Міністерство освіти і науки України Сумський державний університет
Навчально-науковий інститут бізнес-технологій «УАБС» Кафедра
економічної кібернетики

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

на тему «Розробка прототипу інформаційної системи обслуговування клієнтів центру зайнятості»

Виконав студент 2 курсу, групи ЕК.м-61а
(номер курсу) (шифр групи)

Спеціальності 051«Економіка («Економічна кібернетика»)»

_____Харченко В.Л._____
(прізвище, ініціали студента)

Керівник _к.т.н., доцент Яценко В.В.___
(посада, науковий ступінь, прізвище, ініціали)

Суми – 2018 рік

ЗМІСТ

ВСТУП

Кваліфікаційна робота спрямована на розробку прототипу веб-орієнтованої інформаційної системи обслуговування клієнтів центру зайнятості.

Праця є доцільною діяльністю людини, у процесі якої вона видозмінює предмети природи і пристосовує їх для задоволення своїх потреб. Вона є природною необхідністю, неодмінною умовою існування людей, використовується в усіх галузях економіки, і від її ефективності залежить добробут нації. Промисловість дуже специфічна сфера народного господарства, тому функціонування ринку трудових ресурсів у цій сфері має свої особливості. Саме тому особливу увагу потрібно приділити розгляду питання трудових ресурсів підприємства та їхнім проблемам, у силу низьких рівнів продуктивності праці промисловість продовжує залишатись одним із основних споживачів трудових ресурсів, потерпаючи від його браку у періоди пікових навантажень.

Трудові ресурси мають специфічний вплив на рівень конкурентоспроможності підприємства. Достатня забезпеченість підприємств трудовими ресурсами відповідного рівня кваліфікації та професійних навичок, їхнє раціональне використання, високий рівень продуктивності праці мають велике значення для підприємства.

За статистичними даними, на початку 2017 року 62% дорослого населення України використовували ресурси Інтернет. Частка користувачів Інтернет серед людей 18-39 років в Україні сягнула 91%. Старший вік та проживання у сільській місцевості значно зменшують вірогідність користування Інтернетом [6].

Нині все більше компаній починає використовувати мережу Інтернет для взаємодії з клієнтами, що надає зручності користувачам веб-ресурсів підприємств та покращує економічні показники діяльності від електронної

комерції. Цей зв’язок між замовником послуг і компанією забезпечується за допомогою довідкових, комерційних та розважальних веб-ресурсів. Такий спосіб он-лайн спосіб взаємодії з клієнтом набагато ефективніший ніж звичайні канали комунікації: пошта, телефон, факс.

Необхідність оперувати великими обсягами інформації про трудові ресурси обумовлює актуальність автоматизації діяльності центрів зайнятості.

Метою даної дипломної роботи є розробка прототипу веб-орієнтованої інформаційної системи обслуговування клієнтів центру зайнятості.

Об’єктом дослідження є діяльність Сумського міського центру зайнятості.

Предметом дослідження виступає веб-орієнтована інформаційна система пошуку робочих вакансій та надання кадрових пропозицій роботодавцям.

Для досягнення мети кваліфікаційної роботи необхідне розв’язати наступні задачі:

– дослідити діяльність державної установи "Сумський міський центр зайнятості";

– проаналізувати стан автоматизації об’єкту дослідження;

– сформувати вимоги до системи автоматизації;

– спроектувати систему автоматизації Сумського міського центру зайнятості;

– побудувати бізнес-моделі задачі;

– обрати інформаційну архітектуру та технології вирішення задачі;

– описати необхідне забезпечення функціональної частини;

– побудувати структуру та описати особливості реалізації інформаційного та програмного забезпечення;

– реалізувати прототип веб-орієнтованої інформаційної системи обслуговування клієнтів;

– оцінити очікуваний ефект від впровадження системи автоматизації.

РОЗДІЛ 1 ДОСЛІДЖЕННЯ СТАНУ ОБ'ЄКТА АВТОМАТИЗАЦІЇ ТА ФОРМУВАННЯ ВИМОГ ДО СИСТЕМИ АВТОМАТИЗАЦІЇ

1.1 Характеристика ДУ "Сумський міський центр зайнятості"

Центр зайнятості населення є активним посередником на ринку праці між роботодавцем та шукачем роботи. Під час проходження переддипломної практики в Сумському міському центрі зайнятості було проведено дослідження основних напрямів роботи державної установи [5], а саме:

– надання послуг із підходящої роботи для безробітних і підбір персоналу для роботодавців на безоплатній основі;

– надання у випадку безробіття певних виплат, доки людина шукає роботу у зв'язку з тимчасовою втратою її;

– сприяння працевлаштуванню населення;

– надання юридичної допомоги на безоплатній основі.

Основними завданнями Державної служби зайнятості є:

1. Реалізація державної політики у сфері зайнятості населення та трудової міграції.

2. Сприяння громадянам у підборі підходящої роботи та надання роботодавцям послуг з добору працівників.

3. Стимулювання роботодавців до створення нових робочих місць.

4. Підвищення конкурентоспроможності безробітних на ринку праці.

5. Залучення безробітних до тимчасової зайнятості (до громадських та інших робіт тимчасового характеру).

6. Підтримка безробітних в організації підприємницької діяльності.

7. Видача роботодавцям дозволів на застосування праці іноземців та осіб без громадянства [5].

Розглянемо більш детально об'єкт автоматизації – діяльність консультантів відділу обробки даних.

Завданням консультанта є підбір персоналу з врахуванням побажань клієнта та стану ринку праці.

На рисунку 1.1. наводиться модель надання послуг на основі 2 етапів.

Рисунок 1.1 – Модель надання рекрутингових послуг роботодавцю

Модель надання консультаційних послуг з пошуку персоналу складається з двох етапів. Після переговорів з клієнтом відбувається 1-й етап роботи консультанта:

– дослідження кадрового ринку;

– пошук персоналу;

– оцінка персоналу;

– підготовка резюме кандидатів і звіту про виконану роботу;

– подання кандидатів.

У разі позитивного результату пошуку здійснюється перехід до другого етапу (працевлаштування кандидата), інакше повертаємось до 1-го етапу.

В функції відділу обробки даних входить обробка вхідної та вихідної інформації:

1. Формування та ведення журналу прийому-звільнення кандидатів;

2. Ведення особистих карток кандидатів;

3. Ведення архіву договорів;

4. Підготовка звітної документації.

В даній кваліфікаційній роботі вирішуються такі завдання:

– аналіз бізнес-процесів об'єкта автоматизації;

– вироблення пропозицій щодо їх оптимізації;

– моделювання та проектування веб-орієнтованої інформаційної системи центру зайнятості;

– реалізація веб-ресурсу, який забезпечить вирішення виявлених проблем з урахуванням пропозицій по оптимізації і вже наявного досвіду в даній галузі [32].

Нині Сумський центр зайнятості є важливою установою для громадян, який надає офіційне працевлаштування для безробітних та кваліфікованих кадрів для роботодавців. Також дана державна установа надає змогу пройти безоплатні кваліфікаційні курси для певних професій наприклад: водій, швачка, пекар, тощо, при наявності вакансії на дану професію.

Нещодавно для отримання послуг закладу клієнти відвідували його та користувалися послугами персоналу закладу. На зараз Сумський міський центр зайнятості має сайт, який надає дані послуги віддалено, що економить час як клієнту так і працівнику установи [36].

Розглянемо організаційну структуру даного державного закладу (рис. 1.2).

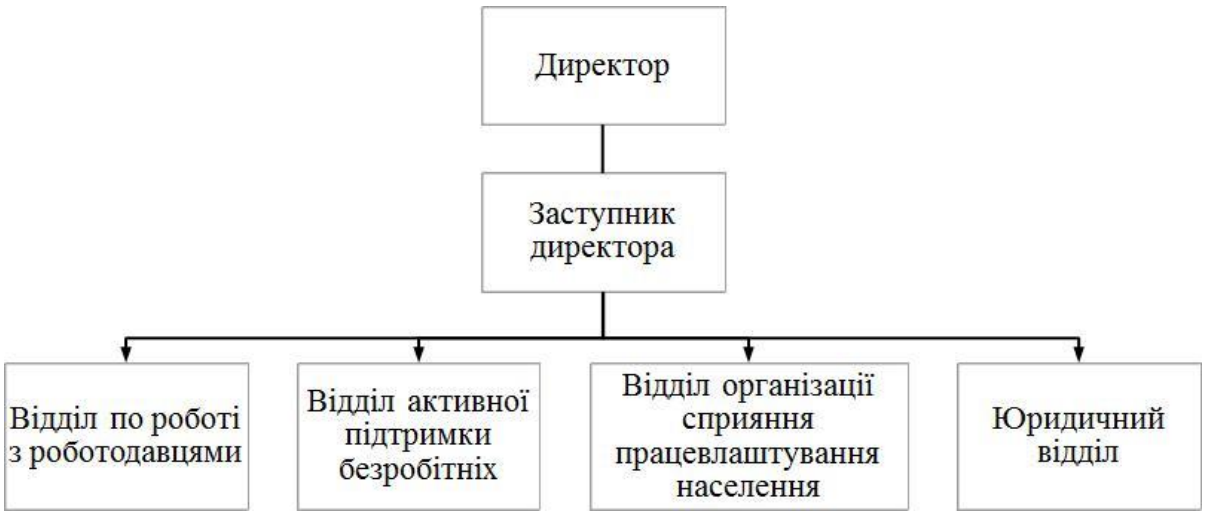


Рисунок 1.2 – Організаційна структура ДУ "Сумський міський центр зайнятості"

Структура закладу спланована таким чином, що директор є головною особою, що приймає основні управлінські рішення. В свою чергу заступник директора відіграє також важливу роль, як людина, що відповідє за діяльність всіх відділів установи. Тобто в центрі зайнятості присутня лінійна структура управління.

Відділ підтримки безробітних займається працевлаштуванням людей які не мають роботи, або тимчасово безробітні. Для обслуговування даних клієнтів є персонал, який допомагає знайти вільні вакансії:

– приймальна група, яка робить запис безробітної особи,

– спеціаліст, до якого звертається клієнт після реєстрації, та надає свої персональні дані, після чого здійснює підбір вільних вакансій.

Відділ по роботі з роботодавцями контактує з роботодавцями та підбирає з числа людей, які шукають роботу відповідно до вимог роботодавця, попередньо домовляючись з ним про дату зустрічі з претендентами на вакансії на базі підприємства чи центру зайнятості. При цьому намагаються якнайшвидше підібрати необхідну кількість персоналу. Великою перевагою є те, що роботодавець не втрачає кошти через простій вакансій [16].

Відділ сприяння працевлаштування населення допомагає знайти роботу безробітним, проводячи постійний аналіз попиту і пропозицій на ринку, з подальшим забезпеченням координації дій в подальшому, та допомогою отримання нової кваліфікації або перекваліфікування для подальшої роботи за новою спеціальністю.

Юридичний відділ, в свою чергу, надає правові консультації, допомогу в питаннях, де є необхідність в знаннях законів України, Конституції та інших законодавчих нормативних актів с питань праці.

Всі відділи поєднані інформаційною системою, з якою працює персонал закладу. Кожен працівник має обмежений доступ до системи, і обробляє лише ту інформацію, яка відноситься до його службових обов’язків.

Недоліком наявної інформаційної системи є те, що вона працює від центрального серверу, який знаходиться в м. Київ, і у випадку припинення його роботи, або збоїв, весь робочий процес установи може призупинити роботу установи. Продовження роботи стає можливим після налагодження ІС відділом ІТ в м. Київ.

Другою складовою інформаційного забезпечення установи є внутрішній сайт – в загальному випадку, веб-інтерфейс для доступу співробітника до корпоративних даних і додатків. Часто корпоративний портал сприймається, як синонім Інтернету. Альтернативна точка зору полягає в тому, що корпоративний портал — це лише видима для користувача частина Інтернету. Основним недоліком сайту є незручний інтерфейс користувача (велика кількість посилань, контекстних меню, вікон), що робить цей сайт важким у розумінні, особливо осіб старшого віку. Тому багато людей не використовує його та особисто відвідує заклад для отримання інформації про наявні вакансії та залишає своє резюме.

Також установа використовує додаткове програмне забезпечення з пакету офісних програм компанії Майкрософт: Microsoft Word 2007, Microsoft Excel 2007, для того щоб працювати з шаблонами документів, для подальшого їх друку [18]. Для обміну службовою інформацією використовується Microsoft Outlook Express – персональний інформаційний менеджер з функціями поштового клієнта Groupware. Працівники установи використовують його не тільки для роботи з електронною поштою. Microsoft Outlook Express для них також є повноцінним органайзером, що надає функції календаря, планування завдань, записника і менеджера контактів. Крім того, Outlook дозволяє відстежувати роботу з документами пакету Microsoft Office для автоматичного складання щоденника роботи [4].

В теперішній час зростає потреба в послугах центрів зайнятості. Роботодавці – замовники послуг – потребують здійснювати швидкий пошук висококваліфікованого персоналу з максимальним задоволенням вимог пошуку.

Для ефективної роботи з клієнтами необхідно оптимізувати роботу з обслуговування клієнтів, а саме зробити його зручним і зрозумілим у використанні – це дозволить прискорити процеси пошуку інформації, але ніяк не підвищити якість пошуку, так як при цьому не виключається людський фактор, але можливо в стислі терміни підібрати кандидата.

Сучасний рівень інформаційних технологій дозволяє просто і ефективно вирішити існуючі проблеми шляхом застосування спеціалізованих інформаційних систем та Інтернет.

Питання про використання автоматизованих інформаційних систем для центрів зайнятості існує вже досить довго і є особливо актуальним на сучасному етапі розвитку економіки України [13]. В умовах сучасного суспільства, в якому лідируючі позиції будь-якої організації, в тому числі центру зайнятості визначаються в першу чергу його можливостями щодо доступу, зберіганню та якісної обробки інформації, особливу значущість набуває всебічне використання передових досягнень в сфері інформаційних технологій.

1.2 Аналіз стану автоматизації об’єкта дослідження

Центри зайнятості України охоплені і працюють і використовують Єдину інформаційно-аналітичну систему(ЄАІС) де знаходяться загальна інформаційна база діяльності установи. Якщо більш детально розглянути цю

інформаційно-аналітичну систему, то можна побачити що вона складається з модулів, які постійно використовуються працівниками певних відділів центру зайнятості, а саме [8]:

1) "Соціальні послуги та Фонд";
2) "Управління фінансами";
3) "Бухгалтерський облік";
4) "Кадри";
5) "Документообіг та канцелярія";

6) "Адміністрування та взаємодія";

7) "Ведення нормативно-довідкової інформації".

