даних кожної персони, а також не дасть пропустити найбільш відповідних критеріям персон.

**1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ**

**1.1 Опис предметної області**

Підбор персоналу в компанію є складним завданням. Необхідно проаналізувати ситуацію не тільки усередині компанії, але й за її межами, тобто на ринку праці. Ринок праці - це місце, де відбуваються угоди між здобувачем і наймачем персоналу. Через цей ринок відбувається продаж на певний строк робочої сили. Основною особливістю ринку праці є те, що об'єктом купівлі-продажу вважається право на використання робочої сили, кваліфікацій, знань і здатностей до трудової діяльності.

По визначенню Міжнародної організації праці, ринок праці - сфера, де підприємці й трудящі ведуть спільні переговори колективного або індивідуального характеру щодо заробітної плати й умов праці [7].

У більше глобальному змісті ринок праці - це система юридичних і соціально-економічних відносин у суспільстві, інститутів і норм, які покликані забезпечити повноцінний безперервний процес відтворення робочої сили й максимально ефективне використання праці.

Основні компоненти ринку праці:

* попит на робочу силу і її пропозиції;
* ціна й вартість робочої сили;
* конкуренція між трудящими й роботодавцями.

Основні суб'єкти ринку праці:

* роботодавці (або їхні союзи);
* працюючі (або їхні профспілки);
* держава і її органи: державна служба зайнятості, міністерство праці й соціального захисту населення й так далі;
* рекрутінгові й кадрові агентства;
* освітні інститути;
* суспільні інститути;
* ЗМІ й Інтернет.

Ринок праці непостійний, він увесь час коливається, тому що відбуваються зміни в попиті та пропозиціях. В умовах вільної конкуренції попит заснований на реальній зарплаті й вартості граничного продукту праці (зробленим останнім найнятим працівником). А пропозиція праці залежить, насамперед від рівня його оплати, при цьому, чим вище заробітна плата, тим більше пропозицій робочої сили. Коливання на ринку праці також пов'язані з мобільністю трудових ресурсів, тобто можливістю працівників і членів їхніх родин переїхати в іншу місцевість, де вони можуть знайти більше вигідну пропозицію про роботу [7].

У спрощеному варіанті процес підбора персоналу виглядає наступним чином:

* скласти докладний опис позиції з обліком необхідних особистих якостей і кваліфікаційних вимог;
* розробити план рекламної кампанії для залучення кандидатів;
* скласти докладні плани проведення інтерв'ю на всіх етапах;
* у міру надходження, вчасно обробляти резюме й проводити телефонні інтерв'ю з кандидатами;
* провести особисті інтерв'ю з кандидатами й прийняти рішення.

Деякі типові помилки, які може зробити погано підготовлений виконавець підбору персоналу:

* некоректна оцінка особистих якостей, результат: новий співробітник через відсутність необхідних особистих якостей є неефективним;
* не приділяється належної уваги мотивації, результат: новий співробітник з недостатньою мотивацією надовго не затримається.
* некоректно зроблена підготовка й погано проведені інтерв'ю, результат: відсутність повної картини про кандидата, неможливість ухвалення правильного рішення.

Роботодавці при пошуку співробітника виявляють наступні параметри майбутнього працівника: особистісні або психофізіологічні якості; соціальний статус; культура, релігія й політичні погляди; знання, уміння й навички; інтереси й хобі; мотивація до праці; трудова активність.

Робота дозволяє співробітникові реалізувати себе й свої мети. Зарплата працівника стає не тільки ціною за ресурс, але й вартістю певного рівня життя, його соціального статусу й найближчого оточення [8].

Для зручності аналізу й оцінки ринкових відносин проводиться умовний поділ або сегментація.

Приводом розподілу на сегменти є різний рівень економічної й соціальної ефективності виробництва. Підрозділяють працівників і робітники місця в окремі стійкі спільності.

Первинною називають таку систему відносин, яка стабільна за рівнем зайнятості. Працівники тут одержують високу заробітну плату й мають можливість кар'єрного й професійного росту й просування. Ринкові відносини характеризуються високим розвитком технологій і систем керування.

Вторинний ринок - це система відносин між працівником і роботодавцем, у якій плинність персоналу висока, зайнятість непостійна, рівень доходу низький, можливостей для кар'єрного росту й професіоналізму майже не передбачено, технології відстають, профспілки не сформовані.

Внутрішній ринок - це трудові відносини в одній єдиній компанії, усередині якої ціна роботи й поділ обов'язків регламентовані внутрішніми документами й правилами. У компанії дотримуються наступних правил: склад і кількість працівників організації; рух працівників усередині підприємства і його причини; рівень зайнятості співробітників; використання встаткування; вільні, нові й робочі місця, що скорочують.

Зовнішній ринок - це відносини на рівні країни, регіону або галузі економіки. Тут передбачається розміщення працівників по сферах і їхнє переміщення між різними роботодавцями. Плинність кадрів забезпечує змінюваність працівників і впливає на появу безробіття.

**1.2 Технології відбору персоналу**

В [9] перелічені основні підходи, що використаються при підборі персоналу:

1. інтерв’ю, яке може бути інтелектуальним, стресовим та brainteaser-інтерв’ю (може бути перекладено як інтерв'ю, що лоскоче мозок); перший варіант інтерв’ю дозволяє виявити високо мотивованих співробітників, новаторів, здатних чітко й логічно мислити; метою другого варіанту є можливість визначити стресостійкість кандидата, необхідність його використання може бути виправдана при підборі працівників на певні вакансії; суть третього варіанту інтерв’ю в тім, що кандидатам необхідно дати відповідь на складне питання або вирішити логічне завдання, мета такого нестандартного методу – перевірити аналітичне мислення й творчі здатності здобувача;
2. headhunting (хедхантінг) та executive search; перший варіант - конфіденційні діалоги про можливості переходу конкретного фахівця з однієї компанії в іншу, це технологія підбора персоналу, спрямована на переманювання певного фахівця з компанії-донора в компанію клієнта; другий варіант - пошук менеджерів вищої ланки й рідких фахівців серед успішно діючих, це технологія, що включає в себе унікальні методи виявлення потенційних кандидатів на всьому заданому сегменті ринку праці, що не перебувають в активному пошуку роботи;
3. рекрутінг, зараз так для стислості називають пошук і підбор кваліфікованих фахівців на платній основі; на відміну від послуг із працевлаштування, коли агентство одержує оплату зі здобувача роботи, рекрутінгове агентство заключає договір і, відповідно, одержує гонорар з фірм-замовників; робота консультантів рекрутінгових фірм у полягає в перегляді вступників анкет і резюме, у відсіванні свідомо невідповідних кандидатів, у проведенні співбесід з найбільш кваліфікованими й у наданні максимально короткого списку претендентів (3-5 чіл.) фірмі замовнику;
4. скринінг, це масовий підбор кандидатів шляхом фільтрації всього потоку резюме здобувачів по формальних вимогах (стать, вік, освіта, досвід роботи, готовність розглядати пропозиції про роботу), найчастіше використається при підборі лінійних фахівців; скринінг відрізняється від послуги рекрутінга повного циклу, тому що агентство по підборі кандидатів тут не проводить глибокої й детальної оцінки кандидатів, не перевіряє рекомендації з минулих місць роботи, не дає гарантійних зобов'язань, у випадку невідповідності, що виявилося, необхідним для займаної позиції професійним і особистісним вимогам рекомендованих кандидатів;
5. відеорезюме, багато роботодавців уважають, що людям творчих професій і працюючих у сфері «людина-людина», відеорезюме може допомогти виділитися з множини кандидатів; основна ціль створення відеорезюме - привернути увагу роботодавця своєї креативністю й неординарністю;
6. соціальні мережі, деяких фахівців цікавить не тільки добре складене резюме й здатності кандидата до самопрезентації, але які особистісні якості здобувача, його захоплення, інтереси, шкідливі звички. Тому практика вивчення профілю кандидата в соціальних мережах стає усе популярнішою;
7. соціоніка - наука, що вивчає процес переробки інформації з навколишнього світу психікою людини; соціоніка опирається на такі поняття, як логіка й етика, інтуїція й сенсоріка, екстраверсія й інтроверсія, раціональність і ірраціональність, названі функціями психіки.

**1.3 Аналіз існуючих аналогів**

**1.3.1 Платформа CVpool.** CVpool — платформа, яка спрощує пошуки кандидатів і дозволяє ефективно працювати де завгодно. Кандидату надсилається доступ до сервісу у вигляді посилання, а він відповідає на заздалегідь створені питання [10].

Процедура пошуку кандидатів складаються з наступних кроків:

1. вибір мови співбесіди та фіксація часу для відповіді на кожне питання, на одну вакансію створюється один перелік питань, які кандидат побачить під час запису відео;
2. внесення даних про прізвище та пошту кандидатів, яка зберігається у бібліотеці;
3. відправлення запрошення на інтерв’ю усім кандидатом одночасно, є можливість звертатися до збережених кандидатів повторно для інших вакансій;
4. питання з’являються тільки під час запису відео без можливості перезаписати відповідь, відповіді відразу відображаються на сторінці вакансії;
5. завантажені відео співбесід можна переглядати в зручний час, можливо виставляти оцінки та коментувати.

Переваги платформи:

1. можливість отримувати відео-відповідь кандидата будь-де та будь-коли;
2. зупиняти перегляд відеобесіди в будь-який момент, якщо кандидат не відповідає вимогам клієнта;
3. ділитися та переглядати співбесіду з людьми, які зацікавлені у пошуку кандидата;
4. зберігати дані і відео кандидатів в бібліотеці, щоб переглядати записи або повторно звертатися до претендента;
5. формувати перелік необхідних питань, які кандидат побачить у визначеній послідовності в момент запису відео;
6. встановлювати критерії оцінювання, ставити бали за відповіді, коментувати інтерв’ю кожного кандидата.

**1.3.2 Крюинговий пакет L-Stream Ltd.** L-Stream Ltd надає персонал для роботи на всіх типах судів з необхідними професійними знаннями [11].

Крюинговий пакет містить у собі деякі або усі нижче перераховані послуги, структуровані відповідно до індивідуальних вимог будь-якого клієнта.

Підбор і наймання моряків:

* пошук і відбір кандидатів, які будуть прийняті для роботи на судах партнерів компанії;
* перевірка сертифікатів, включаючи перевірку довідок про плавання;
* перевірка відгуків попередніх компаній відповідно до перевірочного аркуша.

Послуги перед прийомом на роботу:

* оцінка компетентності й рівня розмовної англійської мови та морської термінології;
* оцінка компетентності й досвіду роботи (комп'ютерний тест);
* перевірка досвіду роботи екіпажу кураторами;
* медичний огляд перед прийомом на роботу, включаючи проходження тесту на вживання алкоголю й наркотичних засобів і інші тести відповідно до вимог власників суден;
* впровадження й контроль дотримання політики відносно з алкоголю й наркотичних засобів

Прийом на роботу:

* перевірка документації, рекомендації, інструктаж при прийманні на роботу відповідно до форми компанії;
* інструктаж щодо політики й правил партнерів компанії;
* одержання віз і організація перевезень екіпажу, що заходить на судно й залишає судно;
* інструктаж щодо роботи, безпеки й процедур на судні;
* постачання форми для персоналу;
* одержання робочих сертифікатів для плавання під прапором держави власника судна.

Політика відносно тривалого співробітництва персоналу з компанією:

* моніторинг відгуків-характеристик екіпажу для повторно прийнятих на роботу співробітників, а також додаткове навчання й повторні курси;
* повернення екіпажу відповідно до плану.

**1.3.3 Платформа Crewell.** Платформа надає наступні послуги [12]:

* керування морською кар'єрою через інтернет-сервіс;
* заповнення й редагування CV (анкети моряка);
* завантаження й зберігання моряком документів в особистому кабінеті;
* розсилання CV і морських документів роботодавцям на emaіl;
* розміщення CV моряка в онлайн-базах роботодавців;
* відправлення CV на будь-який emaіl.

Система містить унікальні інструменти, що допомагають морякам особисто управляти своєю кар'єрою. В «особистому кабінеті моряка» можна заповнити й редагувати власну анкету моряка, а також зібрати й зберігати скан копії всіх морських документів.

Щоб знайти бажану роботу в море, є можливість стежити за всіма морськими вакансіями, опублікованими на Crewell.net, і претендувати на підходящі вакансії. Морські роботодавці одержують резюме, як тільки користувач натискає кнопку «Претендувати».

Crewell не містить резюме моряків у відкритому доступі, на відміну від більшості морських інтернет-сайтів і баз даних. Прізвище, дата народження, морський досвід, контакти й інша особиста інформація стає доступна роботодавцям тільки за бажанням кандидата.

Створено сучасну унікальну онлайн анкету моряка, що максимально зручна в заповненні й редагуванні, форма даної анкети моряка високо оцінена морськими роботодавцями. при заповненні анкети моряка не потрібно численних додаткових кроків. Crewell CV схожа на звичайну паперову анкету. Вся інформація зручно й наочно зібрана одній сторінці.

Є можливість використати всі сервіси повідомлень про вакансії: повідомлення по електронній пошті, додаток Crewell Mobіle, повідомлення в соціальних мережах.

Crewell інформує й підтверджує, коли кандидат стає претендентом на морську вакансію. Можна стежити за станом свого CV у власному аккаунте моряка. Можна також побачити, коли роботодавець одержав і переглянув резюме моряка. Якщо роботодавець схвалює або відхиляє яку-небудь кандидатуру, повідомлення також приходить найближчим часом .

**1.4 Пошук за багатьма критеріями**

Надзвичайно широкий і вкрай важливий із практичної точки зору клас завдань вибору становлять багатокритеріальні задачі, у яких якість прийнятого рішення оцінюється по декількох критеріях одночасно. Успішне рішення багатокритеріальних задач неможливо без використання різного роду відомостей про переваги особи, що приймає рішення (ОПР). Постановка всякого завдання багатокритеріального вибору включає три об'єкти: множину можливих рішень, векторний критерій і відношення переваги ЛПР. Вирішити це завдання - означає на основі векторного критерію й наявних відомостей про відношення переваги ЛПР знайти множину обираних рішень. [13].