Останній модуль є додатком до всіх інших модулів, і відкритий в для всього персоналу. В ньому збережена інформація для всіх робітників, яка дає їм певні інформаційні відомості, а також повідомлення якими вони керуються під час роботи [15].

Модуль "Соціальні послуги та Фонд" в собі містить:

− інформацію про забезпечення центрами зайнятості професійного навчання;

− зв'язок з роботодавцями про вакансії, формування ФЗДССУВБ, організація фахового навчання для наявних вільних місць роботи та проходження стажування;

– надання інформаційних, консультаційних і профорієнтаційних послуг;

– пошук вакансій ;

– підбирання кадрів на замовлення роботодавців;

– інформаційна підтримка безробітних та їх соціальний захист.

На ринку клієнтських аналітичних систем пошуку персоналу на ринку існують продукти альтернативи, наприклад: "Бітрікс24", "Рекрутер".

"Бітрікс24" – CRM-система, для роботи з клієнтською базою. Вона дозволяє заносити туди інформацію про замовників: контакти, особисті дані, історію звернень і покупок, взаємні зобов'язання. Дана система призначена для взаємодії з великою кількістю клієнтів. Також слід засначити, що система має зрозумілий інтерфейс та майже повністю автоматизована, і має наступний вигляд(рис. 1.3) [3].

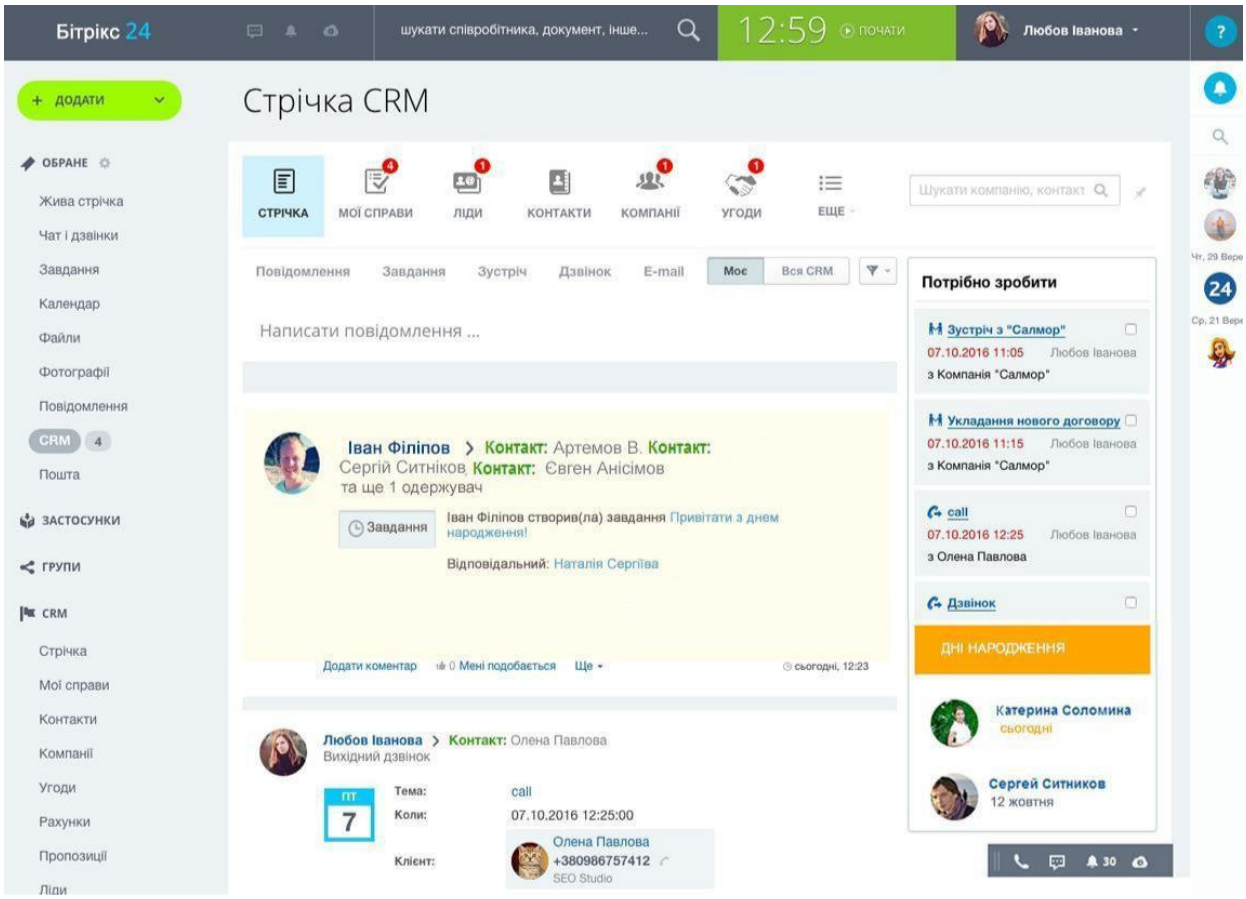


Рисунок 1.3 – CRM-система " Бітрікс24"

CRM-система – система управління взаємовідносин с клієнтами, а якщо розкривати більш детально то це прикладне програмне забезпечення для організацій, призначене для автоматизації стратегій взаємодії з замовниками (клієнтами), зокрема для підвищення рівня продажів, оптимізації маркетингу і поліпшення обслуговування клієнтів шляхом збереження інформації про клієнтів і історію взаємин з ними, встановлення і поліпшення бізнес-процесів і подальшого аналізу результатів [3].

Автоматизована система Рекрутер – програма для автоматизації роботи відділу з підбору персоналу та рекрутингового або кадрового агентства.

Від відкриття вакантної посади до її заповнення Рекрутер проводить по всіх етапах конкурсного відбору та реально допомагає вибирати кращих фахівців. Автоматизована система підбору персоналу складається з наступних розділів: конкурсний відбір, ділове листування, договори і платежі, вакансії, кандидати на роботу.

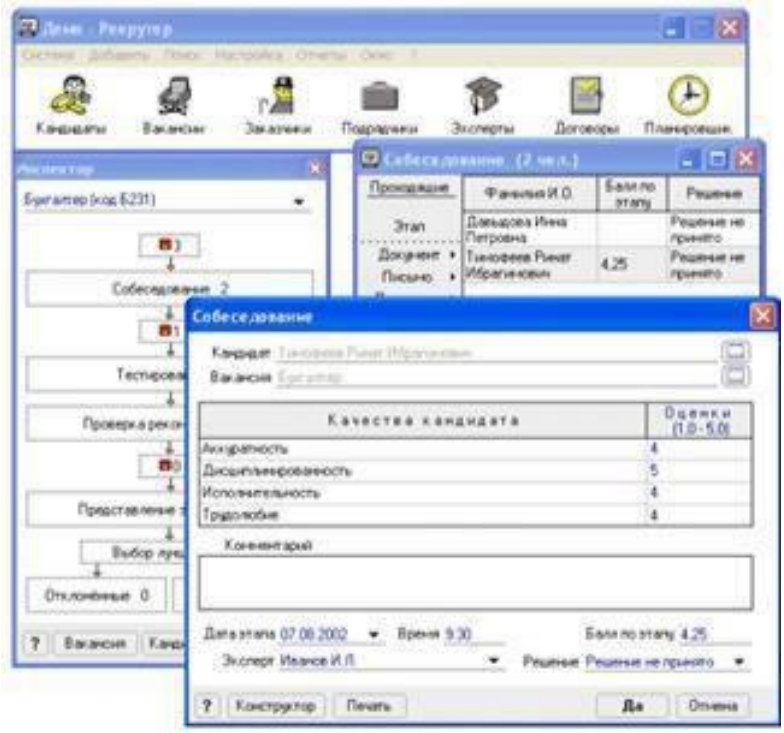


Рисунок 1.4 – Автоматизована система підбору персоналу "Рекрутер"

Але основною метою проходження переддипломної практики було розроблення веб-сторінки, яка була би більш зрозумілою для користувачів, та мала свою базу даних для зберігання заявок клієнтів, а в подальшому працівник установи міг би контактувати з особою, яка залишила свою анкету [28].

Наряду з центром зайнятості існує ряд веб-ресурсів де людина може підібрати собі вільну вакансію чи залишити анкету для пошуку робочої сили зі сторони роботодавця.

Work.ua – сайт пошуку роботи на Україні (рис. 1.5). Згідно з рейтингом top.bigmir, в пошуку роботи щомісяця понад 3,5 млн осіб відвідують Work.ua, більш 115 тис. Роботодавців розміщують свої вакансії і знаходять співробітників за допомогою Work.ua. На сайті зібрана база актуальних резюме - понад 1,6 млн, і вакансій - більше 69 тисяч (дані на квітень 2016 року). Дані про кількість вакансій і резюме розташовані на самому сайті у відповідному розділі. Все резюме і вакансії перевіряються вручну.

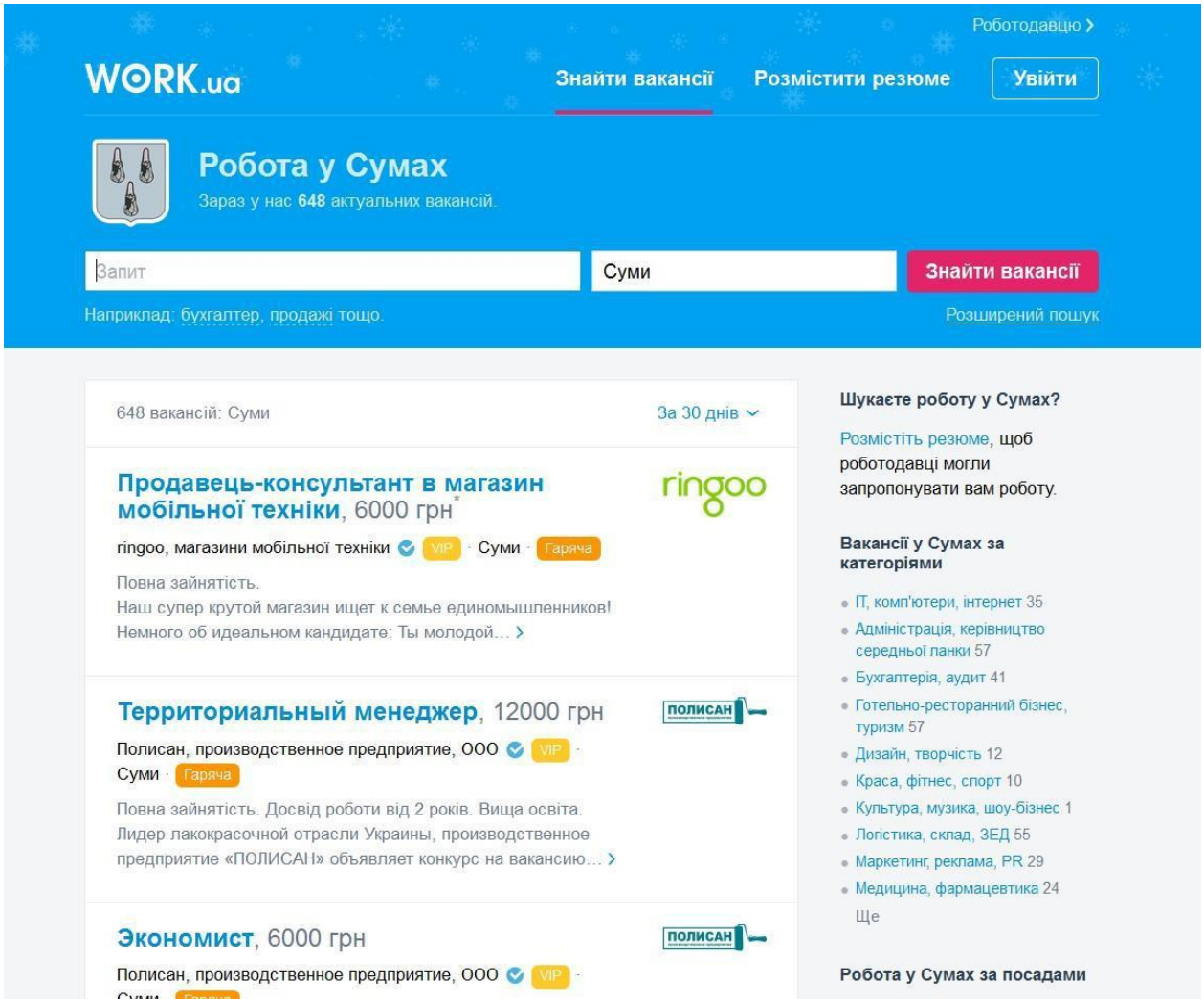


Рисунок 1.5 – Інтерфейс сайту Work.ua

Jobs.ua – ресурс де можливо підібрати роботу не тільки в Україні та закордоном (Рис. 1.6). Досвід роботи сайту в даній сфері з 1996-го року, на сторінках ресурсу зібрані тисячі пропозицій від сотень найвідоміших вітчизняних роботодавців. Має зручний пошуковий інтерфейс. У компанії "Робота Плюс" за плечима не тільки багаторічний успішний досвід роботи на ринку працевлаштування, унікальна система перевірки та оновлення вакансій, він робить пошуки зручними, швидкими і гарантовано безпечними. Ресурс має корисну інформація по темі: "Робота в Україні".