У задачах прийняття рішень часто виникає необхідність оцінки вибору за багатьма показниками, що характеризує різні сторони їхньої якості й конкретизує поняття «досягнення цілей» [14].

Формулювання цілей рішення проблем виконується звичайно в загальній змістовній формі. Тому конкретизація цілей здійснюється шляхом введення сукупності показників досягнення цілей. Важливою вимогою, пропонованою до показників, є їхня вимірність.

Критерії, застосовувані для рішення багатокритеріальних завдань, повинні відповідати наступним принципам:

* мінімальності - постановочна частина задачі повинна містити як можна менше число критеріїв, а відкидання хоча б одного з них міняє результат;
* вимірності - кожний критерій повинен допускати можливість кількісної (у крайньому випадку, якісної) оцінки ступеня досягнення відповідної мети;
* неповторності, тобто різні критерії не повинні враховувати ту саму сторону рішення;
* чіткості, при якій кожний критерій повинен мати зрозумілу для ОПР формулювання, ясний і однозначний зміст, характеризувати певний аспект рішення.

Можливості людини по переробці багатомірної інформації дуже обмежені, тому ймовірність помилкових дій ОПР досить велика. Можливі два варіанти перевірки дій ОПР.

Перший - критерій несуперечності: в аналогічних ситуаціях оцінки або порівняння альтернатив ОПР повинен приймати однакові рішення. Цей критерій заснований на припущенні про існування в ОПР певної, несуперечливої політики. В реальних ситуаціях так буває не завжди.

Другий критерій для оцінки діяльності ОПР - критерій транзитивності, що для трьох альтернатив А, В, С можна записати так:

1) Якщо А > В, B > C, те A>C (для відношення переваги);

2) Якщо А=В, В=С, то А=С (для відношення байдужності).

При порушенні транзитивності виникає порочне коло типу A>B, B>C, C>A і виділити кращий варіант неможливо.

Як і у випадку з першим критерієм, в реальних ситуаціях можлива поява нетранзитивності в перевагах ОПР.

Відомо, що для людини складними є:

* завдання з великим навантаженням на понятійний апарат;
* завдання, що вимагають паралельної обробки багатомірної інформації;
* завдання порівняння альтернатив з малими рівнями розрізнення;
* завдання, що мають значне число критеріїв (5-6-7 і більше);
* показники зі значним числом оцінок на шкалах критеріїв (5 і більше);
* показники з більшим числом результуючих класів рішень (5-6 і більше).

Стандартний прийом боротьби із багатокритеріальним вибором - це перехід до однокритеріальної задачі з використанням методу згортки критеріїв.

Згортка критеріїв означає побудову інтегрального показника на основі приватних критеріїв [15].

Лексикографічний метод припускає, що наявний ряд критеріїв упорядкований по важливості.

Для порівнюваних об'єктів спочатку виміряються значення найбільш важливого критерію. Кращим виявляється той об'єкт, для якого значення цього критерію краще. У тому випадку, коли значення порівнюваних об'єктів по найбільш важливому критерії збігаються, то переходять до порівняння на підставі наступні по важливості критерію. Процедура закінчується на тій ітерації, на якій вдається впорядкувати об'єкти по переваги, або коли проведені порівняння за всіма критеріями.

Найбільшою мірою ідеології багатокритеріального вибору відповідає процедура виділення з Парето (ядра графа).

Множина Парето утворює набір таких об'єктів, що перехід від одного до іншого обов'язково підвищить значення хоча б одного критерію й погіршить значення мінімум одного критерію. Передбачається, що кожний із критеріїв характеризує якісно відмінний від інших аспект, властивість об'єкта й т.п. Оскільки порівняння різноякісних речей не має змісту, то впорядкуванню підлягають тільки ті пари об'єктів, у яких один не гірше іншого по всіх параметрах. Якщо при цьому по одному або декількох критеріях один об'єкт буде краще іншого, то говорять, що він домінує. У множині Парето жоден об'єкт не домінує над іншим. Властиво, процедура знаходження множини Парето й полягає в знаходженні домінуючих об'єктів і їхньому виключенні з розгляду.

**1.5 Висновки до розділу**

В розділі розглянуті різні технології відбору кандидатів на вільну посаду. Задача є особливо важливою в області судноплавства, оскільки праця в цій галузі пов’язана з додатковими вимогами безпеки. Таким чином, обрання кандидата для вакансії потребує від ОПР урахування багатьох критеріїв вибору. Оскільки задача багатокритеріального вибору є складною, то актуальним є використання автоматизованої програмної системи, яка забезпечить усунення помилок, пов’язаних с неувагою людини.

В теперішній час існують різні платформи для підбору кандидатів на вакансії, як загального типу (для будь-якої галузі), так і спеціалізовані (для судноплавства).

В табл. 1.1 наведено порівняльні характеристики розглянутих в розділі окремих програмних платформ та розроблюваної системи SelectCrew.

Таблиця 1.1 – Порівняння платформ для вибору кандидатів на вільну посаду

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Платформа  Характеристика | CVpool | L-Stream Ltd | Crewell | SelectCrew |
| Можливість додання документів | - | + | + | + |
| Можливість пошуку потенційних кандидатів | - | - | - | + |
| Збереження даних про кандидатів | + | + | + | + |
| Зіставлення потрібних документів та вакансії | - | + | - | + |
| Плата за використання | + | + | + | - |

**2 МОДЕЛЬ ПРЕДМЕТНОЇ ОБЛАСТІ**

**2.1 Опис даних**

Предметна область передбачає відбір кандидатів на задану посаду з множини осіб, які присутні у БД компанії.

Нехай *Р* – множина усіх персон, що потенційно можуть розглядатися як кандидати на посаду; *К* – множина відібраних кандидатів.

Відбір виконується за багатьма критеріями. Використовується лексикографічний метод, за яким наявний ряд критеріїв упорядковується по важливості.

Метод розширений тим, що окремі критерії можуть мати дві форми: основну та додаткову. Тобто множина *К* може бути розширена за рахунок використання додаткової форми критерію.

Виділено наступні критерії віднесення особи до множини *К*:

* *С*1 – особа немає діючого контракту;
* *С*2 – діючий контракт завершився *cd* днів потому;
* *С*3 – особа має необхідний досвід;
* *С*4 – особа має пакет необхідних документів.

Критерії *С*2-*С*4 мають додаткові форми:

* *С*2’ – діючий контракт завершився cd’ днів потому:
* *С*3’ – особа має рекомендації на підвищення посади;
* *С*4’ – особа немає повного пакету необхідних документів, але недостатні документи можуть бути підготовлені за наявний час.

Елемент множини *Р* може бути описаний наступним чином:

*p*∈*P*=<*idp*, *DP*, *EP*, *PP*>

де *idp* – ідентифікуючі особу дані (прізвище, ім’я, дата народження, громадянство та інше);

*DP* – множина наявних документів особи;

*EP* – множина даних про досвід роботи особи;

*PP* – множина промоушенів (рекомендацій на підвищення посади).

*EP*=*EO* U *PCP*,

де *EO* – множина даних про досвід роботи у інших компаніях;

*PCP* – множина контрактів у компанії.

Вакансія може бути описана наступним чином:

*v*=<*dstart*1, *dstart*2, *pc*, *ps*, *sh*, *VD*>,

де *dstart*1, *dstart*2 – задають період можливого початку контракту;

*pc* – тривалість контракту, місяці;

*ps* – посада;

*sh* – судно;

*VD* – множина документів, необхідних особі, яка претендує на заняття вакансії.

Для даних встановлені наступні правила.

Кількість днів *cd* для критерію *С*2 встановлюється наступним чином. Обчислюється середній термін контрактів *apc* для посади *ps* судна *sh, cd= apc.* Якщо таких контрактів не має в БД, то значення *cd* встановлюється користувачем. Таким чином, значення *cd=func(ps*, *sh),* тобто є встановлюваною величиною для кожного розрахунку.

Для додаткових критеріїв термін перебування на відпочинку особи *cd* може бути зменшений, тобто особі можна зробити пропозицію заключити контракт раніше.

Елемент множини *DP* також є складеним:

*dp*∈*DP*=<*td*, *dstart*, *dend*>

де *td* – тип документу;

*dstart* – дата видачі документу;

*dend* – дата закінчення дії документу.

Відповідно до дії документу визначається можливість отримання документу за визначений термін. Наприклад, диплом про вищу освіту неможливо отримати швидко, а сертифікат про отримання знань по правилам безпеки – можливо, оскільки тренінги за для отримання окремих сертифікатів тривають кулька днів.

Для документів можливі такі ситуації:

* документу немає зовсім, особа ніколи не отримувала документ такого типу;
* документ є, але в нього закінчився термін дії, тут також можливі два варіанту – документ потрібно отримувати заново або документ може бути подовжений;
* документ ще є діючим, але під час виконання контракту за вакансією його дія буде завершена;
* документ є діючим, під час виконання контракту за вакансією він буде дійсним.

Документ є діючим, якщо виконується умова

*dend< dstart*2 + *(pc+1)\*30*

Додавання 1 необхідно, оскільки звичайно контракт підписується з попередженням затримки або скорочення на 1 місяць.

Однакові посади на різних типах судів можуть мати різні вимоги, що визначається пакетом документів, які повинні бути у наявності у кандидата на займання вакансії.

**2.2 Опис алгоритму отримання результуючої множини кандидатів**

Пошук множини *К* виконується за рахунок послідовного використання критеріїв відбору, впорядкованих за ступенем важливості. На кожному етапі виконується скорочення множини потенційних кандидатів. Але також можливо, що результуюча множина на відповідному кроці виявитися пустою. Тоді виконується повторне визначення відповідної множини з використанням додаткового критерію.

Таким чином, відбір кандидатів виконується за наступним алгоритмом.

1. *К*1=*Р*-*РС*,

де *РС* – множина осіб з діючими контрактами.

1. *К*2*=К*1*-РСЕ,*

де *РСЕ* – множина осіб, у яких контракт завершився менш, ніж *cd* днів потому.

1. Якщо *К*2=∅, то *К*2=*К*1-*РСЕ*’,

де *РСЕ*’ – множина осіб, у яких контракт завершився менш, ніж *cd*’ днів потому.

1. Якщо *К*2=∅, то перейти в п.11.
2. *К*3=*К*2∩*EPS*,

де *EPS* – множина осіб, що мають досвід роботи на посаді *ps* на судах з типом, який зазначений для *sh*.

1. Якщо *К*3=∅, то *К*3=*К*2∩*EPS*’,

де *EPS*’– множина осіб, які мають рекомендації на посаду *ps* на судах з типом, який зазначений для *sh*.

1. Якщо *К*3=∅, то перейти в п.11.
2. *К*4= *К*3∩*PD*,

де *PD* – множина осіб, що необхідний пакет документів *DP*=*VD*.

1. Якщо *К*4=∅, то *К*4= *К*3∩*PD*’,

де *PD*’– множина осіб, у яких *DPS*=*VD*-*DP* містить документи, які можуть бути підготовлені за час до *dstart*1.

1. Якщо *К*4≠∅, то *К*=*К*4.
2. Кінець алгоритму.

Схема алгоритму відображає послідовність наведених кроків (рис. 2.1).



Рисунок 2.1 – Схема алгоритму формування множини кандидатів на вакансію

**2.3 Висновки до розділу**

В розділі наданий детальний опис даних, що дозволяє сформувати необхідну структуру бази даних для збереження необхідної для роботи інформації.

Правила, визначені для даних, з одного боку, дозволяють включити перевірки правильності введених в БД значень, а з іншого боку, визначають запити до таблиць БД, за допомогою яких можливе отримання необхідних даних.

Описаний алгоритм однозначно задає послідовність дій для визначення множини кандидатів на вакансію. Така методика допоможе працівникам, які використовують розроблену програму, не пропустити осіб, які претендують на посаду, та осіб з неповним комплектом документів.

**3 ВИМОГИ ДО ПРОГРАМНОГО ПРОДУКТУ SELECTCREW**

**3.1 Підстава для розробки**

Дана розробка ведеться на підставі завдання до дипломного проектування: створення автоматизованої системи «Програмна система для підбору кандидатів на вакансії у крюінговій агенції».

**3.2 Функціональні вимоги до програми для підбору кандидатів на вакансії у крюінговій агенції**

Описати вимоги до програмного продукту можливо різними способами [16]:

* традиційні вимоги;
* варіанти використання;
* користувальницькі історії.

Традиційні вимоги звичайно розглядаються як можливості й обмеження системи; ключовим терміном у цьому випадку є система. Такі вимоги описують, що система може робити, а що не повинна; проте подібні вимоги, що концентрують значну частину уваги на самій системі, як правило, применшують роль взаємодії з користувачем або роль бізнесу-контексту, пов'язаного з користувачем або бізнесом. Основною складністю при наявності системи в центрі уваги є легкість інтерпретації припущень щодо щирих бажань клієнта.

Відповідно до положень Міжнародного інституту бізнесів-аналітиків (ІІBA), гарні вимоги визначаються наступними критеріями:

* вимоги повинні бути повними; вони повинні бути без незакінчених у значеннєвому плані частин або без можливості для інтерпретації;
* вимоги повинні бути такими, що тестуються; у розроблювача повинна бути можливість створити тест або будь-який інший доказ відповідності реалізації вимозі;
* вимоги повинні бути погоджені одна з одною;
* вимоги не повинні стосуватися дизайну; вони повинні вказувати можливості й обмеження системи, але не методи, за допомогою яких система забезпечує відповідність і належне виконання цих вимог;
* вимоги повинні бути однозначними; не повинно бути розпливчастих тверджень, нічого (концептуального), що може бути інтерпретоване в розріз із передбачуваним значенням.

Варіант використання написаний у вигляді ряду взаємодій між користувачем і системою, подібно запиту й відповіді, де основна увага сфальцьована на тім, як користувач буде використовувати систему. У багатьох відношеннях, варіанти використання краще, ніж традиційні вимоги, оскільки вони акцентують увагу на контекст, орієнтований на користувачів. Цінність варіанта використання для самого користувача можна вгадати, а тести, засновані на відповіді системи, можуть бути обчислені на основі взаємодій. Варіанти використання, як правило, мають два основних компоненти: діаграми варіанта використання, які графічно описують об'єкти і їхні варіанти використання, і безпосередньо сам текст варіанта використання.

Користувальницькі історії відрізняються більш оповідальним характером. Вони призначені для опису побажань користувача в плані його можливостей. Крім того, користувальницькі історії фокусують свою увагу скоріше на цінності, отриманої від використання системи, чим на деталізованій специфікації її можливостей. Метою є фіксація як можна більшої кількості інформації для того, щоб команда змогла оцінити користувальницьку історію й досить у ній розібратися для того, щоб підготувати чорновий дизайн проекту. Деталі історії колективно розробляються в процесі спілкування, тому сама користувальницька історія стає запорукою продовження спілкування протягом усього проекту. Користувальницькі історії служать інструментом, що стимулює спілкування.

Таким чином, традиційні вимоги фокусують свою увагу на варіантах експлуатації системи, схиляючись до деталізованої специфікації системи; варіанти використання фокусують свою увагу на взаємодії між користувачем і системою, також схиляючись до деталізованої специфікації; а користувальницькі історії фокусуються на цінності клієнта з убудованою невизначеністю, націленою на стимулювання спілкування.

На рис. 3.1 зображена діаграма варіантів використання (ВВ), яка описує основні функції системи і акторів, які будуть з нею взаємодіяти.

Далі наведені детальні сценарії варіантів використання.

Варіант використання Реєстрація

Основна діюча особа: Адміністратор.

Мета: Мати список користувачів, що можуть працювати з системою.

Тригер: Користувач хоче змінити список користувачів.

Основний сценарій

* 1. Система виводить список користувачів.
  2. Адміністратор хоче додати нового користувача.
  3. Система виводить форму для введення даних про користувача: логін, пароль, список можливих ролей.
  4. Адміністратор заповнює поля, обирає статус та хоче зберегти дані.
  5. Система додає нового користувача в БД та виводить оновлений список користувачів.

Альтернативний сценарій 1

4а. Адміністратор не ввів необхідні дані.

5а. Система виводить повідомлення про неможливість зберегти дані.

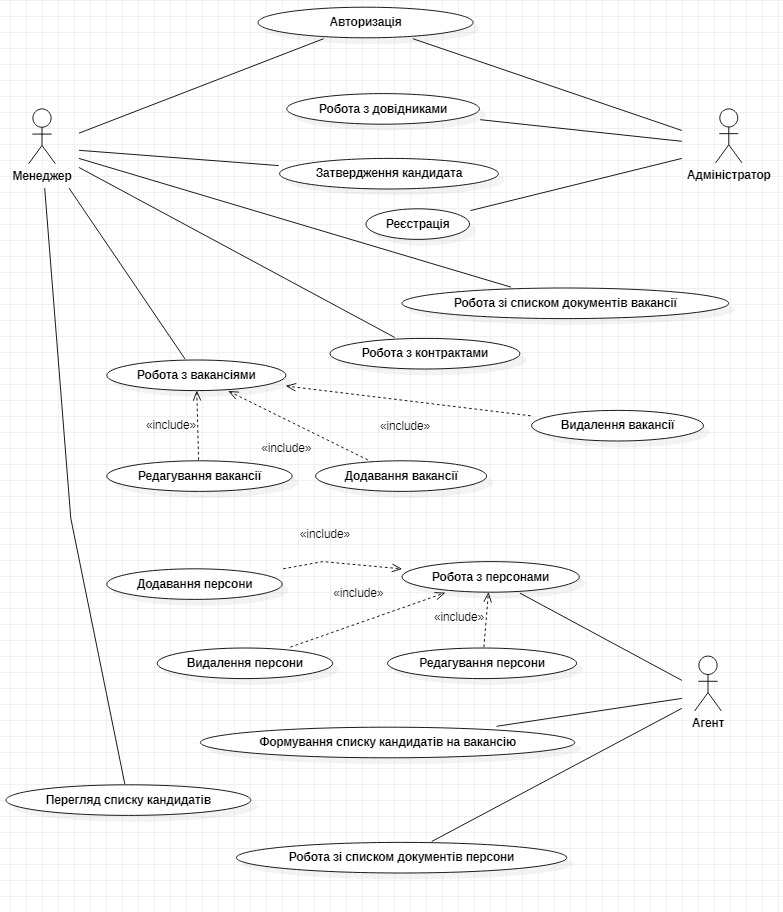


Рисунок 3.1 – Діаграма варіантів використання

Альтернативний сценарій 2

2б. Адміністратор хоче видалити користувача.

3б. Система запитує про підтвердження видалення.

4б. Адміністратор підтверджує видалення.

5б. Система помічає користувача як видаленого.

Альтернативний сценарій 3

2в. Адміністратор хоче змінити дані про користувача.

3в. Система виводить форму з даними про вказаного користувача.

4в. Адміністратор змінює дані та хоче зберегти зміни.

5в. Система вносить дані у БД.

Альтернативний сценарій 4

6г. В БД вже існує користувач з вказаним логіном. Система виводить повідомлення про існування користувача та виводить форму з введеними раніше адміністратором даними про користувача.

Варіант використання Авторизація

Основна діюча особа: Адміністратор, Менеджер, Агент.

Мета: Отримати доступ до функцій системи відповідно да виділеної користувачу ролі.

Тригер: Користувач хоче ввійти в систему.

Основний сценарій

1. Система виводить поля для введення логіну та паролю.
2. Користувач вказує логін та пароль та хоче війти в систему.
3. Система перевіряє наявність користувача з такими даними, визначає його роль та виводить форму відповідно до ролі користувача.

Альтернативний сценарій 1

3а. В БД не існує користувача з вказаним логіном. Система виводить повідомлення про відсутність користувача.

Альтернативний сценарій 2

3а. В БД існує користувач з вказаним логіном, але пароль вказаний невірно. Система виводить повідомлення про невірний пароль.

Варіант використання Робота з довідниками

Основна діюча особа: Адміністратор.

Мета: Мати множину актуальних довідкових значень для роботи з іншими даними.

Тригер: Користувач хоче змінити довідкові значення.

Основний сценарій

1. Система виводить список довідників.
2. Адміністратор обирає довідник.
3. Система виводить список значень з обраного довідника.
4. Адміністратор хоче додати нове значення.
5. Система виводить форму для введення значення.
6. Адміністратор заповнює поле та хоче зберегти дані.
7. Система додає нове значення у відповідний довідник та виводить оновлений список значень.

Альтернативний сценарій 1

6а. Адміністратор не ввів значення для довідника.

7а. Система виводить повідомлення про неможливість зберегти дані.

Альтернативний сценарій 2

2б. Адміністратор хоче видалити значення.

3б. Система запитує про підтвердження видалення.

4б. Адміністратор підтверджує видалення.

4. Система помічає значення як видалене.

Альтернативний сценарій 3

2в. Адміністратор хоче змінити значення.

3в. Система виводить форму з наявним значенням.

4в. Адміністратор змінює значення та хоче зберегти зміни.

5в. Система вносить дані у БД.

Альтернативний сценарій 4

7г. В БД вже існує введене значення. Система виводить повідомлення про існування значення та виводить форму з введеними раніше адміністратором значенням.

Варіант використання Робота з вакансіями

Основна діюча особа: Менеджер.

Мета: Мати множину актуальних вакансій.

Тригер: Користувач хоче змінити дані про вакансії.

Основний сценарій

1. Система виводить список вакансій.
2. Менеджер хоче додати відомості по нову вакансію.
3. Система виводить форму для введення даних.
4. Менеджер заповнює поля та хоче зберегти дані.
5. Система додає нову вакансію та виводить оновлений список.

Альтернативний сценарій 1

4а. Менеджер не ввів всі необхідні дані.

5а. Система виводить повідомлення про неможливість зберегти дані.

Альтернативний сценарій 2

2б. Менеджер хоче видалити вакансію.

3б. Система запитує про підтвердження видалення.

4б. Менеджер підтверджує видалення.

5б. Система помічає вакансію як видалену.

Альтернативний сценарій 3

2в. Менеджер хоче змінити дані про вакансію.

3в. Система виводить форму з наявними даними.

4в. Менеджер змінює дані та хоче зберегти зміни.

5в. Система вносить дані у БД.

Альтернативний сценарій 4

2в. Менеджер хоче помітити вакансію як зайняту.

3б. Система запитує про підтвердження дії.

4б. Менеджер підтверджує дію.

5б. Система помічає вакансію як видалену.

Варіант використання Робота з персонами

Основна діюча особа: Агент.

Мета: Мати актуальні дані про людей, що працюють в агенції, та потенційних працівників.

Тригер: Користувач хоче змінити дані про персони.

Основний сценарій

1. Система виводить список персон.
2. Агент хоче додати відомості по нову персону.
3. Система виводить форму для введення даних.
4. Агент заповнює поля та хоче зберегти дані.
5. Система додає нову вакансію та виводить оновлений список.

Альтернативний сценарій 1

4а. Агент не ввів всі необхідні дані.

5а. Система виводить повідомлення про неможливість зберегти дані.

Альтернативний сценарій 2

2б. Агент хоче видалити персону.

3б. Система запитує про підтвердження видалення.

4б. Агент підтверджує видалення.

5б. Система помічає персону як видалену.

Альтернативний сценарій 3

2в. Агент хоче змінити дані про персону.

3в. Система виводить форму з наявними даними.

4в. Агент змінює дані та хоче зберегти зміни.

5в. Система вносить дані у БД.

Варіант використання Робота зі списком документів вакансії

Основна діюча особа: Менеджер.

Мета: Мати актуальні дані про вимоги до вакансії.

Тригер: Користувач хоче змінити дані про пакет документів, необхідних персоні для зайняття вакансії.

Основний сценарій

1. Система виводить список вакансій.
2. Менеджер обирає вакансію.
3. Система виводить список документів для обраної вакансії.
4. Менеджер хоче додати відомості по новий документ.
5. Система виводить форму для введення даних.
6. Менеджер заповнює поля та хоче зберегти дані.
7. Система додає новий документ до обраної вакансії та виводить оновлений список.

Альтернативний сценарій 1

6а. Менеджер не ввів всі необхідні дані.

7а. Система виводить повідомлення про неможливість зберегти дані.

Альтернативний сценарій 2

4б. Менеджер хоче видалити документ.

5б. Система запитує про підтвердження видалення.

6б. Менеджер підтверджує видалення.

7б. Система видаляє документ.

Альтернативний сценарій 3

4в. Менеджер хоче змінити дані про документ.

5в. Система виводить форму з наявними даними.

6в. Менеджер змінює дані та хоче зберегти зміни.

7в. Система вносить дані у БД.

Варіант використання Робота зі списком документів персони

Основна діюча особа: Агент.

Мета: Мати актуальні дані про вимоги до стан документів можливого кандидата на вакансію.

Тригер: Користувач хоче змінити дані про пакет документів, наявних у персони.

Основний сценарій

1. Система виводить список персон.
2. Агент обирає персону.
3. Система виводить список документів для обраної персони.
4. Агент хоче додати відомості по новий документ.
5. Система виводить форму для введення даних.
6. Агент заповнює поля та хоче зберегти дані.
7. Система додає новий документ до обраної персони та виводить оновлений список.

Альтернативний сценарій 1

6а. Агент не ввів всі необхідні дані.

7а. Система виводить повідомлення про неможливість зберегти дані.

Альтернативний сценарій 2

4б. Агент хоче видалити документ.

5б. Система запитує про підтвердження видалення.

6б. Агент підтверджує видалення.

7б. Система видаляє документ.

Альтернативний сценарій 3

4в. Агент хоче змінити дані про документ.

5в. Система виводить форму з наявними даними.

6в. Агент змінює дані та хоче зберегти зміни.

7в. Система вносить дані у БД.

Варіант використання Перегляд списку кандидатів

Основна діюча особа: Менеджер.

Мета: Мати повну інформацію про всіх кандидатів.

Тригер: Користувач хоче отримати список кандидатів.

Основний сценарій

1. Система виводить список вакансій.
2. Менеджер обирає вакансію.
3. Система виводить список кандидатів.

Варіант використання Формування списку кандидатів на вакансію

Основна діюча особа: Агент.

Мета: Мати список персон, що можуть зайняти вакансію.

Тригер: Користувач хоче мати список людей, які задовольняють вимогам вакансії.

Основний сценарій

1. Система виводить список вакансій.
2. Агент обирає вакансію та хоче сформувати список кандидатів.
3. Система виводить список людей, наявні дані про яких дозволяють зайняти вакансію.

Альтернативний сценарій 1

3а. Система не знайшла людей, які задовольняють вимогам вакансії, та виводить повідомлення про це.

Варіант використання Затвердження кандидата

Основна діюча особа: Менеджер.

Мета: Обрати персону, яка може зайняти вакансію.

Тригер: Користувач хоче укласти контракт на зайняття вакансії .

Основний сценарій

1. Система виводить список вакансій.
2. Менеджер обирає вакансію.
3. Система виводить список кандидатів.
4. Користувач відмічає обраного для контракту кандидата.
5. Система вносить дані до БД.

Альтернативний сценарій 1

3а. Список кандидатів не сформований. Система виводить повідомлення.

**3.3 Умови експлуатації**

Система повинна працювати на ІBM-сумісному комп'ютері під керуванням ОС Lіnux або ОС Wіndows.

Температурні й інші кліматичні характеристики аналогічні тим же, що й для будь-якого персонального комп'ютера.