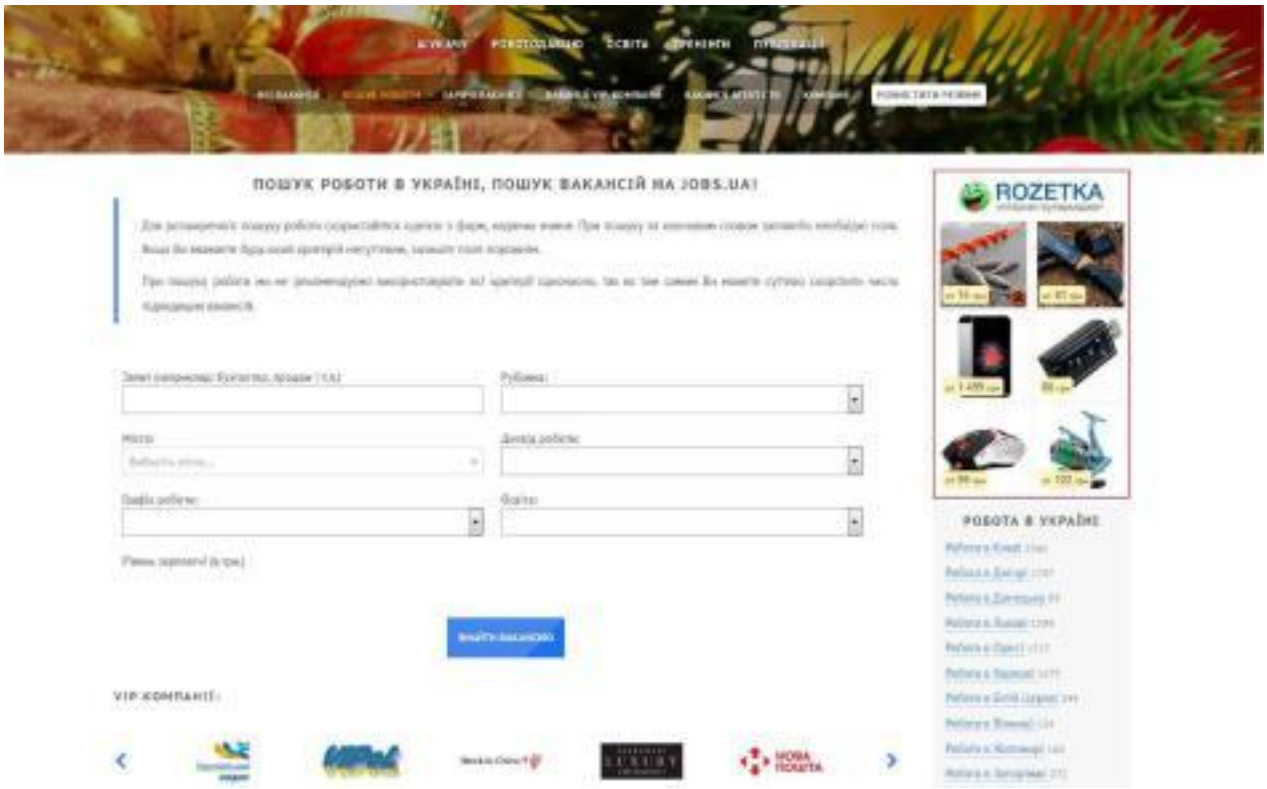


Рисунок 1.6 – Представлення сайту Jobs.ua

Недоліком цих веб-орієнтованих інформаційних ресурсів є надлишок реклами, пошук працівників є платним, знайдена робота не завжди є офіційною і є ризик отримати не то що необхідно.

Для створення сайту можливо скористатись системою управління контентом для створення сайту. На сьогодні ринок пропонує десятки різноманітних рішень, що відрізняються як за функціоналом, так і за вартістю.

На сьогоднішній день зростає динаміка використання інформаційних систем, що відносяться до класу CMS (content management system) – комп'ютерних програм, які використовуються для забезпечення і організації спільного процесу створення, редагування і управління вмістом, інакше – контентом [35].

З усієї маси веб-сайтів, які наповнюють Інтернет, близько 31% використовують CMS. Інструменти систем управління контентом роблять розробку сайтів простіше. Існує велика кількість CMS, з яких можна обрати найбільш зручну та відповідну до цілей і специфіки веб-сайту [14].

В теперішній час існує певна група систем, які найбільш частіше обираються розробниками для створення веб-орієнтованих інформаційних систем (рис.).

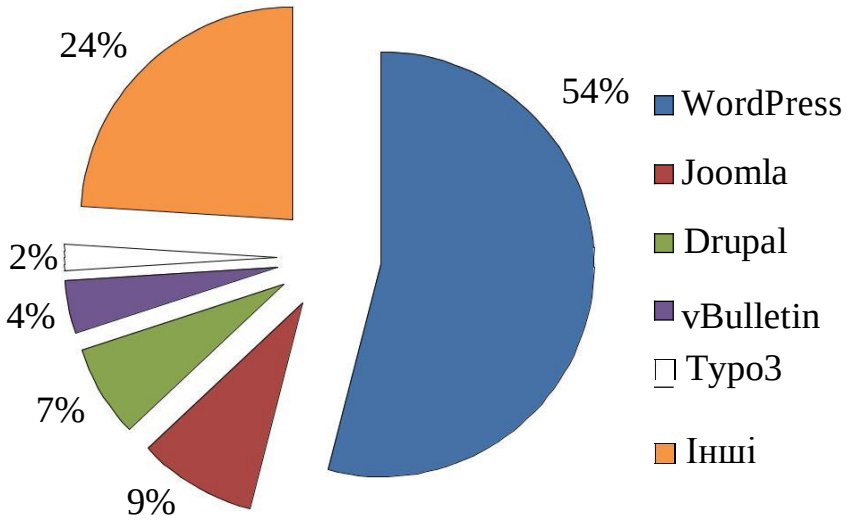


Рисунок 1.7 – Ринкові частки різних CMS на сучасному етапі

Розглянемо кожну систему управління контентом окремо.

WordPress – CMS спочатку зарекомендувала себе як інноваційна блог-платформа з високою юзабіліті. Але розвиток функціоналу системи забезпечило їй високу популярність також і серед інших форматів веб-сайтів. Сьогодні індустрія веб-дизайну забезпечена високим попитом на послуги в області розробки під WordPress.

Перевагами даної CMS є її по популярність серед користувачів, що вже є вагомим доказом переваги даної системи над іншими; найбільш широкий набір плагінів, віджетів для галерей, форумів, багатомовність; наявність WYSIWYG редактора, полегшить життя тим, у кого є проблеми з HTML-розміткою та іншими мовами; не обов’язкова наявність технічного досвіду. Адміністрування набагато простіше, ніж в інших CMS: PHP і CSS файли можна редагувати безпосередньо в кабінеті адміністратора. Наприклад,

можна легко вставити текст з будь-якого текстового редактора, на відміну від Drupal або Joomla; також WordPress – досить сильний інструмент для розробників і дизайнерів, які створюють сайти для клієнтів.

Проте існує ряд недоліків, до них відносяться: різноманітне поле для вибору, тому головна проблема - як використовувати всі її можливості правильно, ця CMS буде працювати набагато краще, якщо правильно кастомізувати настройки; якщо ви новачок, то можете зіткнутися з деякими проблемами при установці, не дивлячись на поширену думку про найлегшому процесі установки.

Joomla – наступна за популярністю CMS, якою користуються 9% користувачів. Це щось середнє між великими можливостями орієнтованого на розробників Drupal і простотою WordPress, але з більш широкими можливостями для розробки. Незважаючи на це, Joomla має дружній користувачеві інтерфейсом.

Перевагами цієї системи можна вважати її повноту і повноцінним інструментів для розробки в порівнянні з Drupal; підтримка протоколів контролю доступу (OpenID, LDAP, Gmail.com); наявність зручної адмін-панелі з широким набором функцій: шаблони, стилі, управління меню і так далі; простий процес установки для недосвідчених користувачів; слід вказати також що ця CMS має досить зрозумілий інтерфейс.

Недоліками Joomla можна вважати те, що система менш функціональна, незважаючи на її універсальність; досить велика кількість платних плагінів в порівнянні з WordPress. Для недосвідчених користувачів може здатися, що Joomla володіє безліччю непотрібних функцій, а для досвідчених користувачів вона занадто проста.

vBulletin – цій CMS віддають перевагу близько 4% користувачів. Але говорячи про популярність, варто відзначити, що ця система в останні роки втрачає свої ринкові позиції. vBulletin забезпечує користувача інструментарієм для створення і адміністрування форумів та блогів.

Перевагами CMS vBulletin є: передове і інноваційне рішення для створення форумів; багато скінів і чистий код; проста і симпатична панель адміністратора; високий рівень безпеки, завдяки чому на цій платформі створена велика кількість форумів; безліч компонентів для SEO.

Недоліками цієї системи являються: велика кількість опцій в системі, що може стати проблемою для недосвідченого користувача; якщо вам потрібно буде імпортувати дані з інших електронних дошок оголошень, то потрібно буде встановлювати окремо модифікаційний патч.

TYPO3 – використовується 2%-ми веб-сайтів, які використовують CMS. Це професійний і потужний інструмент, багатий різними опціями.

Перевагами TYPO3 являється наступне: система надає широкий набір опцій, які можуть бути адаптовані або розширені; простота установки; ефективне управління контентом і графічними елементами; вдосконалений логін для користувачів і адміністраторів; ви можете легко додавати контент, сторінки, документацію, зображення, навіть якщо ви недосвідчений користувач, нічого не знає про HTML і веб-розмітці.

Недоліками системи TYPO3: система вимагає гарного хостингу, оскільки система велика; TYPO3 досить важко вивчати; багато недоліків в коді, які розробники пропонують усунути хешуванням, але насправді це ніяк не вирішує проблему [24].

На ринку безкоштовних CMS також багато гарних, але менш популярних систем.

Diafan.CMS – проста і при цьому функціональна система управління сайтами, що дозволяє створювати web-сторінки. Завдяки вмісту найбільш затребуваних модулів в одній редакції керувати нею можна без глибоких знань в програмуванні. Різко набула популярності серед програмістів у 2015р., і на зараз також має своїх прихильників.

Amiro.CMS – універсальна платформа, що дозволяє легко створювати і підтримувати професійні веб-сайти практично будь-якого рівня складності. Вона включає весь інструментарій, необхідний для ефективної роботи

сучасного інтернет-ресурсу. Завдяки Amiro.CMS працює більше 20000 сайтів і понад 7000 інтернет-магазинів. Ця система управління сайтом входить в 5-ку лідерів російського ринку "коробкових" CMS.

DJEM – безкоштовна платформа з відкритим кодом, призначена безпосередньо для створення і подальшого розвитку складних за структурою взаємозв'язків, систем управління даними. Дан CMS створює такі статичні сторінки, динамічний контент яких дозволяє конструювати не прості web-сторінки, сайти або портали, а масштабовані системи управління контентом Платформа DJEM відповідає базовим вимогам CMS на професійному рівні. DJEM – найбільш економічна CMS-система, що передбачає підтримку сайтів будь-якої складності, серед яких і сайти з нестандартною бізнес-логікою.

NetCat CMS - система управління сайтами (Content Management System), розробляється з 1999 року. Засновником системи є Васильєв Дмитро Євгенович. Згідно з аналізом, проведеним в липні 2012 року порталами 3DNews.ru і CMS Magazine, NetCat займає друге місце серед комерційних CMS на російськомовних сайтах. Його образ активно використовується в промо-матеріалах, а також від NetCat ведуться спільноти в соціальних мережах Facebook, Twitter та ВКонтакте.

HostCMS - комерційна система керування вмістом (CMS), розроблена російською компанією "Хостмейк" і є однією з найстаріших CMS на російському ринку. Має вбудований ORM, будівник запитів QueryBuilder. Використовує технологію AJAX в клієнтському розділі і центрі адміністрування. З версії 6.0 надає можливість редагування вмісту сторінок без переходу в центр адміністрування. Модулі, що входять в HostCMS, дозволять створити сайт практично будь-якого призначення. Самі ж розробники системи позиціонують свій продукт як інструмент для розробки сайтів різної спрямованості – від невеликих корпоративних сайтів до контент-проектів і інтернет-магазинів.

MODX – це безкоштовна професійна система управління вмістом (CMS) і фреймворк для веб-додатків, призначена для забезпечення і

організації спільного процесу створення, редагування і управління контентом (тобто вмістом) сайтів [23].

1.3 Формування вимог до системи автоматизації

Відповідно до мети дипломного проекту веб-розробка повинна бути зорієнтована на побудову зручної, надійної і в той же час швидкої інформаційної системи, за допомогою якої користувач, тобто безробітна особа чи роботодавець зможе с легкістю зорієнтуватись і виконати перед собою поставлену мету, а саме знайти вільну вакансію, яка найбільше підходила йому, або надати перелік робітників на вільне вакантне місце для роботодавця.

Тобто інформаційна система призначена для реєстрації і зберігання відомостей про кандидатів на роботу. Користувач може помістити в неї деталізовану анкету з повною інформацією про кандидата, або просто зареєструвати резюме, написати свої додаткові відомості. Кожному новому кандидату автоматично присвоюється ідентифікаційний номер і одночасно перевіряється, чи не звертався він раніше. Так само роботодавець може заповнити спеціальну форму, в якій він зауважить вакантне місце, на яке йому потрібен працівник, загальну кількість вакансій, розмір заробітної плати і адресу підприємства.

Об'єктом, на який спрямована система автоматизації є користувач, при чому необхідно поєднати взаємодію безробітного з роботодавцем, з найменшим втручанням робітника центру зайнятості, який відповідає за взаємодію обох сторін.

Вхідна інформація необхідна для створення точного опису сторони, яка її подає, для того щоб в подальшому в обох учасників не було ніяких суперечок. Також вхідна інформація безробітного має бути максимально повною тому, що центр зайнятості має велику базу карток клієнтів, які повинні бути унікальними і не повторюватись. В даній створюваної

інформаційної системи точність і правдивість інформації дуже важлива, адже учасниками з обох сторін є люди, перша особа відіграє роль фізичної особи, і особа яка виступає в ролі юридичної особи, і між ними створюється тісний зв'язок, який базується на попиті і пропозиції.

Уявляючи кінцевий веб-додаток, ми вже можемо поставити певні цілі. Для того щоб інформаційна система була розроблена з гарним кінцевим результатом, необхідно перш за все поставити певні вимоги, які на протязі усієї розробки будемо дотримуватися і намагатися виконати їх.

Також можна виділити певні вимоги до кінцевого продукта. Ці вимоги забезпечать якість програмного засобу. До них віднесемо:

– програмний продукт повинен виконувати функцію обслуговування клієнтів (роботодавця, людини що шукає роботу);

– створений сайт має бути орієнтованим на користувача і задовольняти його потреби в цій сфері;

– програмний продукт має бути зрозумілим для кожного користувача системи;

– програмний продукт повинен мати економічний ефект від впровадження.

Необхідно обрати певний підхід до нашої класифікації вимог автоматизованої системи. В даному випадку нам більше підходить стандарт IEEE 830-1998 щодо SRS (Software RequirementsSpecifications) – дана методика описує рекомендовані принципи складання специфікації вимог до програмного забезпечення. Вона заснована на моделі, в якій результат процесу специфікації програмного забезпечення є однозначним і повним специфікаційним документом. Цей стандарт передбачає аналіз таких положень:

– функціональність;

– зовнішній інтерфейс (інтерфейсу користувача, програмного інтерфейсу, інтерфейсу обладнання тощо);

– продуктивність;

– атрибути (точність, здатність бути модифікованою, експлуатаційні якості системи тощо);

– обмеження проектування (використання певних мов програмування, СУБД, форматів обміну інформацією тощо).