До складу технічних засобів повинен входити персональний комп'ютер, що містить у собі:

1) процесор із частотою-2000 MHz, не менше;

2) оперативну пам'ять обсягом не менше 1024 Mегабайт;

3) вільний простір на жорсткому диску не менше 4 Гігабайт;

4) мати доступ до бази даних і логу (журналу) запитів.

Система для підбору кандидатів на вакансії у крюінговій агенції повинна супроводжуватися технічною документацією, що включає: опис структури даних і форматів файлів, опис роботи програмних модулів і контрольний приклад взаємодії користувача із системою.

**3.4 Стадії й етапи розробки**

Розробка проводиться в три етапи:

1) аналіз вимог;

2) проектування й розробка власне програми;

3) налагодження й впровадження.

На етапі аналізу вимог розроблюється повинне бути узгоджено й затверджено технічне завдання.

На етапі проектування повинні бути виконані перераховані нижче етапи робіт:

1) спроектована структура БД;

2) розроблені екранні форми програми;

3) створена програмної документації;

4) виконане випробування програми.

На етапі налагодження й впровадження повинні бути виконані етапи підготовки й передачі програми.

**3.5 Висновки до розділу**

У розділі описані вимоги до програмної системи, описані сценарії окремих варіантів використання, розглянуті основні етапи розробки.

**4 ПРОЕКТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ SELECTCREW**

**4.1 Структура даних**

Етапи проектування структури БД наступні.

І. Концептуальне проектування [17].

Будується ER-діаграма, що включає в себе всі сутності й зв'язки. У результаті отримується концептуальна (інфологічна) модель. Варто розуміти, що така модель може не відповідати реляційної структурі проектованої бази даних.

Схема «зв'язок» (також ERD або ER-діаграма, «Сутність – Зв’язок») - це різновид блок-схеми, де показано, як різні "сутності" (люди, об'єкти, концепції й так далі) зв'язані між собою усередині системи. ER-діаграми найчастіше застосовуються для проектування й налагодження реляційних баз даних у сфері освіти, дослідження й розробки програмного забезпечення й інформаційних систем для бізнесу. ER-діаграми (або ER-моделі) покладаються на стандартний набір символів, включаючи прямокутники, ромби, овали й сполучні лінії, для відображення сутностей, їхніх атрибутів і зв'язків. Ці діаграми влаштовані по тім же принципі, що й граматичні структури: сутності виконують роль іменників, а зв'язку - дієслів [18].

Для великих баз даних побудова ER-моделі дозволяє уникнути помилок проектування, які надзвичайно складно виправляти, особливо , якщо база даних уже експлуатується або на стадії тестування. Помилки в розробці структури бази даних можуть привести до переробки коду програмного забезпечення керуючого цією базою даних. У результаті час, засоби й людські ресурси будуть використані неефективно [19].

ІІ. Логічне проектування [17].

Будується набір попередніх відношень із вказівкою первинного ключа для кожного відношення. Складається список атрибутів, потім ці атрибути розподіляються по відношенням. Необхідно, щоб всі відношення залишалися в нормальній формі Бекуса-Кодда. Перехід до реляційної структури (побудова набору відношень) виконується за наступними правилами.

1. Якщо ступінь бінарного зв'язку дорівнює 1:1 і клас приналежності обох сутностей обов'язковий, то потрібно тільки одне відношення. Первинним ключем цього відношення може бути ключ кожної із цих двох сутностей. У цьому випадку гарантується однократна поява кожного значення ключа в будь-якому екземплярі відношення.
2. Якщо ступінь бінарного зв'язку дорівнює 1:1 і клас однієї із сутностей необов'язковий, то необхідно побудову двох відношень, під кожну сутність необхідне виділення одного відношення. Ключ сутності, для якого клас приналежності є необов'язковим, додається як атрибут у відношення, виділене для сутності з обов'язковим класом приналежності.
3. Якщо ступінь бінарного зв'язку дорівнює 1: М и клас приналежності М-зв'язної сутності обов'язковий, то досить використати два відношення: по одному на кожну сутність, за умови, що ключ сутності служить як первинний ключ для відповідного відношення. Ключ же однозв'язної сутності повинен бути доданий як атрибут у відношення, що відповідає М-зв'язної сутності.
4. Якщо ступінь бінарного зв'язку дорівнює М: М, то для зберігання даних необхідно три відношення: по одному на сутність і одне для зв'язку. Ключі сутності входять у зв'язок.
5. У випадку тристороннього зв'язку необхідно використати чотири відношення: по одному на сутність і одне для зв'язку. Відношення, породжуване зв'язком, має в собі серед атрибутів ключі сутності від кожної сутності.

Області застосування діаграм «сутність-зв'язок» [18]:

* проектування баз даних - ER-діаграми застосовуються для моделювання й проектування реляційних баз даних, причому як у плані логічних і бізнес-правил (логічні моделі даних), так і в плані впровадження конкретних технологій (фізичні моделі даних); у сфері розробки програмного забезпечення ER-діаграма, як правило, служить першим кроком у визначенні вимог проекту по створенню інформаційних систем; на подальших етапах роботи ER-діаграми також застосовуються для моделювання конкретних баз даних;
* налагодження баз даних - ER-діаграми застосовуються для аналізу вже наявних баз даних з метою виявити й усунути помилки в логіку або розгортанні; діаграма дозволяє виявити, де саме виникли помилки;
* інформаційні системи для бізнесу - ER-схеми використаються для проектування й аналізу реляційних баз даних, застосовуваних у бізнес-процесах; реляційніх бази даних можуть використатися в будь-якому бізнес-процесі, де задіяні дані, розбиті на поля, включаючи сутності, дії й взаємозв'язки; бази даних допомагають оптимізувати процеси, витягати дані й підвищувати якість результатів;
* реорганізація бізнесів-процесів (BPR) - ER-діаграми допомагають аналізувати бази даних, застосовувані при реорганізації бізнес-процесів і моделюванні нових баз даних;
* освіта - бази даних є широко розповсюдженим способом зберігання реляційної інформації, застосовуваної з метою освіти й для наступного витягу даних, тому ER-діаграми грають важливу роль у плануванні подібних структур даних;
* дослідницька діяльність - оскільки дослідницька робота багато в чому опирається на чітко структуровані дані, ER-діаграми відіграють ключову роль у побудові оптимальних баз даних для аналізу інформації.

Перший принцип розробки структури БД полягає в тому, що потрібно видалити повторювані дані (які також називаються надлишковими даними), оскільки вони займають зайве місце та збільшують імовірність виникнення помилок і невідповідностей. Другий принцип полягає в тому, що важливою є правильність і повнота інформації. Якщо БД містить неправильні відомості, усі звіти, які отримують інформацію з неї, також відображатимуть неправильні дані. І люди, які виносять рішення на підставі цих звітів, будуть дезінформовані [20]. Отже, правильна структура бази даних:

* розподіляє ваші дані на тематичні таблиці, щоб зменшити надлишкові дані;
* надає СКБД інформацію, потрібну для з’єднання відомостей у таблицях за необхідності;
* допомагає підтримувати та забезпечувати точність і цілісність даних;
* відповідає потребам користувача щодо обробки даних і звітування.

На основі аналізу предметної області та її моделі визначені наступні сутності:

* Персона;
* Користувач;
* Вакансія;
* Судно;
* Посада;
* Досвід роботи;
* Контракт;
* Рекомендація;
* Документ.

Визначені такі види взаємозв’язків між означеними сутностями.

Персона може працювати на протязі різних періодів на різних посадах, на одній посаді можуть працювати різні персони, таким чином, між сутностями Персона та Посада існує взаємозв’язок «багато-до-багатьох».

Персона може мати пакет різних типів документів, один тип документу можуть мати різні персони, таким чином, між сутностями Персона та Документ існує взаємозв’язок «багато-до-багатьох».

При цьому у кожної персони визначений свій власний термін дії окремого типу документу.

Посада може потребувати пакет різних типів документів, один тип документу може використовуватися для різних посад, таким чином, між сутностями Посада та Документ існує взаємозв’язок «багато-до-багатьох».

Контракт з персоною укладається для роботи на одній посаді, на одному судні, таким чином, існують такі взаємозв’язки типу «один-до-багатьох»: Персона та Контракт, Посада та Контракт, Судно та Контракт.

Вакансія визначається для роботи на одній посаді, на одному судні, таким чином, існують такі взаємозв’язки типу «один-до-багатьох»: Судно та Вакансія, Посада та Вакансія.

Персона може мати досвід роботи, який надбаний у інших крюінгових агенціях. Кожний факт досвіду пов'язаний з однією персоною, однією посадою, одним типом судна. Але багато людей мають досвід на одній посаді, на одному типу судна, тому існують такі взаємозв’язки типу «один-до-багатьох»: Посада – Досвід, Тип судна – Досвід, Персона – Досвід.

Також кожен користувач має єдину можливу роль у системі, але одну роль можуть мати кілька користувачів. Кожна посада пов’язана с одним типом судна. але з одним типом пов’язано багато посад. Кожне судно має один тип, але існує багато суден одного типу. Кожен вид документу має один тип з точку зору швидкості отримання, але один тип мають багато документів. Перелічені дані –тип судна, тип документу, роль користувача – відносяться до таких, що мають невелику кількість значень, і їх значення змінюються відносно нечасто. Дані з такими характеристиками можуть бути віднесені до довідкових и зберігатися в таблицях-довідниках. Можливий варіант збереження всіх довідкових даних в одній таблиці-довіднику з визначенням типу довідника у якості атрибуту такої таблиці.

Тобто визначені ще такі взаємозв’язки типу «один-до-багатьох»: Довідник – Користувач, Довідник – Судно, Довідник – Посада, Довідник – Документ.

Для предметної області визначена модель «Сутність-Зв'язок» наступного виду (рис. 4.1).

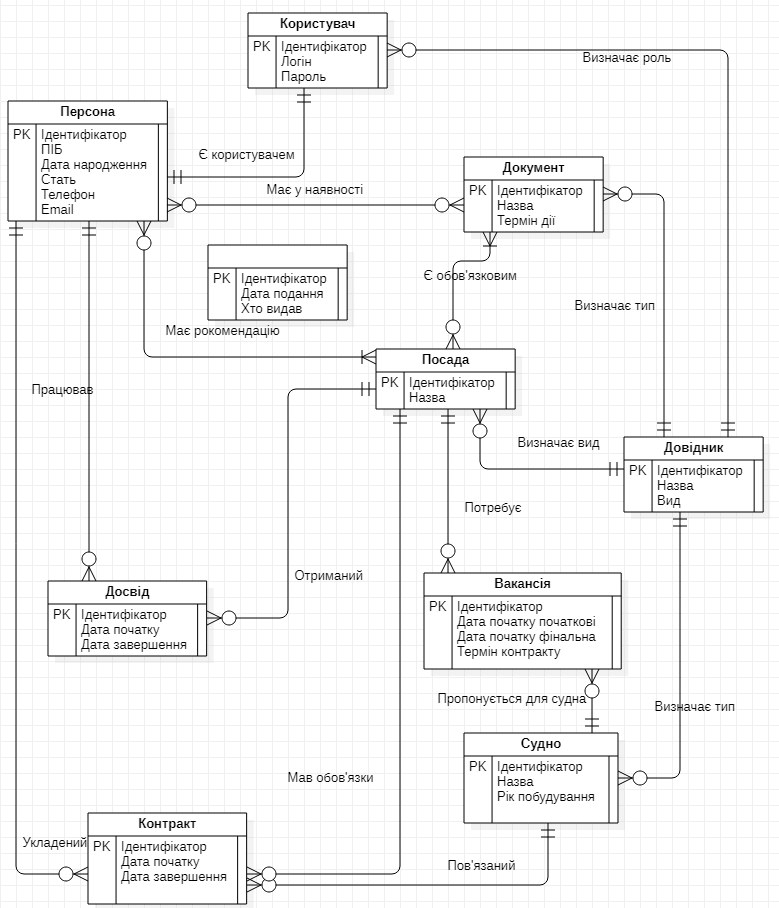


Рисунок 4.1 – Модель «Сутність – Зв’язок»

На основі представленої моделі розроблена реляційна БД, яка містить множину взаємопов’язаних таблиць.

Таблиця 4.1 наводить детальний опис структури таблиці для збереження даних про екземпляри сутності Контракт.

Таблиця 4.1 – Структура реляційної таблиці Contract

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип даних | Опис | Додаткові дані |
| idc | int (7) | Ідентифікатор контракту | Первинний ключ |
| number | varchar (50) | Номер контракту |  |
| datestart | date | Дата початку |  |
| dateend | date | Дата завершення |  |
| idd | int (3) | Ідентифікатор посади | Зовнішній ключ |
| ids | int (4) | Ідентифікатор судна | Зовнішній ключ |
| idp | int (4) | Ідентифікатор персони | Зовнішній ключ |

Таблиця 4.2 наводить детальний опис структури таблиці для збереження даних про екземпляри сутності Документ.

Таблиця 4.2 – Структура реляційної таблиці Document

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип даних | Опис | Додаткові дані |
| idc | int (7) | Ідентифікатор контракту | Первинний ключ |
| named | varchar (50) | Назва документу |  |

Продовження табл. 4.2

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип даних | Опис | Додаткові дані |
| period | int (3) | Термін дії у місяцях |  |
| madeperiod | int (3) | Мінімальний термін отримання діючого документу |  |

Таблиця 4.3 наводить детальний опис структури таблиці для збереження даних про екземпляри сутності Досвід.

Таблиця 4.3 – Структура реляційної таблиці Experience

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип даних | Опис | Додаткові дані |
| ide | int (7) | Ідентифікатор факту досвіду | Первинний ключ |
| idd | int (3) | Ідентифікатор посади | Зовнішній ключ |
| datestart | date | Дата початку |  |
| dateend | date | Дата завершення |  |
| ids | int (4) | Ідентифікатор судна | Зовнішній ключ |
| idc | int (7) | Ідентифікатор контракту | Зовнішній ключ |
| idp | int (4) | Ідентифікатор персони | Зовнішній ключ |

Таблиця 4.4 наводить детальний опис структури таблиці для збереження даних про екземпляри сутності Персона.