Отже, в роботі є вже певні умови яких необхідно дотримуватись і слідкувати за їх виконанням.

РОЗДІЛ 2 ПРОЕКТУВАННЯ СИСТЕМИ АВТОМАТИЗАЦІЇ СУМСЬКОГО МІСЬКОГО ЦЕНТРУ ЗАЙНЯТОСТІ

2.1 Моделі бізнес-процесів задачі

Основна мета професійної орієнтації служби зайнятості – сприяти громадянам, які звертаються в службу зайнятості, в отриманні підходящої роботи відповідно до їх особистих інтересів, потреб роботодавців і ринку праці шляхом їх професійного інформування та консультування.

Система професійної орієнтації в державній службі зайнятості надає наступні послуги:

– інформування та консультування громадян, які звертаються в службу зайнятості з метою вибору роботи, режиму праці;

– професійна орієнтація безробітних громадян.

Основна проблема в роботі фірм – відсутність взаєморозуміння з клієнтами. Замовляючи працівника, клієнти зазвичай самі не знають, чого саме вони хочуть. Нерідкі випадки, коли потенційні роботодавці просять фірму підібрати фахівця, але при цьому назва посади, функціональні обов'язки майбутнього працівника і запропоновану йому заробітна плата між собою ніяк не корелюють. Багато в чому це пов'язано з тим, що в більшості фірм обов'язки співробітників сформульовані погано і без урахування їх кваліфікації.

Подавши заяву в систему, безробітний або роботодавець стає її клієнтом і починає обслуговуватися протягом терміну обслуговування заявки. Термін обслуговування заявки розглядається кілька місяців. Якщо за цей час заявка(анкета) не виконуються то вона повертається.

Основним призначенням системи, що розробляється, є автоматизація занесення та зберігання даних по неактивному населенню і роботодавцям.

Система дозволяє змінювати, доповнювати, вести пошук і перегляд інформації про працевлаштовуючих громадян і роботодавців.

Для проведення аналізу і реорганізації бізнес-процесів використовують CASE-засоби — набір інструментів і методів програмної інженерії для проектування програмного забезпечення, що допомагає забезпечити високу якість програм, відсутність помилок і простоту в обслуговуванні програмних продуктів. До CASE програм відносяться – Design/IDEF (Meta Software), BPwin та ERwin (Logic Works), Designer/2000 (ORACLE), CASE.Аналитик (МакроПроджект), Microsoft Project (Microsoft) [29].

Побудова моделі інформаційної системи починається з опису функціонування підприємства (системи) в цілому у вигляді контекстної діаграми. На рисунку 2.1 представлена контекстна діаграма інформаційної системи "Центра зайнятості" створена за допомогою методології IDEF0, що являє собою побудову ієрархічної системи діаграм – одиничних описів фрагментів системи [2].

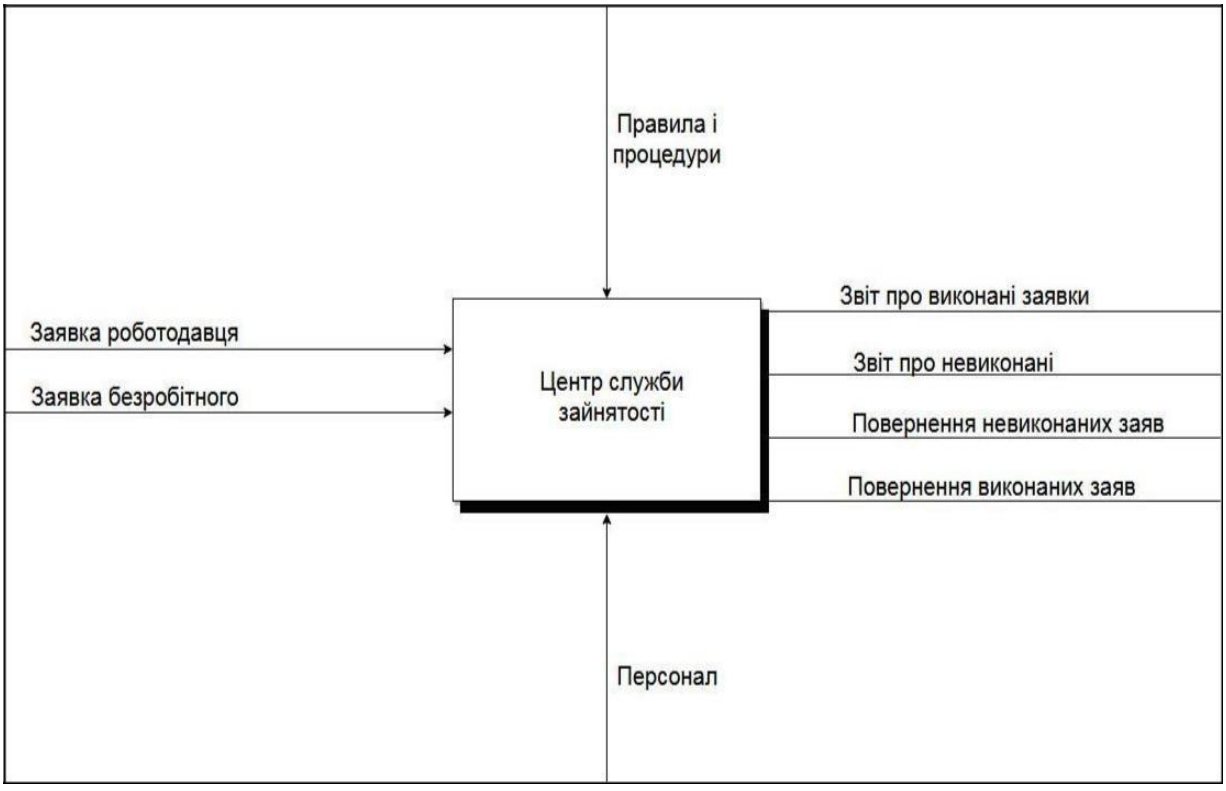


Рисунок 2.1 – Контекстна діаграма "Центр зайнятості"

Взаємодія системи з навколишнім середовищем описується в термінах входу (на рис 2.1 – "Заявка роботодавця" і "Заявка безробітного"), виходу ("Звіт про виконані заявки", "Звіт про невиконаних заявки", "Повернення невиконаних заявок" і "Повернення виконаних заявок"), управління ("Правила і процедури" – правила, якими керується процес функціонування служби зайнятості) і механізмів ("Персонал" – це ресурси, необхідні для процесу функціонування служби зайнятості).

Загальні стрілки і діаграми, опишемо за допомогою таблиця 2.1:

Таблиця 2.1 – Опис елементів контекстної діаграми "Центр зайнятості"

Назва стрілки/діаграми	Функція	Короткий опис
Центр зайнятості	Робота	Служба займається наданням інформації про робочі місця і про працевлаштовуємо
Заявка роботодавця	Вхідна	Анкета, що заповнюється роботодавцем при зверненні до служби зайнятості
Заявка безробітного	Вхідна	Анкета, що заповнюється працевлаштовуємо громадянином при зверненні в службу зайнятості
Повернення невиконаних заявок	Вихідна	Повідомлення працевлаштовується громадянина або роботодавця про припинення розгляду його заявки
Повернення виконаних заявок	Вихідна	Повідомлення працевлаштовується громадянина або роботодавця про виконання його заявки
Персонал	Управління	Співробітники "Центру зайнятості"
Правила та процедури	Механізм	Правила і процедури, якими керується компанія при роботі

Після опису контекстної діаграми проводиться функціональна декомпозиція – система розбивається на підсистеми і кожна підсистема описується окремо. Потім кожна підсистема розбивається на більш дрібні і так далі до досягнення потрібного ступеня деталізації. В результаті такого розбиття, кожен фрагмент системи зображується на окремій діаграмі декомпозиції [40].

Після подальшого розбиття діаграми отримуємо три діаграми декомпозиції, що описують кожна одну з робіт, представлених на діаграмі верхнього рівня (рис. 2.2).

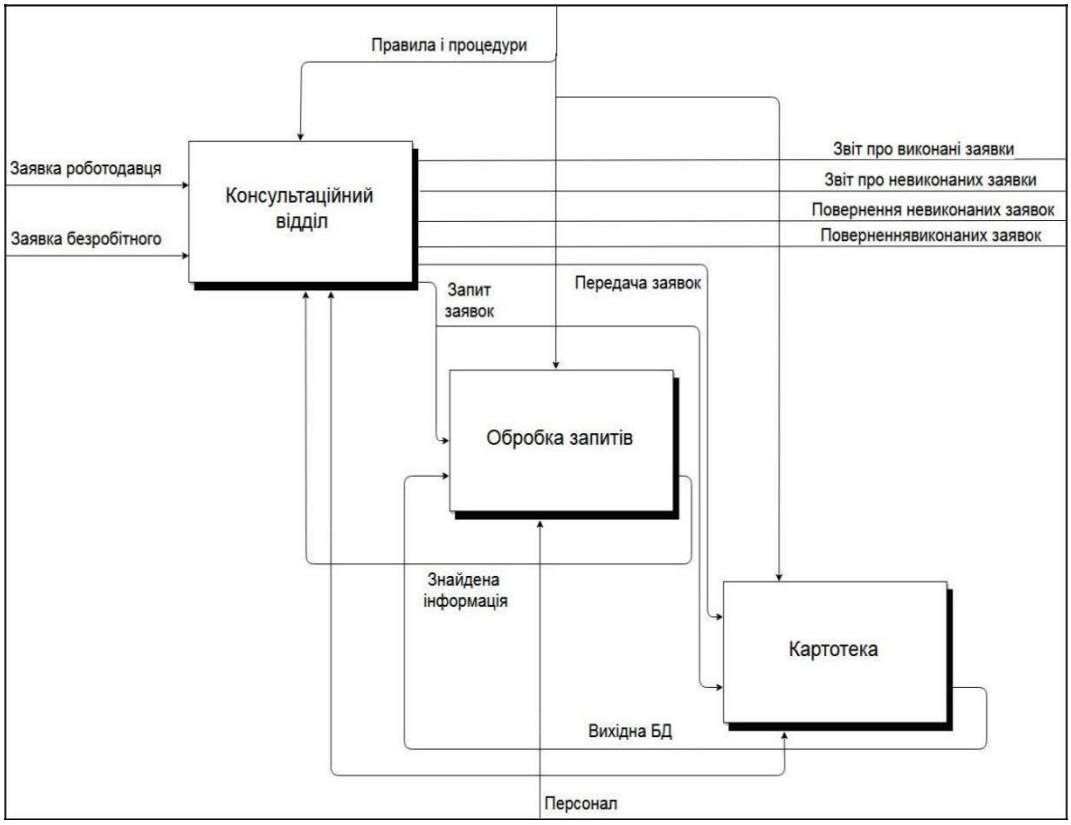


Рисунок 2.2 – Декомпозиція роботи центру зайнятості

Весь процес функціонування служби зайнятості розбивається на три діаграми:

1. "Консультаційний відділ" – займається консультацією, одержанням заявок, формуванням звітів;

2. "Обробка запиту" – являє собою процес пошуку інформації за заявками;

3. "Картотека" – зберігання заявок.

Загальні елементи, що перейшли з діаграми верхнього рівня і діаграми, наведено в таблиці 2.2.

Таблиця 2.2 – Опис значень в декомпозиції центру зайнятості

Назва стрілки/діаграми	Функція	Короткий опис функції
Запит заявок	Вхідна	Запит на вакансії або працівників
Передача заявок	Вхідна	Передача заявок пошуку
Вихідна БД	Вхідна	Вихідна база даних
Знайдена інформація	Управління	Знайдена інформація за заявкою

Далі уточнюємо діаграму "Консультаційний відділ", де отримуємо дві діаграми декомпозиції, що описують кожна одну з робіт, представлених на діаграмі верхнього рівня (рис. 2.3).

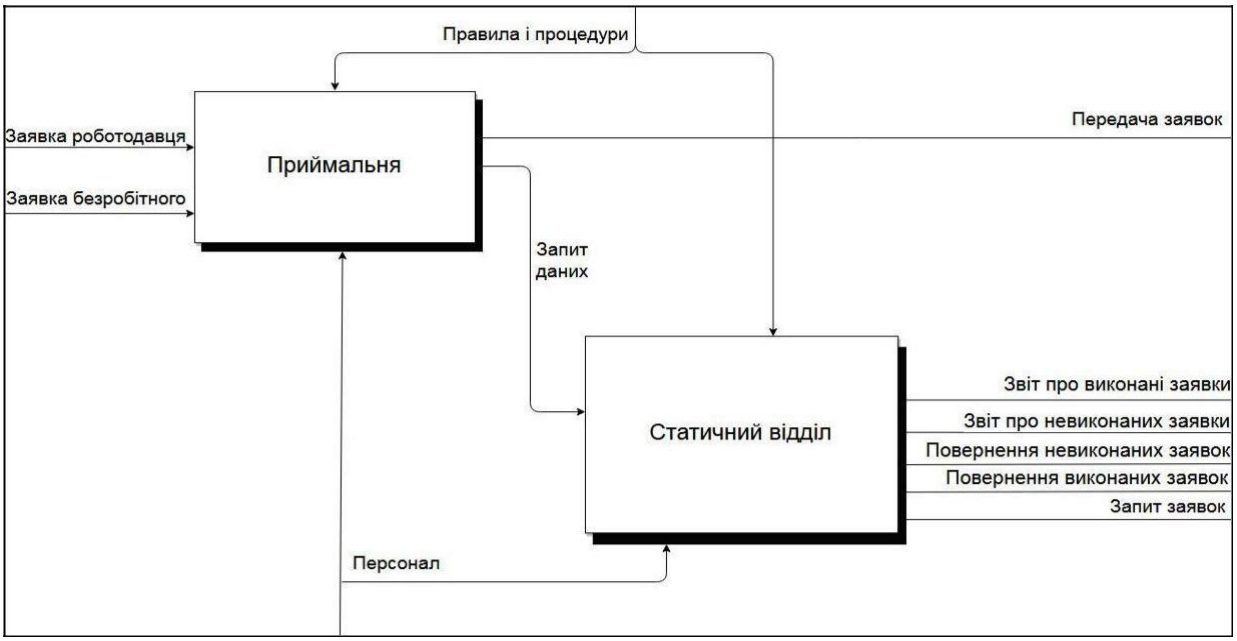


Рисунок 2.3 – Декомпозиція роботи "Консультаційний відділ"

Загальні елементи, що перейшли з діаграми верхнього рівня і діаграми, опишемо в таблиці 2.3:

Таблиця 2.3 – Опис значень декомпозиції "Консультаційний відділ"

Назва стрілки/діаграми	Функція	Короткий опис функції
Приймальня	Робота	Консультації та прийом заявок
Статичний відділ	Робота	Перевірка заявок і формування звітів
Запит даних	Вхідна	Запит інформації

Розглянемо більш детально діаграму "Обробка запиту". Вона поділяється на "Відкриття бази даних" та "Виконання запиту" (детальний опис декомпозиції "Обробка запиту" наведено на рис. 2.4).

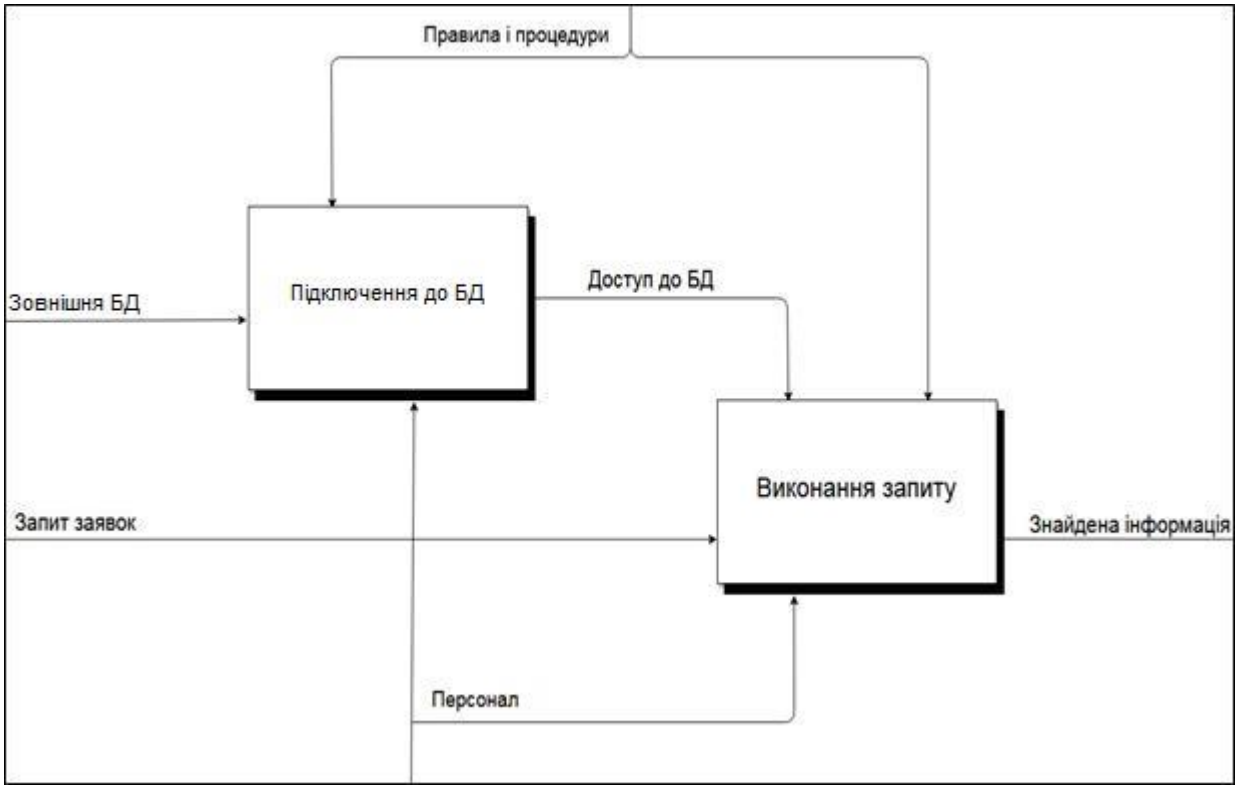


Рисунок 2.4 – Декомпозиція роботи "Обробка запиту"

Опишемо параметри елементів діаграми верхнього рівня в таблиці 2.4:

Таблиця 2.4 – Опис значень декомпозиції "Обробка запиту"

Назва стрілки/діаграми	Функція	Короткий опис функції
Підключення до БД	Робота	Відкриття бази даних для пошуку і зміни
Виконання запиту	Робота	Пошук і зміна бази даних
Доступ до БД	Механізм	Запит на доступ до бази даних

Якщо в процесі моделювання потрібно висвітлити специфічні сторони технології підприємства знаходиться необхідність у створенні змішаної моделі. Діаграми потоків даних (Data Flow Diagramming) використовуються для опису документообігу та обробки інформації. Нотація діаграми потоків даних DFD включає такі поняття, як "зовнішнє посилання" і "сховище

даних", що робить її більш зручною (в порівнянні з IDEF0) для моделювання документообігу. На рис. 2.5 представлена "Декомпозиція в нотації DFD "Виконання запиту"", що описує діяльність з пошуку інформації в базі даних [2].

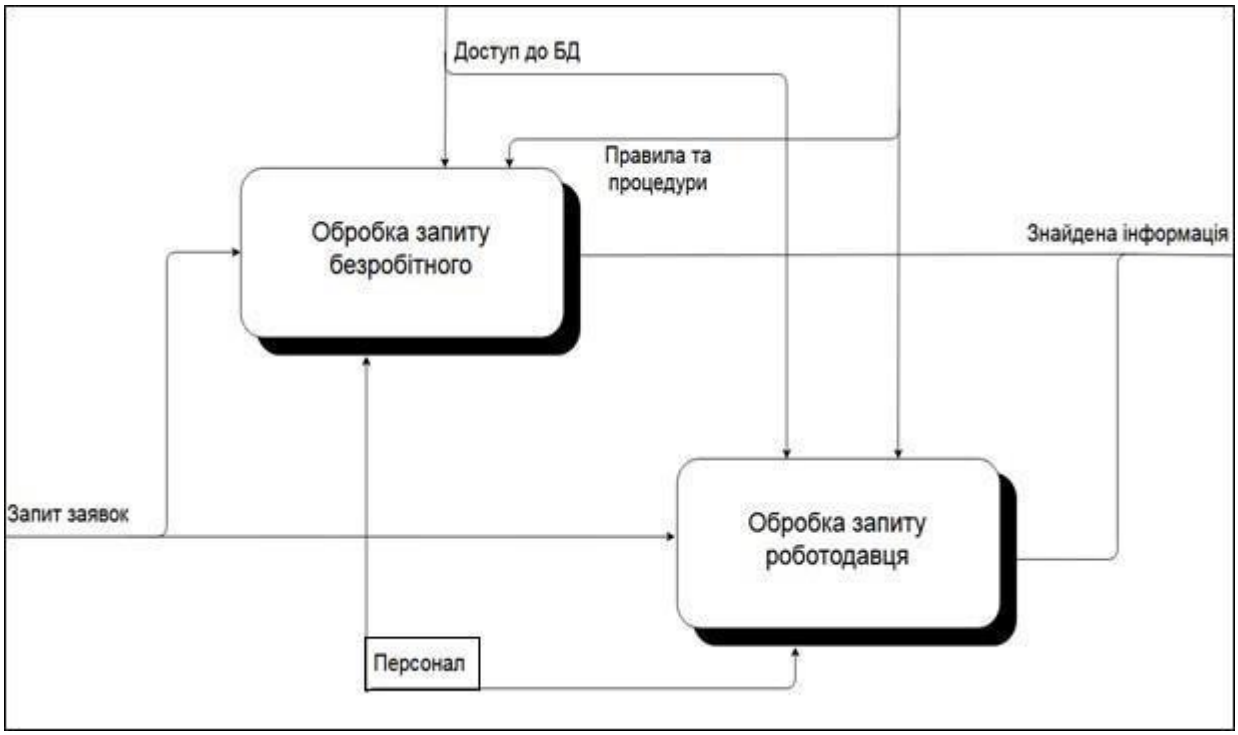


Рисунок 2.5 – Декомпозиція в нотації DFD "Виконання запиту"

Всі роботи, представлені на діаграмі виконуються "Персоналом" у відповідність з переліком обов'язків.

Деталізуємо діаграму "Картотека" за допомогою нотації DFD (рис. 2.6).

Електронний каталог — каталог на електронному носії, який подає переважно зміст паперового каталогу та містить інформацію про продукти й послуги для клієнтів або ділових партнерів.

Перевагами "е-каталогу" являється скорочення витрат на придбання і постачання в організаціях; являє собою складову частину електронної торговельної системи. Від рівня розвитку технології залежить форма каталогу, процеси його створення, підтримки розвитку та використання, що включають в себе правила формування записів і організації пошуку.

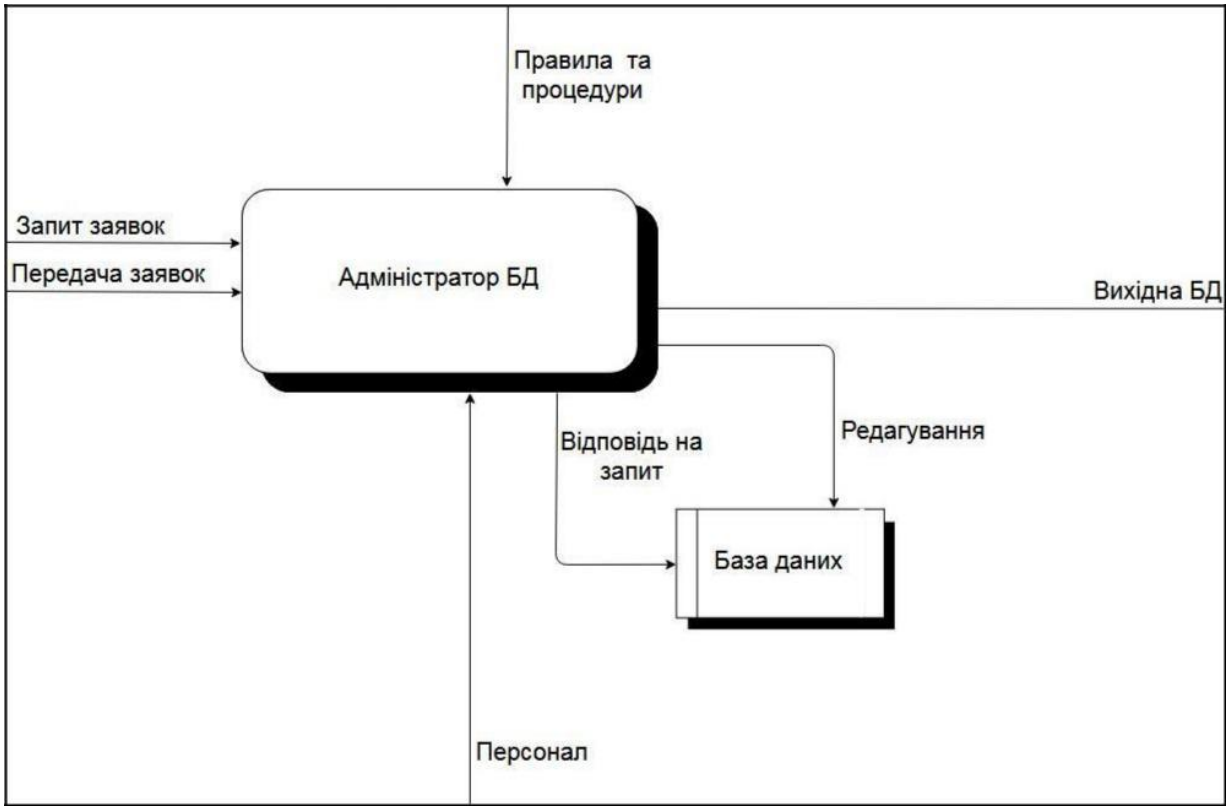


Рисунок 2.5 – Декомпозиція в нотації DFD "Картотека"

Наведена нижче таблиця 2.5 містить загальні елементи, що перейшли з діаграми верхнього рівня [2].

Таблиця 2.5 – Опис значень декомпозиції в нотації DFD "Картотека"

Назва стрілок діаграм	Короткий опис функції
Адміністратор БД	Редагування і перегляд бази даних
Відповідь на запит	Надання наявною інформацією на заявку
Редагування	Редагування бази даних
База даних	Збереження даних

Діаграма дерева вузлів показує ієрархію робіт в моделі та дозволяє розглянути всю модель цілком, але не показує взаємозв'язку між роботами. Підсумуємо розташування робіт у дереві вузлів (рис 2.6).

Рисунок 2.6 – Діаграма дерева вузлів

Отже, при проведенні аналізу бізнес-процесів центру зайнятості, сформовано дерево рішень, яке складається с 4-х рівнів:

– діаграма "Центр служби зайнятості" – 1-ий рівень дерева вузлів;

– діаграми "Консультаційний відділ", "Обробка запиту" і "Картотека" – 2-ий рівень дерева вузлів;

– діаграми "Приймальна", "Статичний відділ", "Відкриття БД", "Виконання запиту" і "Адміністратор БД" – 3-й рівень;

– діаграми "Обробка запиту безробітного" і "Обробка запиту роботодавця" – 4-ий рівень дерева вузлів.

2.2 Інформаційна архітектура та технології вирішення задачі

Для створення веб-сторінки Сумського міського центру зайнятості обрано середовище розробки Visual Studio, а саме технологію ASP.NET. Дана технологія створення веб-додатків і веб-сервісів розроблена компанією Microsoft і є частиною платформи Microsoft.NET. ASP.NET зовні багато в чому зберігає схожість із старішою технологією ASP, що дозволяє розробникам відносно легко перейти на ASP.NET. У той же час внутрішній устрій ASP.NET істотно відрізняється від ASP, оскільки вона заснована на

платформі.NET і, отже, використовує всі нові можливості, що надаються цією платформою. ASP.NET ґрунтується на методиці Common Language Runtime (CLR), який є основою всіх додатків Microsoft. NET. Розробники можуть писати код для ASP.NET, використовуючи практично будь-які мови програмування, що входять у комплект .NET Framework (C#, Visual Basic.NET, і JScript.NET). Технологія ASP.NET є швидшою у порівнянні зі скриптовими технологіями, це пояснюється тим, що при зверненні код компілюється і поміщається в спеціальний кеш, і вже потім виконується, не вимагаючи витрат часу на парсинг і оптимізацію [10].