Таблиця 4.4 – Структура реляційної таблиці Person

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип даних | Опис | Додаткові дані |
| idp | int (7) | Ідентифікатор факту досвіду | Первинний ключ |
| fio | varchar (150) | ПІБ |  |
| fioen | varchar (150) | ПІБ на англійській мові |  |
| citizen | int (3) | Громадянство | Зовнішній ключ |
| bday | date | Дата народження |  |
| gender | varchar (10) | Стать |  |
| email | varchar (30) | Почтова адреса |  |
| phone | varchar (30) | Телефон |  |

Таблиця 4.5 наводить детальний опис структури таблиці для збереження даних про взаємозв’язок між сутностями Персона та Документ.

Таблиця 4.5 – Структура реляційної таблиці PersonDocument

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип даних | Опис | Додаткові дані |
| idpd | int (7) | Ідентифікатор факту наявності документу | Первинний ключ |
| dateend | varchar (150) | Дата закінчення дії документу |  |

Продовження табл. 4.5

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип даних | Опис | Додаткові дані |
| idd | varchar (150) | Ідентифікатор документу | Зовнішній ключ |
| idp | int (3) | Ідентифікатор персони | Зовнішній ключ |

Таблиця 4.6 наводить детальний опис структури таблиці для збереження даних про сутність Посада.

Таблиця 4.6 – Структура реляційної таблиці Position

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип даних | Опис | Додаткові дані |
| idp | int (7) | Ідентифікатор посади | Первинний ключ |
| namep | varchar (150) | Назва посади |  |
| idt | int (10) | Тип посади | Зовнішній ключ |

Таблиця 4.7 наводить детальний опис структури таблиці для збереження даних про сутність Рекомендація.

Таблиця 4.7 – Структура реляційної таблиці Promotion

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип даних | Опис | Додаткові дані |
| idprm | int (7) | Ідентифікатор рекомендації | Первинний ключ |
| description | varchar (150) | Опис рекомендації |  |
| datep | int (10) | Дата видачі |  |

Продовження табл. 4.7

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип даних | Опис | Додаткові дані |
| idp | int (7) | Ідентифікатор персони | Зовнішній ключ |
| idd | int (4) | Ідентифікатор посади | Зовнішній ключ |

Таблиця 4.8 наводить детальний опис структури таблиці для збереження даних про сутність Судно.

Таблиця 4.8 – Структура реляційної таблиці Ship

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип даних | Опис | Додаткові дані |
| ids | int (7) | Ідентифікатор судна | Первинний ключ |
| number | varchar (50) | Міжнародний номер |  |
| names | varchar (50) | Назва |  |
| yearbuild | int (4) | Рік побудови |  |
| country | varchar (50) | Під прапором якої країни |  |
| idtype | int (10) | Ідентифікатор типу | Зовнішній ключ |

Таблиця 4.9 наводить детальний опис структури таблиці для збереження даних про довідникові значення.

Таблиця 4.9 – Структура реляційної таблиці spr

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип даних | Опис | Додаткові дані |
| ids | int (7) | Ідентифікатор значення | Первинний ключ |
| names | varchar (50) | Значення |  |
| kinds | int (2) | Тип довідника |  |

Таблиця 4.10 наводить детальний опис структури таблиці для збереження даних про довідникові значення.

Таблиця 4.10 – Структура реляційної таблиці user

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип даних | Опис | Додаткові дані |
| idu | int (7) | Ідентифікатор користувача | Первинний ключ |
| login | varchar (50) | Логін |  |
| password | int (2) | Пароль |  |

Таблиця 4.11 наводить детальний опис структури таблиці для збереження даних про сутність Вакансія.

Таблиця 4.11 – Структура реляційної таблиці Vacancy

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип даних | Опис | Додаткові дані |
| idv | int (7) | Ідентифікатор вакансії | Первинний ключ |
| idd | varchar (50) | Ідентифікатор посади | Зовнішній ключ |
| datestart | int (2) | Дата початку |  |
| period | int (2) | Тривалість контракту в місяцях |  |
| comment | varchar (250) | Коментар |  |
| ids | int (4) | Ідентифікатор судна | Зовнішній ключ |

Таблиця 4.12 наводить детальний опис структури таблиці для збереження даних про взаємозв’язок між сутностями Посада та Документ.

Таблиця 4.12 – Структура реляційної таблиці Vacancy

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип даних | Опис | Додаткові дані |
| idpd | int (7) | Ідентифікатор вакансії | Первинний ключ |
| idp | int (4) | Ідентифікатор посади | Зовнішній ключ |
| idd | int (4) | Ідентифікатор документу | Зовнішній ключ |

**4.2 Архітектура програмної системи**

Існує багато визначень поняття архітектури програмного забезпечення. Нижче наведені деякі з них.

Архітектура - це набір значимих рішень із приводу організації системи програмного забезпечення, набір структурних елементів і їхніх інтерфейсів, за допомогою яких компонується система, разом з їхнім поводженням, обумовленим у взаємодії між цими елементами, компонування елементів у поступово, що укрупнюються підсистеми, а також стиль архітектури, який направляє цю організацію - елементи і їхні інтерфейси, взаємодії й компонування. [21]

Архітектура програми або комп'ютерної системи - це структура або структури системи, які включають елементи програми, видимі ззовні властивості цих елементів і зв'язки між ними [22].

Архітектура програмного забезпечення системи або набору систем складається із всіх важливих проектних рішень із приводу структур програми й взаємодій між цими структурами, які складають системи. Проектні рішення забезпечують бажаний набір властивостей, які повинна підтримувати система, щоб бути успішної. Проектні рішення надають концептуальну основу для розробки системи, її підтримки й обслуговування [23].

Шаблони архітектури являють собою готові форми для рішення стандартних архітектурних завдань. Середовище архітектури або інфраструктура архітектури (проміжне програмне забезпечення) - це набір компонентів, на базі яких можна побудувати певну архітектуру. Середовище (інфраструктура) повинна містити компоненти для рішення основних завдань архітектури, звичайно в межах певної предметної області, наприклад керування [24].

В [25] шаблони архітектури згруповані по характеристиках систем, у яких вони найбільше часто застосовуються, при цьому одна категорія відведена під шаблони загальної структури (табл.4.1).

Усі веб-сайти працюють за однаковим принципом [26]:

* одержання й обробка запиту від користувача (GET-запит на вебсторінку)
* розуміння того, як на цей запит потрібно відреагувати (одержати інформацію з бази даних і повернути її користувачеві)
* робота з даними, їхнє одержання/зміна в базі даних (одержання інформацію з бази даних)
* формування представлення для користувача (заповнення HTML-шаблона даними з бази даних)
* відправлення відповіді користувачеві (відправлення сформованої HTML-сторінки з таблицею).

Таблиця 4.1 – Приклади шаблонів архітектури ПЗ

|  |  |
| --- | --- |
| Категорія | Шаблон |
| Структура | Рівні |
| Класна дошка |
| Класна дошка |
| Розподілені системи | Посередники |
| Інтерактивні системи | Модель-представлення-контролер |
| Представлення-абстракція-керування |
| Адаптивні системи | Відбиття |
| Мікроядра |

Саме для такого сценарію використовується готовий паттерн проектування, який називається MVC (Модель-представлення-контролер).

Модель - частина застосування, що працює з даними. Вона містить у собі дані й уміє їх відображати в базу даних. Тобто вона може додавати записи в базу даних, видаляти їх, змінювати, або ж просто одержувати їх звідти.

Vіew (Представлення) - частина програми, що відповідає за те, у якому виді користувач одержить дані від застосування. У цей блок приходять деякі дані від контролера, наприклад, ім'я HTML-шаблона й змінні, які в цей шаблон потрібно передати.

Controller (Контролер) - сполучна ланка між запитом від користувача, моделями й представленням. Саме контролер є крапкою входу в застосування. Сюди приходить запит від користувача й приймається рішення про те, що із цим запитом робити [26].

Розроблювана програма SELECTСREW відноситься до інтерактивних систем та працює за допомогою браузера, тому обраний шаблон Модель-представлення-контролер (.

На рис. 4.2 представлення схема реалізації архітектури програми SELECTСREW за шаблоном MVC.

Програма реалізована у вигляді набору модулів, які згруповані у папки.

Папка Classes включає модулі, кожен з яких містить опис класу для відповідного об’єкту предметної області.

Існує клас (модуль DB) для підключення до БД, через метод якого виконується підключення для будь-якого об’єкту. Також існує клас (модуль Total), у методах якого зібрані та реалізовані загальні операції для усіх модулів.

Інші модулі реалізують специфічні для кожного об’єкта функції, найчастіше у вигляді запитів до БД.

Папка View включає модулі, кожен з яких містить код, який формує вигляд сторінок для відображення відповідної інформації. Модулі містять код для формування таблиць для об’єктів предметної області, а також для таблиць, що збирають дані з різних таблиць БД для реалізації зв’язку «багато до багатьох».

Папка Edit включає модулі, кожен з яких містить код, який ж контролером для виклику відповідних методів з класів частини Модель. Модулі згруповані по об’єктам, для кожного об’єкта існує модуль для реалізації видалення екземпляру об’єкта (модуль має назву, яка включає назву обєкта та слово delete), отримання даних для реалізації функції редагування (модуль має назву, яка включає назву об’єкта та слово edit), та для збереження даних, нових або відредагованих(модуль має назву, яка включає назву об’єкта та слово save).

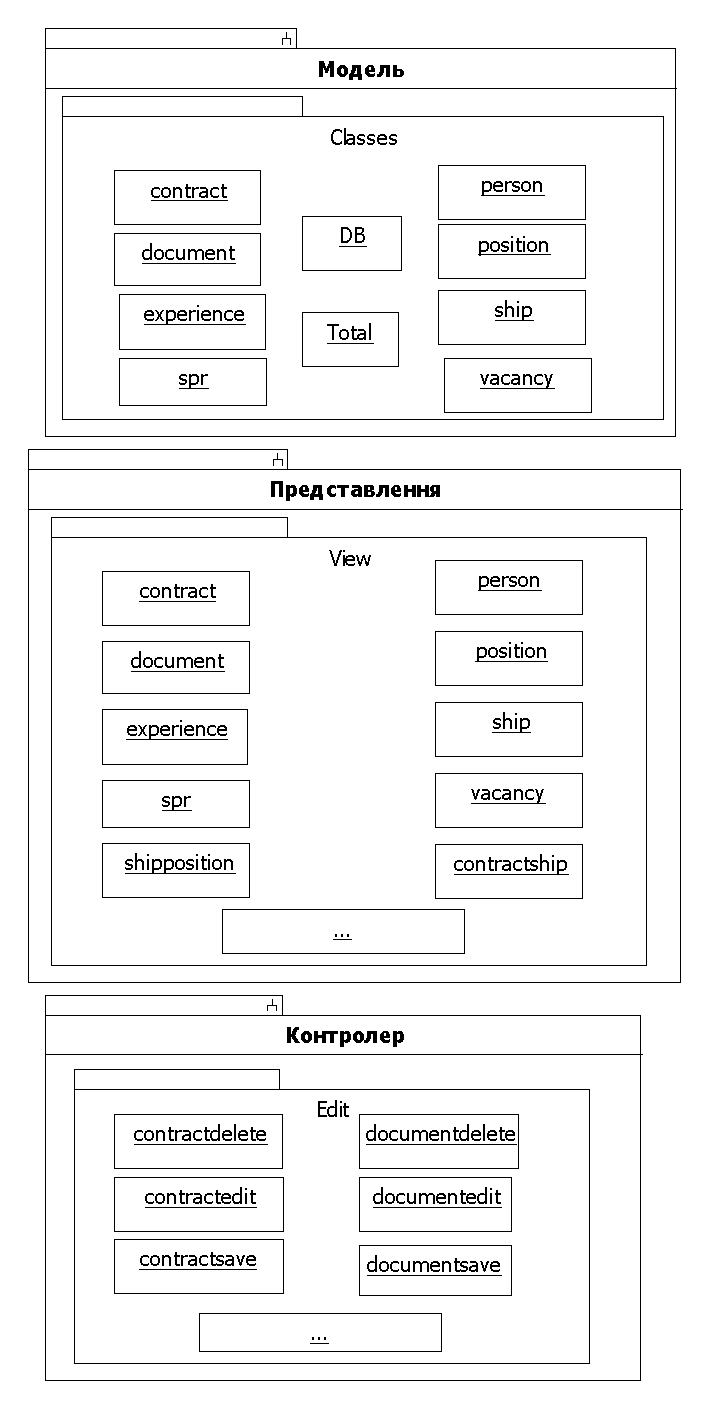


Рисунок 4.2 – Архітектура програмного забезпечення

**4.3 Висновки до розділу**

У розділі на основі формалізованого представлення предметної області розроблена структура БД, яка містить набор нормалізованих таблиць для збереження інформації про об’єкти предметної області. Описана архітектура програмного забезпечення, обґрунтований вибір шаблону модель-представлення-контролер.

**5 РЕАЛІЗАЦІЯ ПРОГРАМИ ДЛЯ КРЮІНГОВОЇ АГЕНЦІЇ**

**5.1 Класи програми**

Для реалізації функцій програми створені класи, діаграма яких відображена на рис. 5.1.