Якщо порівнювати ASP та ASP.NET, ASP — похідна від Win32, XML і HTML; PHP — від XML, HTML, Java і CDI, тоді ASP.NET — від HTML і.NET (XML і XAML відповідно). ASP.NET — більш функціональне середовище для розробки та розгортання веб-ресурсів. У ASP.NET можна працювати з будь-якою .NET мовою, аж до Managed C++ і Visual Basic, що дозволяє не замислюватися про перехід на C# [11].

На сьогоднішній день існує файл-серверна, клієнт-серверна та веб-серверна інформаційна архітектури.

Архітектура клієнт-сервер є одним із архітектурних шаблонів програмного забезпечення та є домінуючою концепцією у створенні розподілених мережних програм і передбачає взаємодію та обмін даними між ними. Вона передбачає такі основні компоненти [17]:

– набір серверів, які надають інформацію або інші послуги програмам, які звертаються до них;

– набір клієнтів, які використовують сервіси, що надаються серверами;

– мережа, яка забезпечує взаємодію між клієнтами та серверами.

Сервери є незалежними один від одного. Клієнти також функціонують паралельно і незалежно один від одного. Немає жорсткої прив'язки клієнтів до серверів. Більш ніж типовою є ситуація, коли один сервер одночасно обробляє запити від різних клієнтів; з іншого боку, клієнт може звертатися то

до одного сервера, то до іншого. Клієнти мають знати про доступні сервери, але можуть не мати жодного уявлення про існування інших клієнтів [1, 9].

Веб-сервер — це сервер, що приймає HTTP-запити від клієнтів, зазвичай веб-браузерів, видає їм HTTP-відповіді, зазвичай разом з HTML-сторінкою, зображенням, файлом, медіа-потоком або іншими даними. Веб-сервери лежать в основі Всесвітньої павутини.

Веб-сервером називають як програмне забезпечення, що виконує функції веб-сервера, так і комп'ютер, на якому це програмне забезпечення працює. Клієнти дістаються веб-сервера за URL-адресою потрібної їм веб-сторінки або іншого ресурсу.

Інформаційна архітектура "файл-сервер" – це виділений сервер, призначений для виконання файлових операцій введення-виведення і зберігає файли будь-якого типу. Як правило, має великий обсяг дискового простору, реалізованому в формі RAID-масиву для забезпечення безперебійної роботи і підвищеної швидкості запису і читання даних. RAID – технологія віртуалізації даних, яка об'єднує кілька дисків в логічний елемент для надмірності і підвищення продуктивності. Абревіатура "RAID" спочатку розшифровувалась як "Redundant Array of Inexpensive Disks" що в перекладі означає – надмірний масив недорогих дисків [9].

При створенні веб-орієнтованої інформаційної системи центру зайнятості будемо спиратися на клієнт-серверну архітектуру. Так як клієнт повинен спочатку зробити запит на подання чи перегляд інформації і вже потім база даних виводить необхідний по запиту результат.

Переваги цієї архітектури:

– архітектура "клієнт-сервер" уможливлює, в більшості випадків, розподіл функцій обчислювальної системи між декількома незалежними комп'ютерами в мережі. Це дозволяє спростити обслуговування обчислювальної системи. Зокрема, заміна, ремонт, модернізація або переміщення сервера не зачіпають клієнтів;

– всі дані зберігаються на сервері, який, як правило, захищений набагато краще за більшість клієнтів. На сервері простіше забезпечити контроль повноважень, щоб вирішувати доступ до даних тільки клієнтам з відповідними правами доступу;

– дозволяє об'єднати різні клієнти. Використовувати ресурси одного сервера часто можуть клієнти з різними апаратними платформами, операційними системами.

До основних недоліків клієнт-серверної архітектури відносяться:

– непрацездатність сервера може зробити непрацездатною всю обчислювальну мережу;

– підтримка роботи даної системи вимагає окремого фахівця – системного адміністратора;

– висока вартість обладнання.

Архітектура "клієнт-сервер" визначає загальні принципи організації взаємодії в мережі, де є сервери, вузли-постачальники деяких специфічних функцій (сервісів) і клієнти (споживачі цих функцій) [1].

Практичні реалізації такої архітектури називаються клієнт-серверними технологіями. Кожна технологія визначає власні або використовує наявні правила взаємодії між клієнтом і сервером, які називаються протоколом обміну (протоколом взаємодії).

Архітектура "клієнт-сервер" поділяться на два типи: дворівнева та трирівнева.

У будь-якій мережі (навіть тимчасової), побудованої на сучасних мережевих технологіях, присутні елементи клієнт-серверного взаємодії, найчастіше на основі дворівневої архітектури.

Дворівнева архітектура (two-tier, 2-tier) отримала таку назву через необхідність розподілу трьох базових компонентів між двома вузлами (клієнтом і сервером).

Дворівнева архітектура використовується в клієнт-серверних системах, де сервер відповідає на клієнтські запити безпосередньо і в повному обсязі,

при цьому використовуючи тільки власні ресурси. Тобто сервер не викликає сторонні мережеві програми, але не звертається до сторонніх ресурсів для виконання будь-якої частини запиту (рис. 2.7).

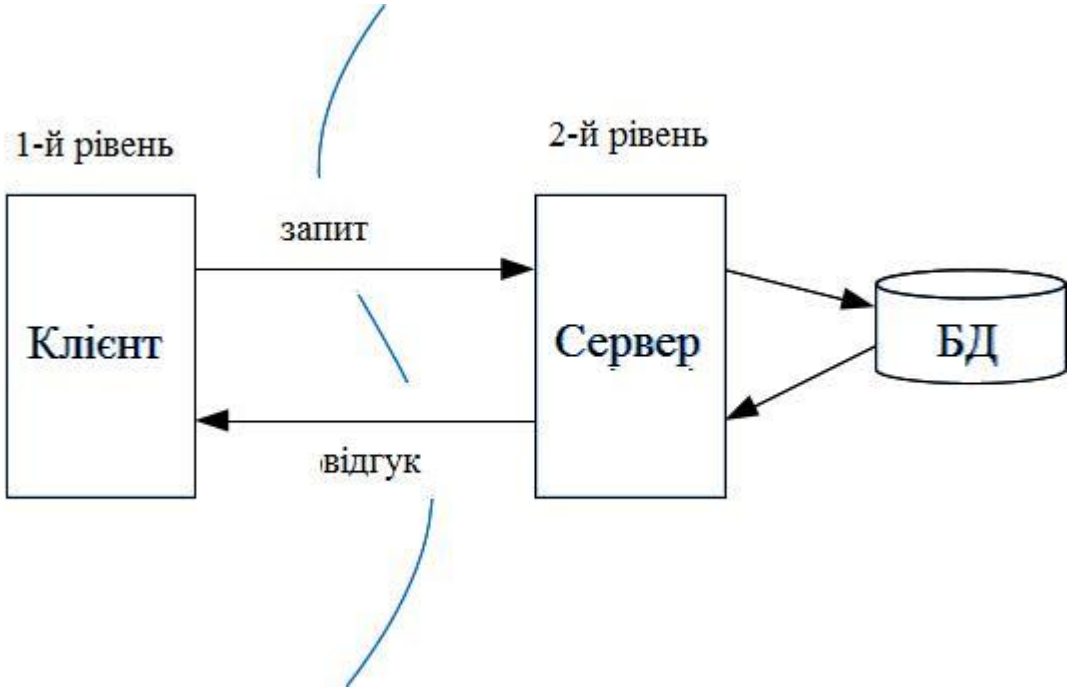


Рисунок 2.7 – Дворівнева клієнт-серверна архітектура

Розташування компонентів на стороні клієнта або сервера визначає наступні основні моделі їх взаємодії в рамках дворівневої архітектури:

– сервер терміналів – розподілене представлення даних;

– файл-сервер – доступ до віддаленої базі даних і файлових ресурсів;

– сервер БД – віддалене уявлення даних;

– сервер додатків – віддалений додаток.

Ще одна тенденція в клієнт-серверних технологіях – трирівнева архітектура. Пов'язана вона з дедалі більшим використанням розподілених обчислень. Вони реалізуються на основі моделі сервера додатків, де мережевий додаток розділене на дві і більше частин, кожна з яких може виконуватися на окремому комп'ютері. Виділені частини додатка взаємодіють один з одним, обмінюючись повідомленнями в заздалегідь

узгодженому форматі. В цьому випадку дворівнева клієнт-серверна архітектура стає трирівневою (three-tier, 3-tier)(рис. 2.8).

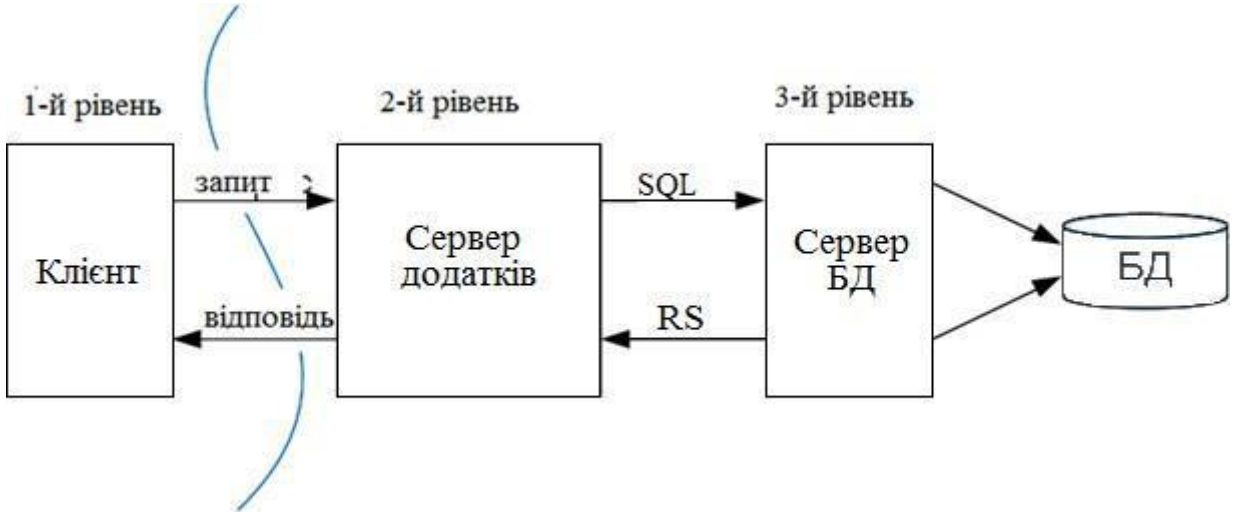


Рисунок 2.8 – Трирівнева клієнт-серверна архітектура

Як правило, третьою ланкою в трирівневої архітектурі стає сервер додатків, тобто компоненти розподіляються наступним чином (рис. 2.8):

– представлення даних – на стороні клієнта;

– прикладний компонент – на виділеному сервері додатків (як варіант, що виконує функції проміжного ПО);

– управління ресурсами – на сервері БД, який і представляє запитувані дані [1].

Якщо порівняти ці дві архітектури то підсумовується, що дворівнева архітектура простіше, так як всі запити обслуговуються одним сервером, але саме через це вона менш надійна і висуває підвищені вимоги до продуктивності сервера. Трирівнева архітектура складніше, але завдяки тому, що функції розподілені між серверами другого і третього рівня, ця архітектура надає:

– високий ступінь гнучкості і масштабованості;

– високу безпеку (тому що захист можна визначити для кожного сервісу або рівня);

– високу продуктивність (тому що завдання розподілені між серверами).

Тому для створення веб-сайту використаємо трирівневу клієнт-серверну архітектуру.

2.3 Функціональна структура задачі

У будь-яких управлінських інформаційних системах організації є багато прикладних програм, які використовують одні й ті самі робочі дані і відбувається дублювання робіт у процесі збирання, зберігання і пошуку цих даних. Зі збільшенням кількості прикладних програм, що обслуговують всі рівні управління та обробляють одні й ті самі робочі дані, зростає обсяг дублювання, що стає гальмом на шляху комп'ютеризації управління. Більш того, це дублювання часто неефективне, оскільки призводить до несумісності прикладних програм. Виходом із цієї ситуації стала концепція створення єдиної централізовано керованої бази даних. Системи управління базами даних (СУБД) обслуговують всі прикладні програми організацій. СУБД має на меті об'єктивного, незалежного від окремих прикладних програм, спростити колективне використання даних різними прикладними програмами.

В центрі зайнятості для кожної людини, що звернулась за пошуком роботи враховується інформація про наявність дипломів про освіту, володіння мовами, інші відомості, зокрема наявність посвідчення водія, наявність закордонного паспорту.

Служба зайнятості веде облік заявок підприємств на фахівців. Для обліку заявки кожного підприємства враховується: підприємство, адреса, номер телефону, контактна особа, вакантна посада, зарплата, данні про фахівця (професія, вік, стать, стаж роботи за фахом, оплата, інші відомості). В центрі зайнятості ведеться облік укладених договорів з працевлаштування. При цьому враховується: заявка від підприємства, заявка від людини що

шукала роботу, працівник центру зайнятості, за сприянням якого був укладений договір. Розроблена багаторівнева інформаційна система полегшить і зробить більш ефективним процес працевлаштування фахівців.

Розроблена модель інтерфейсу веб-сторінок будуть формувати інтерфейс користувачів інформаційної системи. Використовуючи інструментарій середовища розробки Visual Studio створимо екземпляр веб-сайту. Використовуючи технологію майстер-сторінок розробимо загальні макети сторінок веб-додатку [41].

Для створення веб-інтерфейсів системи будуть використані серверні елементи управління HTML, серверні веб-елементи управління, модель зворотної передачі ASP.NET. Також реалізуємо інтерфейс для доступу до об'єктів бази даних.

Структурна схема створюваної веб-орієнтованої інформаційної системи центру зайнятості виглядає наступним чином (рис. 2.9).