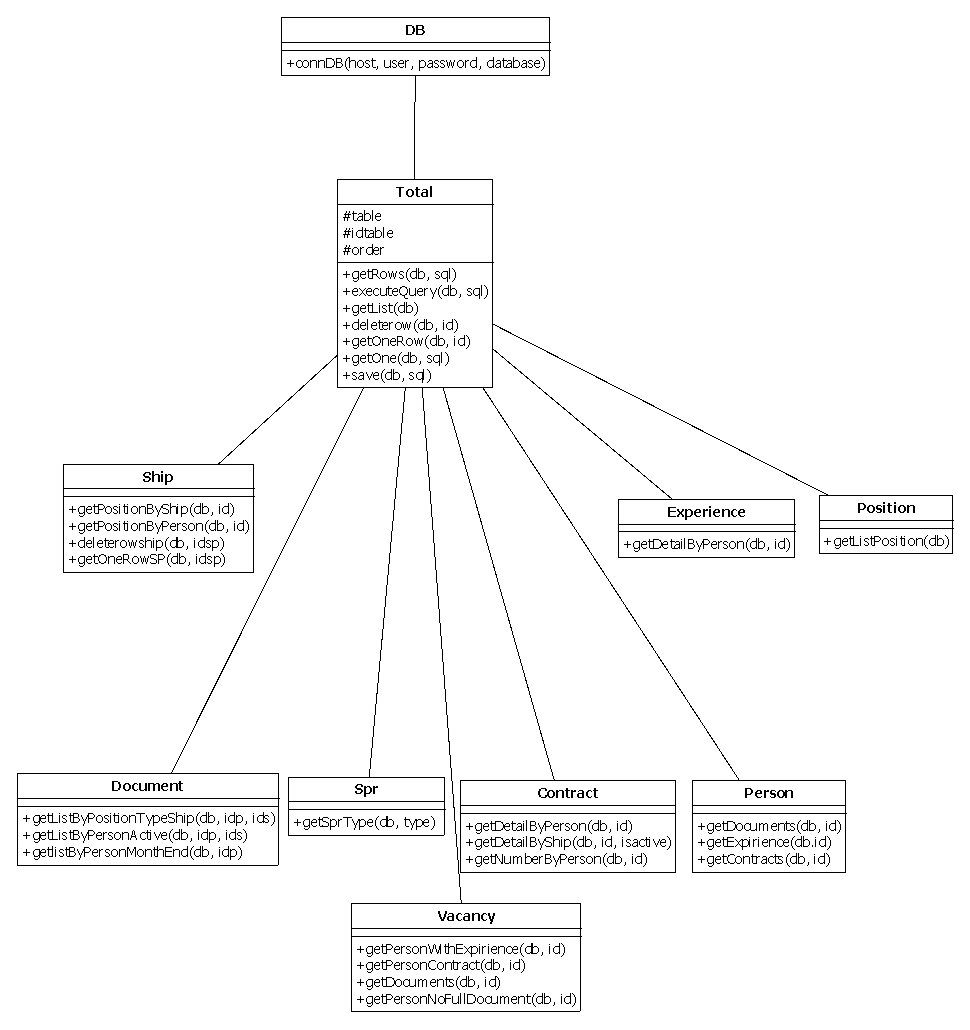


Рисунок 5.1 Діаграма програмних класів

Нижче наведений опис окремих класів.

Клас **DB** призначений для встановлення з’єднання з БД.

Методи класу

connDB() – встановлює з’єднання з БД

Аргументи методу

host – хост, на якому знаходиться БД;

user – ім’я користувача, який має доступ до БД;

password – пароль для доступу до БД;

database – ім’я БД.

Метод повертає об’єкт, що визначає з’єднання з БД.

Клас **Total** призначений для виконання загальних дій з таблицями БД.

Змінні класу

table – ім’я таблиці, для якої виконуються загальні дії;

idtable – первинний ключ таблиці для доступу к конкретному рядку таблиці;

order – порядок, за яким за замовчуванням виконується виведення рядків таблиці.

Методи класу

getRows() – отримує таблицю, сформовану на основі запиту до БД

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

sql – текст запиту.

Метод повертає об’єкт, що визначає набор рядків результуючої таблиці запиту.

executeQuery() – виконує запит до БД на зміну даних

db – об’єкт, що визначає з’єднання з БД;

sql – текст запиту.

Метод повертає повідомлення про помилку, якщо запит не може бути виконаний.

getList(db) – отримує усю таблицю

db – об’єкт, що визначає з’єднання з БД.

Метод повертає об’єкт, що визначає набор рядків таблиці, вказаної у змінній table.

deleterow() – видаляє рядок із заданим ідентифікатором

db – об’єкт, що визначає з’єднання з БД;

id – значення ідентифікатору рядка (ім’я визначається змінною idtable) таблиці, визначеної у змінній table.

Метод повертає кількість видалених рядків.

getOneRow() – визначає єдиний рядок таблиці, пов'язаний з заданим значення ідентифікатора.

db – об’єкт, що визначає з’єднання з БД;

id – значення ідентифікатору рядка (ім’я визначається змінною idtable) таблиці, визначеної у змінній table.

Метод повертає набор значень з обраного рядка таблиці

getOne() – визначає єдине значення, сформоване запитом

db – об’єкт, що визначає з’єднання з БД;

sql – текст запиту.

Метод повертає значення , отримане у результаті виконання запиту.

save() – зберігає дані з використанням запиту модифікації даних

db – об’єкт, що визначає з’єднання з БД;

sql – текст запиту.

Метод повертає кількість рядків, яких стосувалось оновлення.

Клас **Ship** призначений для отримання даних, які стосуються судів.

Методи класу

getPositionByShip() – отримує список посад, що є на судні, з вказівкою дати закінчення контракту, що складений для відповідної посади

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

id – значення ідентифікатору судна.

Метод повертає набор посад.

getPositionByPerson() – отримує список поточних контрактів для судна

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

id – значення ідентифікатору судна.

Метод повертає набор контрактів.

deleterowship() – видаляє посаду для судна

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

idsp – значення ідентифікатору посади на судні.

getOneRowSP() – визначає ідентифікатор посади для заданої позиції екіпажу судна

db – об’єкт, що визначає з’єднання з БД;

idsp – значення ідентифікатору посади на судні.

Метод повертає ідентифікатор посади.

Клас **Document** призначений для отримання даних, які стосуються документів.

Методи класу

getListByPositionTypeShip() – отримує список документів, які потрібно мати моряку для зайняття заданої посади на заданому типі судна

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

idp – значення ідентифікатору посади

ids – значення ідентифікатору судна, за яким визначається тип судна.

Метод повертає набор документів.

getListByPersonActive() – отримує список персон, які мають повний комплект діючих документів для зайняття зазначеної посади на зазначеному типі судна

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

idp – значення ідентифікатору посади;

ids – значення ідентифікатору судна, за яким визначається тип судна.

Метод повертає набор персон.

getlistByPersonMonthEnd() – отримує список персон, у яких закінчується термін дії документів

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

idp – значення ідентифікатору персони.

Метод повертає набор персон.

Клас **Spr** призначений для отримання даних, які стосуються довідникових даних.

Методи класу

getSprType() – отримує список значень з довіднику, які відповідають зазначеному типу

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

type – значення типу довідникових даних.

Метод повертає набор довідникових значень.

Клас **Vacancy** призначений для отримання даних, які стосуються вакансій.

Методи класу

getPersonWithExpirience() – отримує список персон, що мають відповідний досвід

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

id – значення ідентифікатору вакансії.

Метод повертає набор персон.

getPersonContract() – отримує список персон, що мали контракт на відповідній посаді

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

id – значення ідентифікатору вакансії.

Метод повертає набор персон.

getDocuments() – отримує список документів, які необхідні для зайняття вакансії

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

id – значення ідентифікатору вакансії.

Метод повертає набор документів.

getPersonNoFullDocument() – отримує список персон, які мають неповний пакет документів, але мають змогу підготувати документи у термін, що залишився до відкриття вакансії

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

id – значення ідентифікатору вакансії.

Метод повертає набор персон.

Клас **Contract** призначений для отримання даних, які стосуються контрактів.

Методи класу

getDetailByPerson() – отримує детальні дані про контракти зазначеної персони

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

id – значення ідентифікатору персони.

Метод повертає набор контрактів.

getDetailByShip() – отримує детальні дані про контракти зазначеного судна

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

id – значення ідентифікатору судна.

isactive – визначає, діючи або архівні контракти повертати, якщо isactive=1, то повертати діючи контракти, якщо isactive=0, то повертати архівні контракти, тобто такі, що вже закінчилися

Метод повертає набор контрактів.

getNumberByPerson()– отримує номера контрактів для зазначеної персони

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

id – значення ідентифікатору персони.

Метод повертає набор номерів контрактів.

Клас **Person** призначений для отримання даних, які стосуються персон.

Методи класу

getDocuments () – отримує детальні дані про документи зазначеної персони

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

id – значення ідентифікатору персони.

Метод повертає набор документів.

getExpirience () – отримує детальні дані про досвід зазначеної персони

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

id – значення ідентифікатору персони.

Метод повертає набор фактів роботи персони в інших компаніях.

getContracts() – отримує детальні дані про контракти зазначеної персони

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

id – значення ідентифікатору персони.

Метод повертає набор контрактів.

Клас **Experience** призначений для отримання даних, які стосуються досвіду роботи персон у інших компаніях.

Методи класу

getDetailByPerson () – отримує детальні дані про досвід зазначеної персони

Аргументи методу

db – об’єкт, що визначає з’єднання з БД;

id – значення ідентифікатору персони.

Метод повертає набор фактів роботи персони в інших компаніях.

Клас **Position** призначений для отримання даних, які стосуються посад.

Методи класу

getListPosition() – отримує детальні дані про посади з приєднанням даних з довіднику

Аргументи методу

db – об’єкт, що визначає з’єднання з БД.

Метод повертає набор посад.

**5.2 Робота користувача з програмою**

Після авторизації користувач попадає у вікно, яке містить меню у правій частині та детальні дані у лівій частині (рис.5.2).

Активний пункт меню має темний колір.



Рисунок 5.2 – Структура вікна програми

Для модифікації даних, які в поточний момент часу знаходяться на екрані (у відповідності до обраного пункту меню), призначений набор кнопок: для додання, для редагування та для видалення.

При натисканні на кнопку Новий відкриється форма для внесення відповідних даних (рис. 5.3).

Необхідно заповнити виведені на екрані поля та натиснути кнопку Зберегти. Якщо дані заповнені коректно, то вікно буде закрите, а додані дані відображені у таблиці у основному вікні.

Для відмови додавання нових даних потрібно натиснути кнопку Скасувати. дані збережені не будуть.

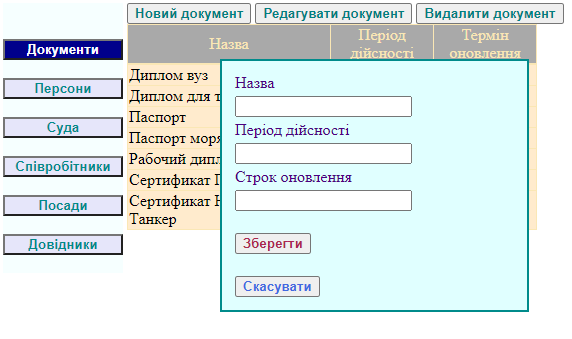


Рисунок 5.3 – Додавання даних

Для редагування даних потрібно натиснути обрати рядок у таблиці. Для цього необхідно кликнути мишею по рядку. рядок змінить колір. Потім необхідно натиснути кнопку Редагувати. Відкриється модальне вікно, аналогічне вікну для додавання даних, але з заповненими полями (рис. 5.4).

Для полів, де можливий вибір, організовані списки, користувач обирає потрібне значення. На прикладі рис. 5.4 можна обрати стать персони (відкритий список з малою кількість рядків) та громадянство (випадаючий список, сформований на основі довідникових даних).

Користувач має змогу змінити виведені дані, а потім для їх збереження необхідно натиснути кнопку Зберегти. В БД буде виконано оновлення, в таблиці в основній формі будуть відображені зроблені зміни.

Для відмови від редагування потрібно натиснути кнопку Скасувати. Змінені дані не будуть збережені, вікно для редагування закриється.

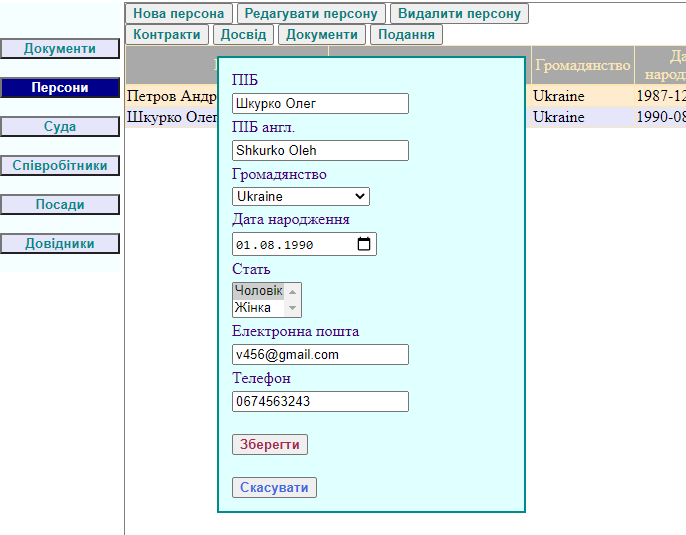


Рисунок 5.4 – Редагування даних

Для видалення даних необхідно обрати рядок та натиснути кнопку Видалити. Після підтвердження дані будуть видалені.

Для персон існують додаткові кнопки для перегляду та редагування їх контрактів, досвіду, документів та представлень на підвищення посади (рис. 5.5).

При натисканні на додаткову кнопку буде виведений список відповідних даних для обраної персони (рис. 5.6). Рядок, що відповідає обраній персоні має інший колір.



Рисунок 5.5 – Додаткові кнопки для персон

У нижній частині екрану виводиться список контрактів, фактів досвіду, документів або представлення на підвищення посади у відповідності до натиснутої кнопки. також тут містяться кнопки для додавання, редагування та видалення відповідної позиції.

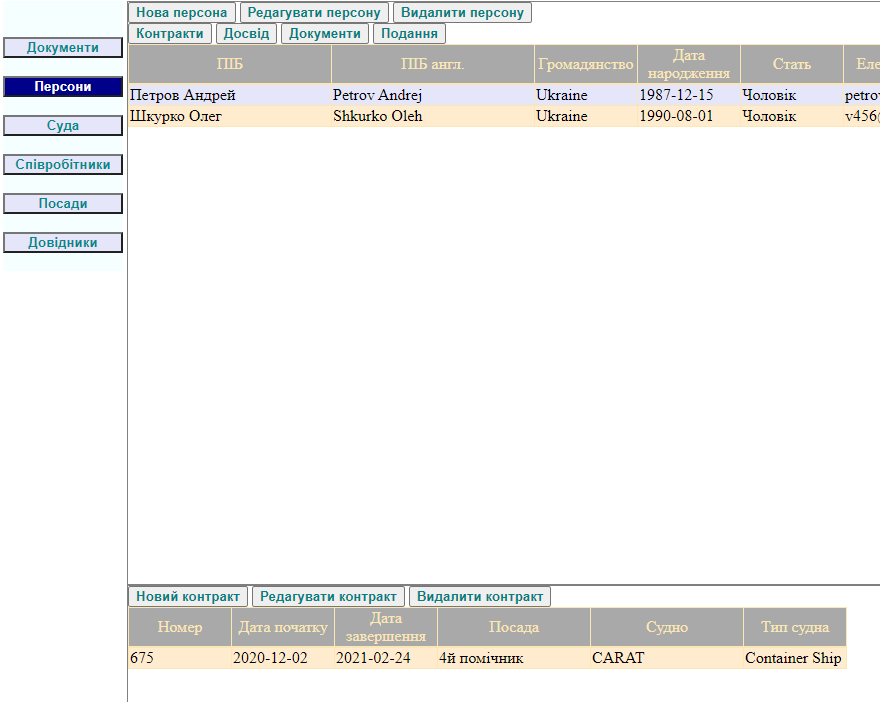


Рисунок 5.6 – Виведення додаткових даних

На рис.5.7 показано вікно для редагування контракту. Вікно містить поля зі списком, що випадає. У список включаються дані, які отримані на основі запитів з БД.

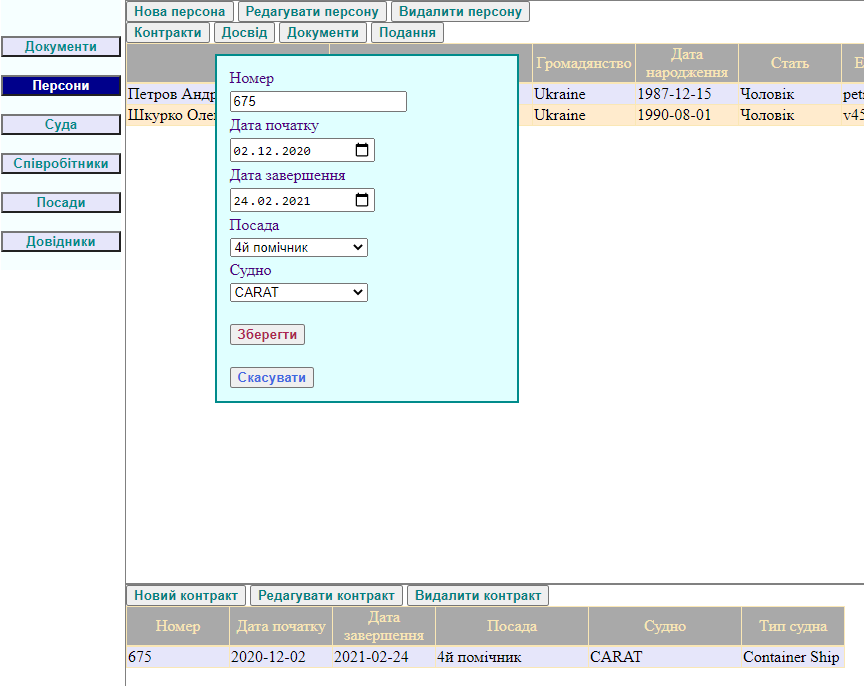


Рисунок 5.7 – Редагування підлеглих даних

Для судів також можливі додаткові дані, наприклад контракти та посади. Під час перегляду контрактів можна поділити їх на діючі та архівні (рис. 5.8). Для цього потрібно натиснути відповідну кнопку.

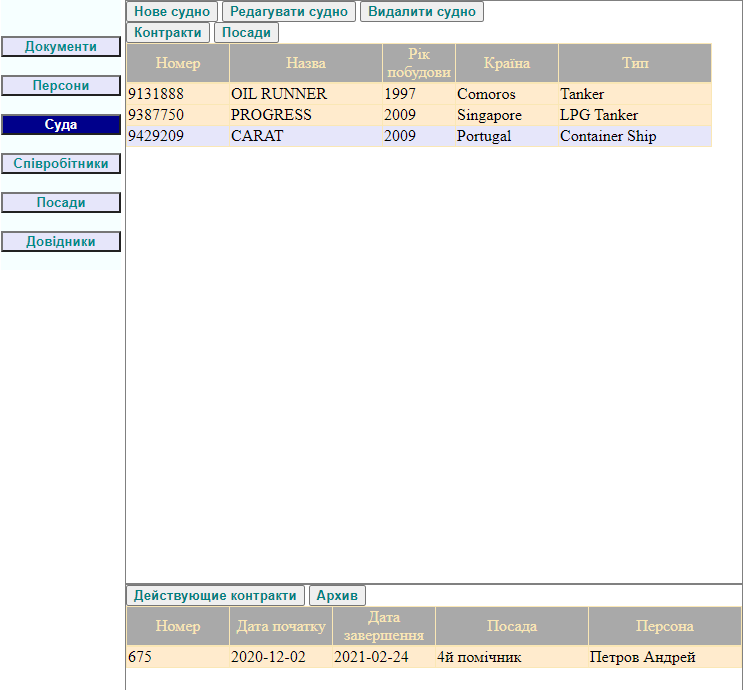


Рисунок 5.8 – Перегляд діючих та архівних контрактів

**5.3 Висновки до розділу**

У розділі надана структура класів, які використаються для реалізації функцій програми, детально описані методи класів. Показано, як програма взаємодіє з користувачем, за допомогою яких засобів можна переглядати та змінювати дані.

**6 ТЕСТУВАННЯ ПРОГРАМНОГО ПРОДУКТУ**

Перевірку програмного продукту проводиться за допомогою поділяють функціональних та нефункціональних тестів.

Функціональне тестування перевіряє відповідність бізнес-вимогам програмного продукту. Основне призначення функціонального тестування — переконатися, що весь функціонал ПЗ працює саме так, як зазначено в технічному завданні [27]. Тобто під час проведення функціонального тестування, важливий результат роботи системи в цілому.

Нефункціональне тестування передбачає перевірку зручності, продуктивності, надійності тощо.

Типи функціональних тестів [27]:

* юніт (модульне) тестування - допомагає переконатися, що окремі компоненти програми виконують свої функції на рівні коду та мають очікувану поведінку, можуть проводитися вручну або автоматизовано;
* інтеграційне тестування призначено для перевірки, чи працюють обєднані незалежні компоненти програмибез помилок;
* системне тестування проводиться методом «чорної скриньки», та перевіряє, чи відповідає програма заданим вимогам;
* приймальне тестування виконується перед випуском ПЗ, для того, щоб переконатись, чи готовий продукт для реальної роботи і виконується така перевірка кінцевими споживачами.

Типи нефункціональних тестів [27]:

* тестування надійності перевіряє, чи може програма підтримувати заданий рівень продуктивності за заданих умов в заданий період часу;
* тестування міцності перевіряє, чи працює система правильно за неочікуваних умов;
* стрес-тестування перевіряє поведінку системи за нетипових умов (наприклад, значне навантаження);
* тестування продуктивності перевіряє швидкість реагування програми за умови звичайного навантаження;
* тестування навантаження перевіряє поведінку системи для різного рівня навантаження.
* тестування зручності перевіряє, чи удобно користуватися продуктом кінцевим споживачам;
* тестування підтримки оцінює, наскільки можна змінити програму без проблем;
* конфігураційне тестування визначає, наскільки складно програму перенести в інше середовище.

Функціональні тести виконують і коректні, і некоректні операції для перевірки поведінки ПЗ у звичайних і неочікуваних умовах. Нефункціональне тестування враховує параметри, що не завжди потребують успішності дії. Функціональні тести проводяться першими.

**6.1 Функціональне тестування**

Юніт-тестування проведено перевіркою роботи різних методів класів.

Наприклад, для перевірки методу getDetailByPerson класу Experience виконана перевірка при різних умовах (табл. 6.1).

Інтеграційне тестування передбачає виконання контрольного прикладу для системи в цілому. Частина такого тестування описана нижче.

Користувач натискає кнопку меню Суда. Програма виводить на екран список персон (рис. 6.1).

Таблиця 6.1 – Тестування методу getDetailByPerson класу Experience

|  |  |  |  |
| --- | --- | --- | --- |
| Умова | | | Результат |
| Приєднання до БД | | Ідентифікатор персони |  |
| Підключення до бази даних crew | Ідентифікатору персони дорівнює 0 | | Виведення попередження про не обрану персону |
| Підключення до бази даних crew | Ідентифікатору персони дорівнює 1 | | Виведення списку досвіду для персони з заданим ідентифікатором |
| Підключення до бази даних crew | Ідентифікатору персони дорівнює 2 | | Виведення списку досвіду для персони з заданим ідентифікатором |
| Відсутнє підключення до бази даних | Ідентифікатору персони дорівнює будь-якому значенню | | Виведення попередження про відсутність зв’язку з базою даних |
| Підключення до іншої бази даних | Ідентифікатору персони дорівнює будь-якому значенню | | Виведення попередження про неможливість виконати запит |

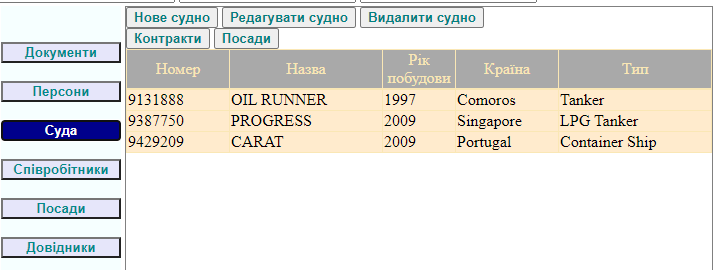


Рисунок 6.1 – Перегляд списку суден

Користувач натискає кнопку Нове судно, виводиться вікно, показане на рис.6.2.

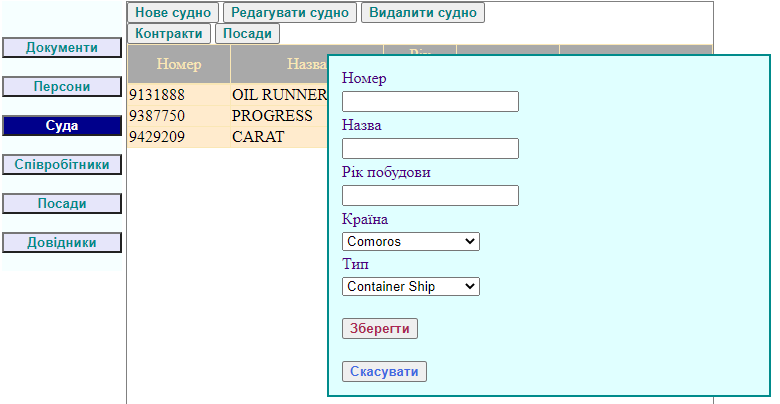


Рисунок 6.2 – Форма додавання судна

Користувач натискає кнопку Зберегти. Програма виводить попередження про відсутність вказаної назви судна (рис. 6.3).

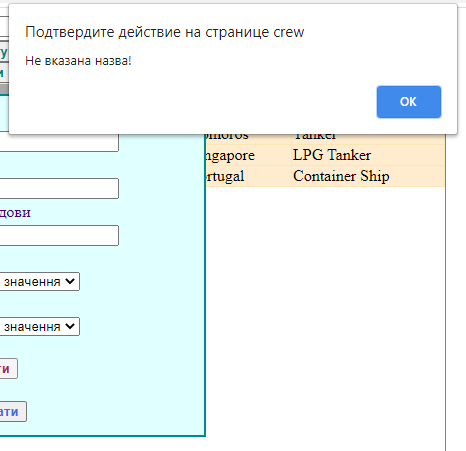


Рисунок 6.3 – Попередження про необхідність ввести дані

Користувач вносить дані, як показано на рис.6.4, та натискає кнопку Зберегти.

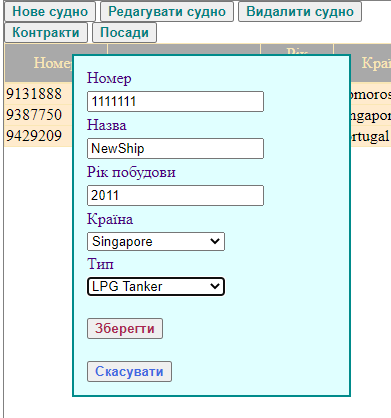


Рисунок 6.4 – Контрольні дані

Форма введення даних закривається, в списку суден з’являється нове судно (рис.6.5).

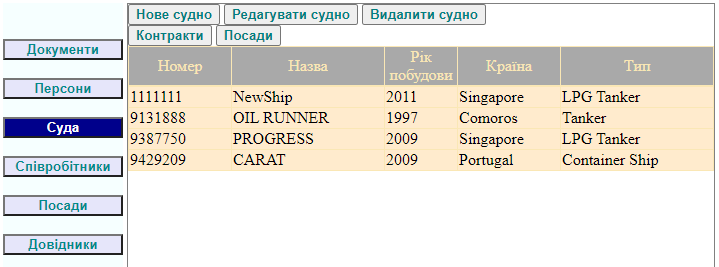


Рисунок 6.5 – Додане судно у списку

**6.2 Нефункціональне тестування**

Для проведення стрес-тестування виконаний вхід в систему одночасно 5 користувачів. Перевірений час отримання списків даних при натисканні на кнопку. З’ясовано, що час реакції системи є прийнятним для користувачів.