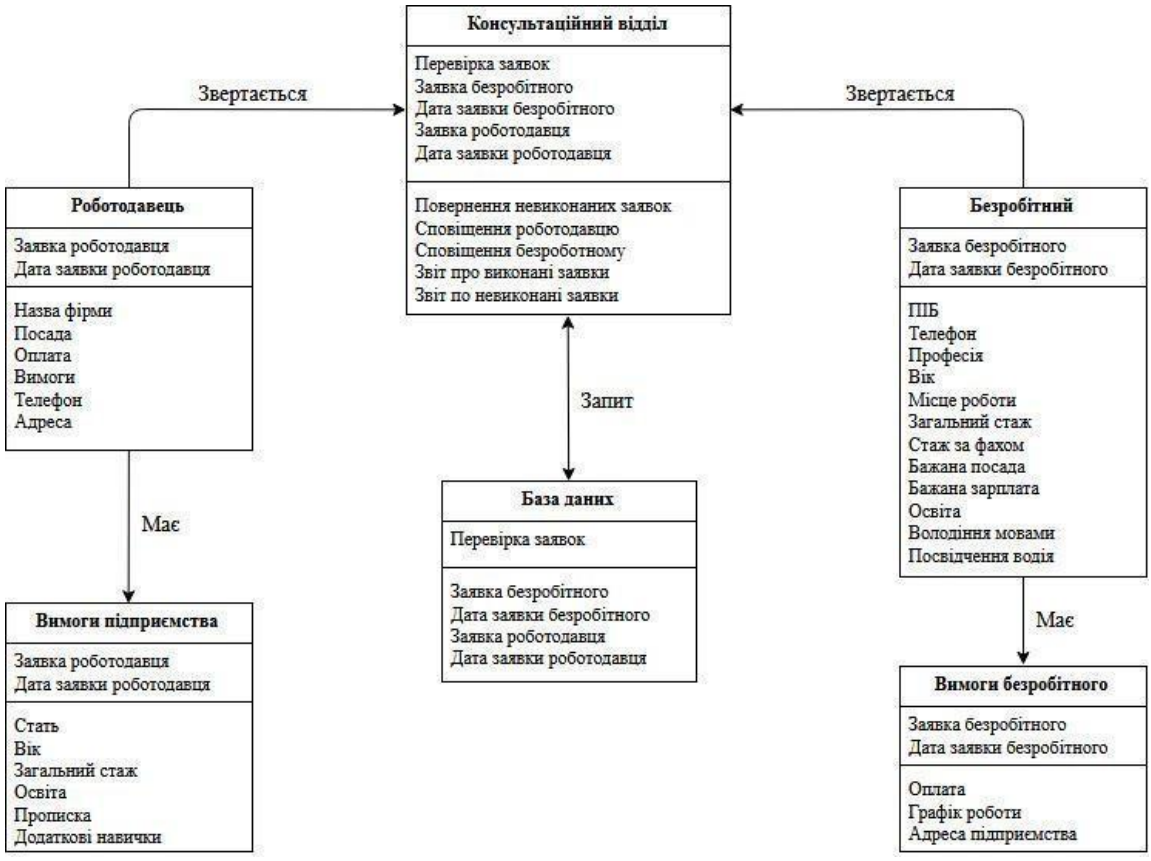


Рисунок 2.9 – Схема інформаційної системи

Схема інформаційної системи представлена у вигляді сутностей, їх атрибутів і зв'язків між ними [12]. Кожна сутність представляє безліч подібних об'єктів. Зв'язок на схемі відображає логічну залежність однієї сутності від іншої. На нашій схемі залежними сутностями є: "Консультаційний відділ". Батьківськими для неї є суті "Роботодавець" і "Безробітний" відповідно.

2.4 Підсистеми забезпечення функціональної частини

2.4.1 Технічне забезпечення

Технічне забезпечення – це комплекс технічних засобів, призначених для роботи інформаційної системи, а також відповідна документація на ці засоби і технологічні процеси. В свою чергу технічні засоби автоматизації і управління є апаратними, програмними і конструктивними засобами і комплексами засобів, орієнтованими на вирішення як типових, так і конкретних завдань з автоматизації технологічних процесів. Компоновка технічних засобів автоматизації та керування в агрегатні комплекси технічних засобів виконується згідно міжнародної стандартизації. Від застосованих технічних засобів значною мірою залежить підвищення техніко-економічних показників систем управління технічними процесами і виробництвом в цілому. До цих показників відносять:

– якість управління;

– надійність;

– витрати на проектування;

– безпеку;

– експлуатації;

– можливість адаптації систем управління до властивостей об'єктів технічних процесів у разі зміни останніх.

Програмне забезпечення (ПЗ) складається із загального ПЗ (операційні системи, локальні і глобальні мережі і комплекси програм технічного обслуговування, спеціальні обчислювальні програми) і спеціального ПЗ (організуючі програми і програми, що реалізують алгоритми контролю та управління).

Персонал та інструктивно-методичні матеріали складають організаційне забезпечення системи.

Процедури і технології розробляються на основі логіко-математичних моделей і алгоритмів, що становлять основу математичного забезпечення системи, і реалізуються за допомогою ПЗ і КТС, а також інтерфейсу, що забезпечує доступ користувача до інформації.

Під технічним забезпеченням розуміють склад, форми і способи експлуатації різних технічних пристроїв, необхідних для виконання інформаційних процедур: збирання, реєстрації, передачі, зберігання, обробки та використання інформації. До елементів технічного забезпечення відносяться: комплекс технічних засобів, організаційні форми використання технічних засобів, персонал, який працює на технічних засобах, інструктивні матеріали щодо використання техніки.

Комплекс технічних засобів – це сукупність взаємопов'язаних технічних засобів, призначених для автоматизованої обробки даних.

Вимоги до комплексу технічних засобів:

– мінімізація витрат на придбання та експлуатацію;

– надійність;

– захист від несанкціонованого доступу;

– раціональний розподіл за рівнями обробки.

Комплекс технічних засобів становлять:

– комп'ютери будь-яких моделей;

– пристрої збору, накопичення, обробки, передачі й виводу інформації;

– пристрої передачі даних і ліній зв'язку;

– оргтехніка й пристрої автоматичного знімання інформації;

– експлуатаційні матеріали.

До теперішнього часу склалися дві основні форми організації технічного забезпечення (форми використання технічних засобів) – централізована і частково або повністю децентралізована. Централізоване технічне забезпечення базується на використанні в інформаційній системі великих комп'ютерів і обчислювальних центрів. Децентралізація технічних засобів передбачає реалізацію функціональних підсистем на персональних комп'ютерах безпосередньо на робочих місцях. Перспективним підходом слід вважати, мабуть, частково децентралізований підхід – організацію технічного забезпечення на базі розподілених мереж, що складаються з персональних і великих комп'ютерів для зберігання баз даних, загальних для будь-яких функціональних підсистем [6].

До складу програмного забезпечення входять загальносистемні та спеціальні програмні продукти, а також технічна документація (рис. 2.10).

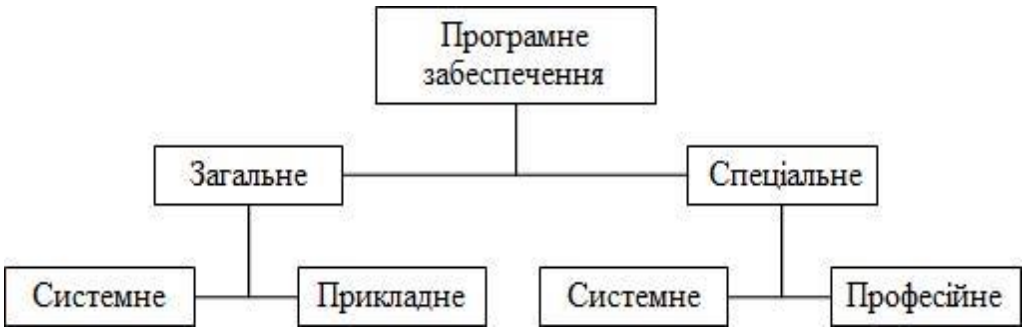


Рисунок 2.10 – Програмне забезпечення інформаційної системи

При створенні веб-сайту з допомогою середовища розробки Microsoft Visual Studio використовується технологія ASP.NET.

Вимоги до апаратного забезпечення наведені у таблиці 2.11. Загалом, вимоги щодо програмного та апаратного забезпечення не є дуже високими, що є перевагою даного програмного продукту перед можливими майбутніми аналогами [34].

Таблиця 2.16 – Системні вимоги для коректної роботи із середовищем розробки ASP.NET

Назва	Параметри
Процесор	1,6 ГГц чи вище
ОЗУ – RAM	1 ГБ
Дисковий простір	10 ГБ вільного простору на жорсткому дисці
Жорсткий диск	5400 RPM(об/хв)
Монітор	1024 x 768 чи вище
Операційна система	Windows 8; Windows 7 SP 1; Windows Server 2012; Windows Server 2008 R2 SP1;

2.4.2 Програмне та апаратне забезпечення

Програмне забезпечення – комп'ютерні програми, процедури і, можливо, відповідна документація і дані, що відносяться до функціонування комп'ютерної системи (IEEE Std 829-2008).

Програмне забезпечення є одним з видів забезпечення обчислювальної системи, поряд з технічним (апаратним), математичним, інформаційним, лінгвістичним, організаційним і методичним забезпеченням.

Програмне забезпечення – це те, що робить комп'ютери універсальними, дозволяючи використовувати типову обчислювальну машину для вирішення найрізноманітніших завдань.

У комп'ютерному сленгу часто використовується слово "софт", що походить від англійського слова "software", яке в цьому сенсі вперше застосував в статті журналу "American Mathematical Monthly" математик з Пристанського університету Джон Тьюкі в 1958 році.

Розрізняють системне програмне забезпечення (зокрема, операційна система, транслятори, редактори, графічний інтерфейс користувача); прикладне програмне забезпечення, що використовується для виконання конкретних завдань, наприклад, статистичне програмне забезпечення;

інструментальне програмне забезпечення (комп'ютерні програми, призначені для проектування, розробки, адміністрування і супроводження системного та прикладного програмного забезпечення).

Виконання програмного забезпечення комп'ютером полягає у маніпулюванні інформацією та керуванні апаратними компонентами комп'ютера. Наприклад, типовим для персональних комп'ютерів є відтворення інформації на екран та отримання її з клавіатури.

Програмне забезпечення (software) та апаратне забезпечення (hardware) — це два комплементарні компоненти комп'ютера, причому межа між ними нечітка: деякі фрагменти програмного забезпечення на практиці реалізуються суто апаратурою мікросхем комп'ютера, а програмне забезпечення, в свою чергу, здатне виконувати функції електронної апаратури. По суті, призначення програмного забезпечення полягає в керуванні як самим комп'ютером так і іншими програмами та маніпулюванні інформацією.

Для реалізації веб-орієнтованої інформаційної системи центру зайнятості обрано систему Microsoft Visual Studio 2008, програмне забезпечення ASP.NET, які використовують мову програмування C# [27].

Visual Studio – це набір інструментів розробки програмних додатків, які базуються на використанні компонентів та інших технологій для створення потужних реалізацій. Крім того, середовище Visual Studio оптимізоване для спільного проектування, розробки та розгортання корпоративних рішень. В основі середовища Visual Studio покладено мову програмування C#, яка є однією з найпопулярніших на сьогодні. C# займає деяку проміжну позицію: із стандарту мови прибрані найбільш неприємні і неоднозначні особливості C++, але в той же час мова зберегла могутні виразні можливості, властиві для таких мов, як C++, Java або VB. Головною особливістю мови C# є її орієнтованість на платформу Microsoft .NET. Мова програмування C# розроблена таким чином аби приховати від розробника якомога більше незначних технічних деталей, включаючи операції по упаковці розпаковуванню типів, ініціалізації змінних і збірці сміття. Завдяки цьому

програміст, що пише на C#, може краще концентруватися на змістовній частині завдання. Таким чином, C# є новою мовою програмування, орієнтованою на розробку для платформи .NET і придатний як для створення малих так і великомасштабних програмних рішень [27].

РОЗДІЛ 3 РЕАЛІЗАЦІЯ ПРОТОТИПУ АВТОМАТИЗОВАНОЇ СИСТЕМИ ЕКОНОМІЧНОГО ОБ'ЄКТА

3.1 Структура та особливості реалізації інформаційного забезпечення

Безумовно одною з найважливіших частин розробки веб-сайту є створення інтерфейсу його сторінок. Веб-інтерфейс – це сукупність засобів, за допомогою яких користувач взаємодіє з веб-сайтом або веб-застосунком через браузер. Веб-інтерфейси отримали широке поширення у зв'язку із зростанням популярності всесвітньої павутини і відповідно повсюдного розповсюдження веб-браузерів.

Одним з основних вимог до веб-інтерфейсів є їхній однаковий зовнішній вигляд і однакова функціональність при роботі в різних браузерах.

Класичним і найпопулярнішим методом створення веб-інтерфейсів є використання HTML із застосуванням CSS і JavaScript, як правило за допомогою скриптових мов на стороні сервера. Проте різна реалізація HTML, CSS, DOM і інших специфікацій в браузерах викликає проблеми при розробці веб-застосунків і їхньої подальшої підтримки. Крім того, можливість користувача настроювати багато параметрів браузера (наприклад, розмір шрифту, кольору, відключення підтримки сценаріїв) може перешкоджати коректній роботі інтерфейсу [11].

Інший (менш універсальний) підхід полягає у використанні Adobe Flash, Silverlight або Java-аплетів для повної або часткової реалізації користувацького інтерфейсу. Оскільки більшість браузерів підтримує ці технології (як правило, за допомогою плагінів), Flash- або Java-застосунки можуть легко виконуватися. Вони здатні обходити багато несумісності в конфігураціях браузерів, бо надають програмісту більший контроль над інтерфейсом, хоча несумісність між Java або Flash реалізаціями на стороні клієнта може призводити до різних ускладнень.

З розвитком DHTML та JavaScript набув популярності підхід до розробки інтерфейсної частини веб-застосунків, названий AJAX. Серцем технології є здатність веб-сторінки зініціювати запит до веб-сервера і отримати потрібні дані, так щоб інтерфейс не перезавантажував сторінку цілком, а лише довантажують необхідні дані і змінив потрібні частини сторінки, що робить їх більш інтерактивними і продуктивними [31].

Веб-інтерфейси зручні тим, що дають можливість вести спільну роботу співробітникам, які не перебувають в одному офісі (наприклад, веб-інтерфейси часто використовуються для заповнення різних баз даних або публікації матеріалів в Інтернет-ЗМІ).

Веб-інтерфейс дає можливість універсального віддаленого доступу до служб та пристроїв, у даної технології практично нема альтернатив. Але водночас, оскільки такий інтерфейс доступний усім, постають серйозні питання забезпечення безпеки, зокрема автентифікація та авторизація користувачів, шифрування переданих даних від сторонніх очей, модерація вмісту тощо.

Перш ніж створювати веб-сторінки в редакторі, потрібно зробити макет майбутньої сторінки. Макет – це основа верстки, каркас, на якому збираються елементи сторінки та інформаційне наповнення.

Макет дозволяє створити цілісну картину елементів сторінки, єдність та відчуття неподільного образного ряду. Макет впорядковує структуру сторінки, робить її зручною та зрозумілою сприйняття. Іноді, макетом сайту називають готовий зверстаний шаблон сторінки сайту.