Кілька користувачів одночасно вводили нову персону. З’ясовано, що введення даних виконано без помилок.

Кілька користувачів одночасно вводили нові контракти для різних персони. З’ясовано, що введення даних виконано без помилок.

Для тестування продуктивності в БД за допомогою скрипта внесені дані про 5000 суден, 1500 персон та про 10000 контрактів. Визначено, що час відповіді на запити користувачів є прийнятним.

**6.3 Експериментальне дослідження досягнення мети проекту**

Пошук кандидатів на посаду є складною задачею з наступних причин:

* вакансія включає у себе кілька складових – посад. тип судна, необхідний пакет документів;
* необхідно знайти людей, які відповідають багатьом критеріям – мають досвід, відповідний посаді, або мають рекомендації на підвищення посади відповідно вакансії, мають необхідний пакет документів або можуть його підготувати у термін до відкриття вакансії, не мають діючого контракту и тощо.

У якості мети роботи зазначено зменшення кількості помилок при обранні кандидата на вільну посаду співробітниками крюінгової агенції.

Для перевірки досягнення цілі проведений експеримент. Співробітник агенції для заданих 5 вакансій мусив сформувати список кандидатів.

На другому етапі той же співробітник мав виконати таку ж задачу з використанням програми.

Результати експерименту показані у табл. 6.2. Підраховано кількість помилок двох видів. Помилка першого виду передбачає включення у список людей, що не відповідають критеріям. Помилка другого виду передбачає не включення у список людей, які можуть зайняти посаду. Також виміряний час на виконання задачі на обох етапах.

Таблиця 6.1 – Результати експериментального дослідження

|  |  |  |  |
| --- | --- | --- | --- |
| Етап дослідження | Кількість помилок 1-го виду | Кількість помилок 2-го виду | Затрачений час, години |
| 1) без використання програми | 3 | 4 | 8 |
| 2) з використанням програми | 0 | 1 | 1,2 |

Одна помилка 2-го роду з використанням програми викликана внесенням невірних даних про термін підготовки документу, наявність якого є необхідною для зайняття посади.

Основна частина часу при отриманні даних з використанням програми затрачений на введення даних про нову посаду та документи, які потрібні для зайняття її. Не врахований час на введення даних про персон та їх документи, оскільки ці дані вносяться під час поточної роботи співробітника.

Таким чином час скорчено на 85%, кількість помилок знижена на 85,7%.

**6.4 Висновки до розділу**

Описані можливі види тестування та наведені приклади тестів для розроблюваної системи Проведений експеримент, який підтвердив досягнення мети роботи.

**7 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА У НАДЗВИЧАЙНИХ СИТУАЦІЯХ**

**7.1 Опис розглянутих питань охорони праці**

У роботі розглянуті питання (дослідження) створення системи для підбору кандидатів на вакансії у крюінговій агенції, вирішення яких здійснюються проектною групою з різними фахівцями, а саме їх здійснює інженери-програмісти. Отже розглянемо питання охорони праці та безпеку у надзвичайних ситуаціях робоче місце інженер-програміста, яке розташоване у офісної будівлі.

Організація заходів із забезпечення безпечних умов праці та поведінки у надзвичайних ситуаціях засновується на наступних нормативних документах:

* ДСТУ 2293-2014. Охорона праці. Терміни та визначення основних понять [28];
* Наказ №30.12.2014  № 1417. Міністерства внутрішніх справ України . Про затвердження Правил пожежної безпеки в Україні [29];
* ДСН 3.3.6.037-99. Санітарні норми виробничого шуму, ультразвуку та інфразвуку;
* НПАОП 0.00-1.28-10. Правила охорони праці при експлуатації ЕОМ;
* ДСН 3.3.6.042-99. Санітарні норми мікроклімату виробничих приміщень;
* НАПБ Б.03.001-2204. Типові норми належності вогнегасників;
* ДБН В.2.5-28-2006. Природне і штучне освітлення.

Вивчення найважливіших заходів з охорони праці та безпеки у надзвичайних ситуаціях для працівників галузі інженерія програмного забезпечення наведено у Додатку Б та висвітлює наступні питання:

* організація та управління охороною праці на фірмі, де знаходиться робоче місце програміста;
* обґрунтування заходів з покращення умов охорони праці;
* розрахунок захисного занулення;
* надзвичайні ситуації та шляхи їх запобігання.

В ході роботи проаналізовано робоче місце інженера-програміста, вимоги до інтервалів робочих змін, необхідний рівень освітленості при умовах роботи на моніторі високої точності, та режим відпочинку.

Спираючись на проаналізовані дані, було розраховано захисне занулення та визначення категорії пожежонебезпечних приміщень.

В якості рекомендацій щодо поліпшення умов праці, основне – придбання та монтаж засобів сигналізації про порушення нормального функціонування виробничого обладнання, засобів аварійної зупинки, а також пристроїв, що дозволяють виключити виникнення небезпечних ситуацій при повному або частковому припиненні енергопостачання і подальшому його відновленні, а також організація навчання працівників наданню першої допомоги потерпілим на виробництві.

**7.2 Висновки за розділом**

В розділі розглянуто питання з охорони праці та безпеки у надзвичайних ситуаціях, а саме – основні вимоги до організації охорони праці, управління охороною праці, надзвичайні ситуації та шляхи їх запобігання. В ході роботи проаналізовано робоче місце інженера-програміста, вимоги до інтервалів робочих змін, необхідний рівень освітленості при умовах роботи на моніторі високої точності, та режим відпочинку.

**ВИСНОВКИ**

У процесі виконання кваліфікаційної роботи вирішені такі задачі:

* визначено критерії, які повинні бути враховані під час обрання кандидатів на посаду;
* проаналізовано функціональність існуючих автоматизованих платформ для визначення кандидатів на вакансію;
* розроблено модель предметної області підбору персоналу для роботи на судах;
* спроектовано та створено базу даних для зберігання потрібної інформації,
* визначені алгоритми виконання функції визначення списку кандидатів, що задовольняють множині критеріїв.

Наукова новизна отриманих результатів: вперше надано множинне представлення предметної області пошуку кандидатів на посаду, яке формалізує структуру даних та алгоритми пошуку за основними та додатковими критеріями.

Науково-практична цінність розробки полягає в тому, що автоматизовано пошук людей для зайняття посади з підвищеними формальними вимогами до складу наявних документів.

Показано експериментально, що завдяки використанню програми зменшено час на пошук кандидатів та зменшено кількість помилок у списках кандидатів на 85%.

Результати роботи доповідалися на XIII міжнародній науково-практичній конференції «Інформаційні технології і автоматизація – 2020».

**СПИСОК ЛІТЕРАТУРИ**

1. Синяева Л. П., Герасимова Е. А. Компетентностный подход в подборе персонала. Концепт. – 2013. Спецвыпуск №04. С. 1-5.
2. Комлев В. А., Носов А. Л. Комплексный анализ основных показателей эффективности подбора сотрудников. Концепт. 2016. № 02 (февраль). С. 1-6.
3. Армстронг М. Практика управления человеческими ресурсами. 8-е изд. – СПб.: Питер, 2007. – 832 с.
4. Федорова А.С. Сучасні проблеми у процесі відбору та найму персоналу в організації. Вісник КНУДТ. – 2014. – №1(75). – С. 187 - 192.
5. Данилова А. С. К вопросу о человеческом потенциале, как конкурентном преимуществе организации. В мире научных открытий. – 2014. – № 7.2 (55). – С. 924–935.
6. Козина И.М. подбор персонала: внутренний и внешний рынки человек. Сообщество. Управление. 2007. №2. С. 66-78.
7. Кадровая служба. URL: http://xn----7sbabednt4ap3axr3a0p.xn--p1ai/%D1% 81%D1%82%D0%B0%D1%82%D1%8C%D0%B8-%D0%B4%D0% BB%D1%8F-%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BE%D0%B4%D0%B0% D1%82%D0%B5%D0%BB%D0%B5%D0%B9/%D0%BE%D1%86%D0%B5%D0%BD%D0%BA%D0%B0-%D1%80%D1%8B%D0%BD%D0%BA%D0%B0-%D1%82%D1%80%D1%83%D0%B4%D0%B0-%D0%BF%D1%80%D0%B8-%D0%BF%D0%BE%D0%B4%D0%B1%D0%BE%D1%80%D0%B5-%D0% BF%D0%B5%D1%80%D1%81%D0%BE%D0%BD%D0%B0%D0%BB%D0%B0. Дата звертання – 13.09.2020.
8. Анализ рынка труда: руководство для HR-директора. URL: https://www.hr-director.ru/rynok\_truda. Дата звертання – 15.09.2020.
9. Жариков В. В., Лыжникова М. А. Инновационные технологии отбора персонала. ЭКОНОМИНФО. 2014. №21. С. 71-79.
10. Cvpool. Відеоспівбесіди кандидатів. URL: https://www.cvpool.com.ua. Дата звертання – 03.09.2020.
11. Управление персоналом. URL: http://l-stream.com.ua/ru/kryuing/ Дата звертання – 25.09.2020.
12. Crew Selection System for Recruiters and effective Job Search tools for Seafarers. URL: <https://crewell.net/en/> Дата звертання – 17.09.2020.
13. Ногин В. Д. Принятие решений в многокритериальной среде: количественный подход. – М.: ФИЗМАТЛИТ, 2002. – 144 с.
14. Многокритериальный выбор управленческих решений URL: https://infopedia.su/16x34ac.html. Дата звертання – 14.09.2020.
15. Управленческие решения. URL: http://eos.ibi.spb.ru/umk/10\_8/ 5/5\_R1\_T4.html#2. Дата звертання – 07.09.2020.
16. [Сравнение типов требований: классика и Agile](http://myalm.ru/news/%D0%A1%D1%80%D0%B0%D0%B2%D0%BD%D0%B5%D0%BD%D0%B8%D0%B5-%D1%82%D0%B8%D0%BF%D0%BE%D0%B2-%D1%82%D1%80%D0%B5%D0%B1%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B9-%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%B8%D0%BA%D0%B0-%D0%B8-Agile). URL: https://myalm.ru/news/%d0%a1%d1%80%d0%b0%d0%b2%d0%bd%d0%b5%d0% bd%d0%b8%d0%b5-%d1%82%d0%b8%d0%bf%d0%be%d0%b2-%d1%82%d1% 80%d0%b5%d0%b1%d0%be%d0%b2%d0%b0%d0%bd%d0%b8%d0%b9-%d0% bf%d0%be%d0%bb%d1%8c%d0%b7%d0%be%d0%b2%d0%b0%d1%82%d0% b5%d0%bb%d1%8c%d1%81%d0%ba%d0%b8%d0%b5-%d0%b8%d1%81%d1% 82%d0%be%d1%80%d0%b8%d0%b8. Дата звертання – 13.09.2020.
17. Построение реляционной структуры из ER-модели. URL: https://habr.com/ru/post/50312/ Дата звертання – 17.10.2020.
18. Что такое ER-диаграмма и как ее создать? URL: https://www.lucidchart.com/pages/ru/erd-%D0%B4%D0%B8%D0%B0%D0% B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0. Дата звертання – 08.10.2020.
19. Понятие ER-модели. Понятие сущности (entity). Атрибуты. Виды атрибутов. URL: https://www.bestprog.net/ru/2019/01/24/the-concept-of-er-model-the-concept-of-essence-and-communication-attributes-attribute-types-ru/ Дата звертання – 13.10.2020.
20. Основи розробки баз даних. URL: https://support.microsoft.com/uk-ua/office/%D0%BE%D1%81%D0%BD%D0%BE%D0%B2%D0%B8-%D1%80%D0%BE%D0%B7%D1%80%D0%BE%D0%B1%D0%BA%D0%B8-%D0%B1%D0%B0%D0%B7-%D0%B4%D0%B0%D0%BD%D0%B8%D1%85-eb2159cf-1e30-401a-8084-bd4f9c9ca1f5#bmterms. Дата звертання – 10.10.2020.
21. Kruchten P. The Rational Unified Process: An Introduction. Third Edition. Addison-Wesley Professional, 2004 - 310 р.
22. Bass L., Clements P., Kazman R. Software Architecture in Practice. Second Edition. Аddison-Wesley Professional, 2013. – 587 р.
23. McGovern J., et al. A Practical Guide to Enterprise Architecture. Prentice Hall, 2004ю – 306 р.
24. Концепция: Архитектура программного обеспечения. URL: http://dit.isuct.ru/Publish\_RUP/core.base\_rup/guidances/concepts/software\_architecture\_4269A354.html
25. Buschmann F., Meunier R, Rohnert H., Sommerlad P., Stahl M. Pattern-Oriented Software Architecture Volume 1: A System of Patterns, New York, NY: John Wiley and Sons, Inc. 1996. – 476 р.
26. Архитектура MVC - Model, View, Controller. URL: https://webshake.ru/oop-v-php-prodvinutyj-kurs/arhitektura-prilozheniya-i-pattern-mvc. Дата звертання – 03.11.2020.
27. Функціональне та нефункціональне тестування: що потрібно знати. URL: https://codeguida.com/post/2379. Дата звертання – 21.11.2020.
28. Національний стандарт України. Охорона праці. Терміни та визначення основних понять. ДСТУ 2293:2014. Мінекономрозвитку України. - Київ, 2015. – 18 с.
29. Про затвердження Правил пожежної безпеки в Україні. Наказ №30.12.2014  № 1417. Міністерства внутрішніх справ України. URL: https://zakon.rada.gov.ua/laws/show/z0252-15#Text. Дата звертання – 21.11.2020.