Перш ніж почати створення макету, необхідно визначити кількість принципово різних сторінок у майбутньому сайті.

Якщо перша сторінка сайту зовні дещо відрізняється від решти сторінок, тоді створення макету для сайту зводитися до планування однієї сторінки і єдиного шаблону, за яким будуть створено всі внутрішні сторінки. Всі сторінки майбутнього сайту повинні бути виконані в єдиному стилі, в окремих випадках компоновка текстової області на головній сторінці може

бути дещо складнішою, ніж у внутрішніх. Для такого сайту створюють один-два макети.

Якщо перша сторінка сайту суттєво відрізняється від внутрішніх, або сторінки розділів різняться за наявності різних типів матеріалів, тоді потрібно створювати кілька макетів для одного сайту, причому стилістично схожих між собою.

Веб-сторінка фактично розглядається як набір прямокутних блоків, які складаються в певному порядку. Завдання макетування полягає у гармонійному розташуванні різноманітних текстових та графічних блоків майбутньої сторінки [20].

Традиційно інформація розподіляється наступним чином [7]:

1) у верхній частині сторінки (шапка, header)

містяться: логотип; заголовок; слоган; телефон;

вибір мовної версії;

меню навігації (по сайту).

2) у середній (основній) частині сторінки:

меню навігації по тематичних розділах

сайту; основна інформація; зображення,

банери;

3) у нижній частині сторінки

(footer): копірайти; адреси,

телефони;

лічильники і банери;

Створимо проекти макетів сторінок, які будуть формувати інтерфейс веб-додатку у вигляді ескізів.

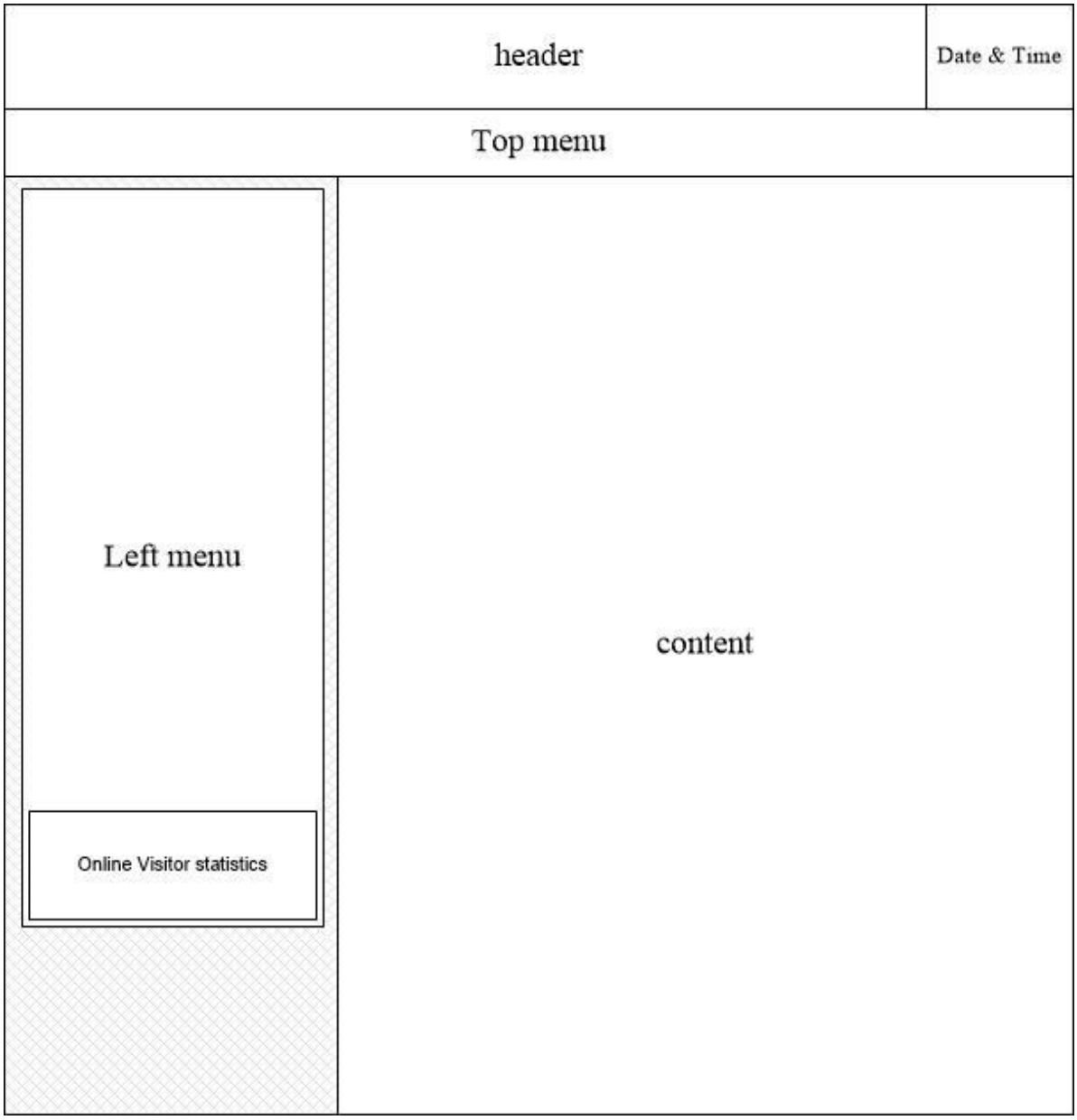


Рисунок 3.1 – Ескіз головної сторінки інформаційної системи

Як бачимо на рисунку 3.1 у верхній частині сторінки розміщується "header" із заголовком нашої інформаційної системи та поточним часом. Нижче шапки розміщується "topmenu", після чого сторінка сайту поділятиметься на дві частини: ліворуч – "leftmenu", а праворуч – вміст сторінки.

Меню складатиметься з категорій (груп), кожна з яких в свою чергу складається з відповідних підпунктів меню.

Внизу "leftmenu" відображатиметься он-лайн статистика користувачів на сайті. Аналогічні елементи веб-сторінок можемо спостерігати й на рисунках 3.2–3.3, на яких зображено макети сторінок "Залишити відгук" та "Діагностика" відповідно:

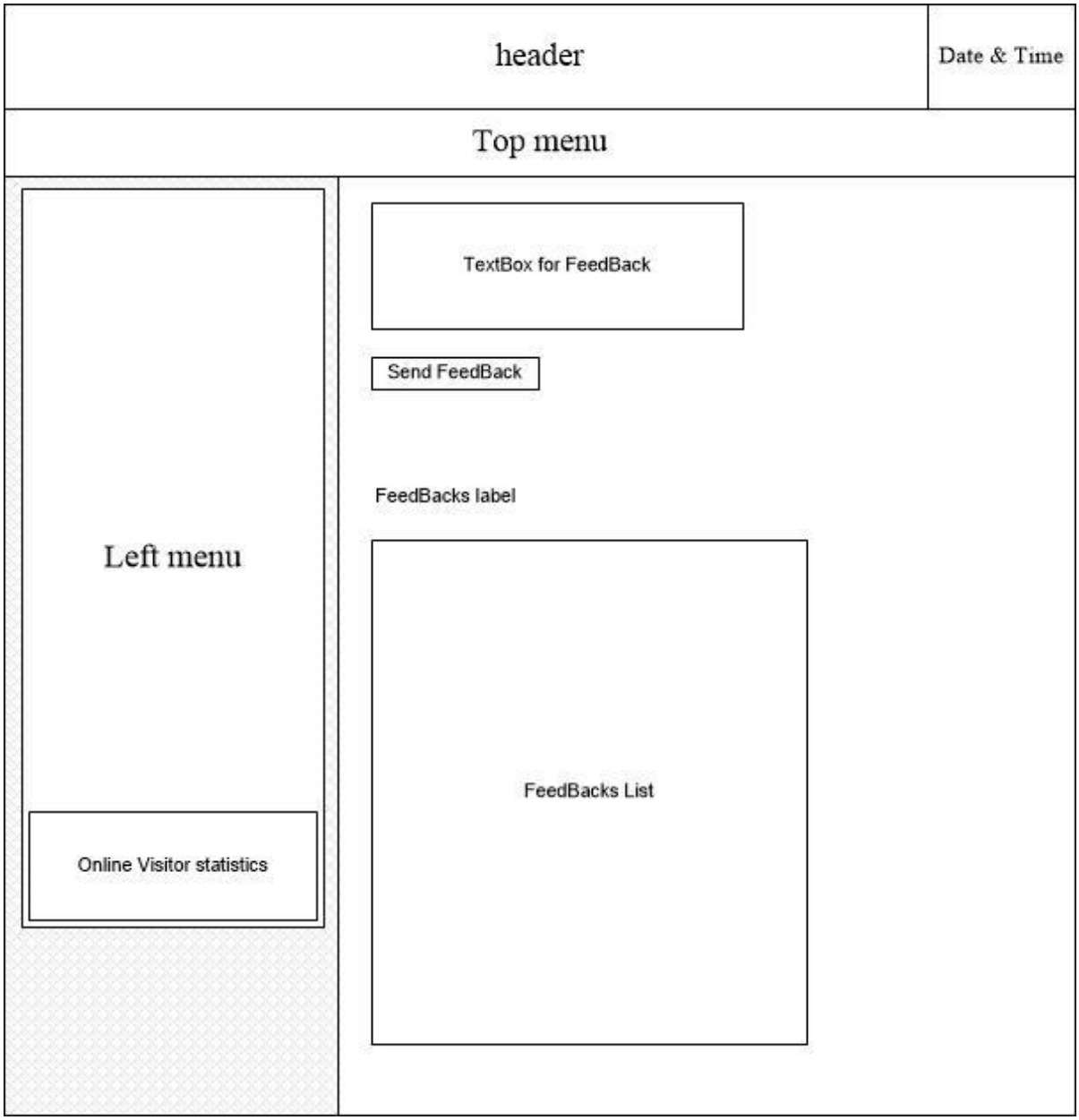


Рисунок 3.2 – Ескіз сторінки "Залишити відгук"

Розглянемо структуру сторінки "Залишити відгук" (рис. 1.2), головними відмінностями є те, що в області контенту відображаються: поле вводу відгуку та кнопка його відправки. Нижче них відображатиметься список відгуків з БД.

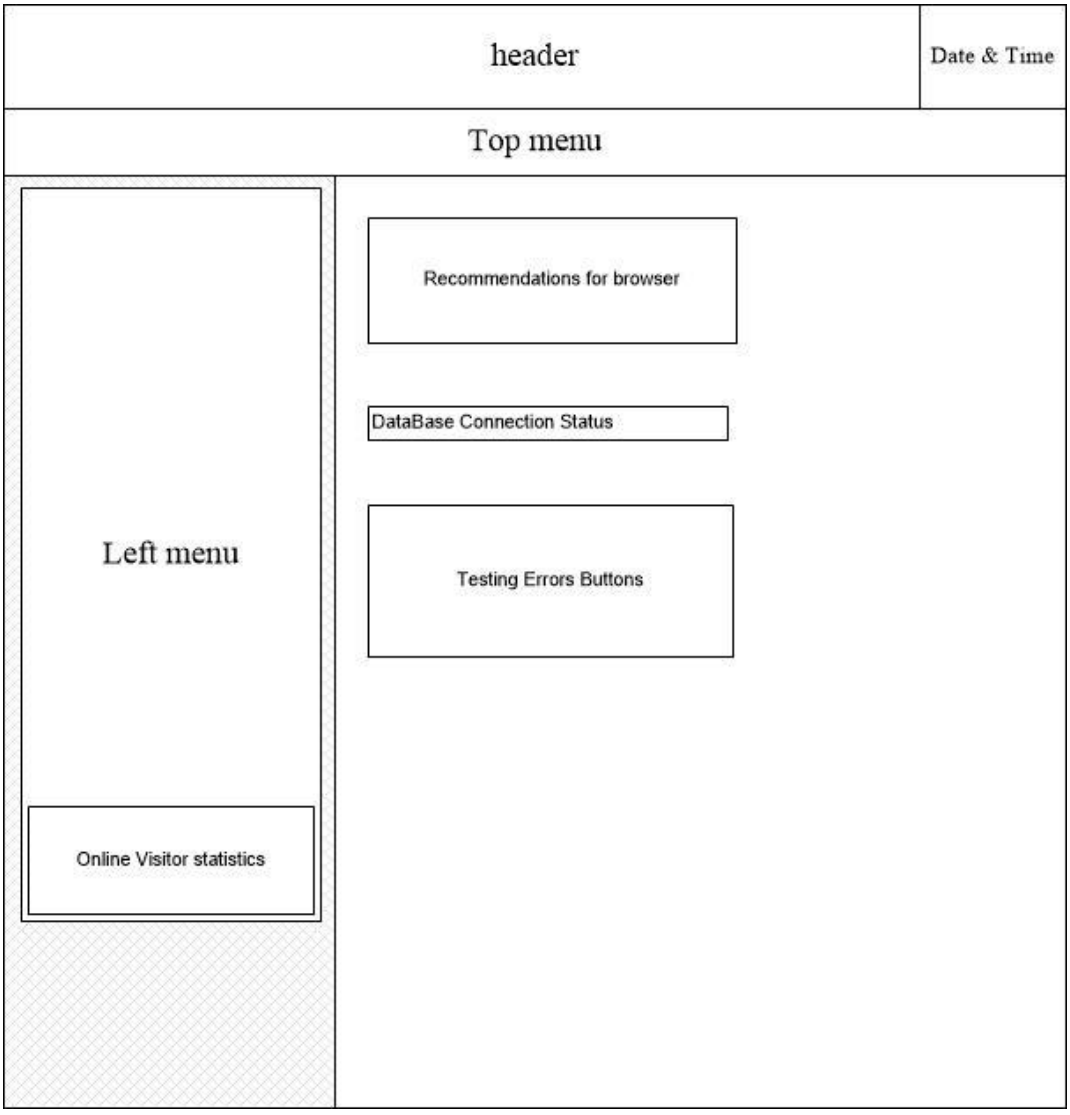


Рисунок 3.3 – Ескіз сторінки "Діагностика"

Розглянемо структуру сторінки "Діагностика" (рис. 3.3), головними відмінностями є те, що в області контенту відображаються рекомендації для браузера та результати тестування його придатності. Також відображається статус з’єднання сайту інформаційної системи з базою даних на сервері.

Нижче знаходитимуться кнопки для тестування перехоплення помилок сайтом.

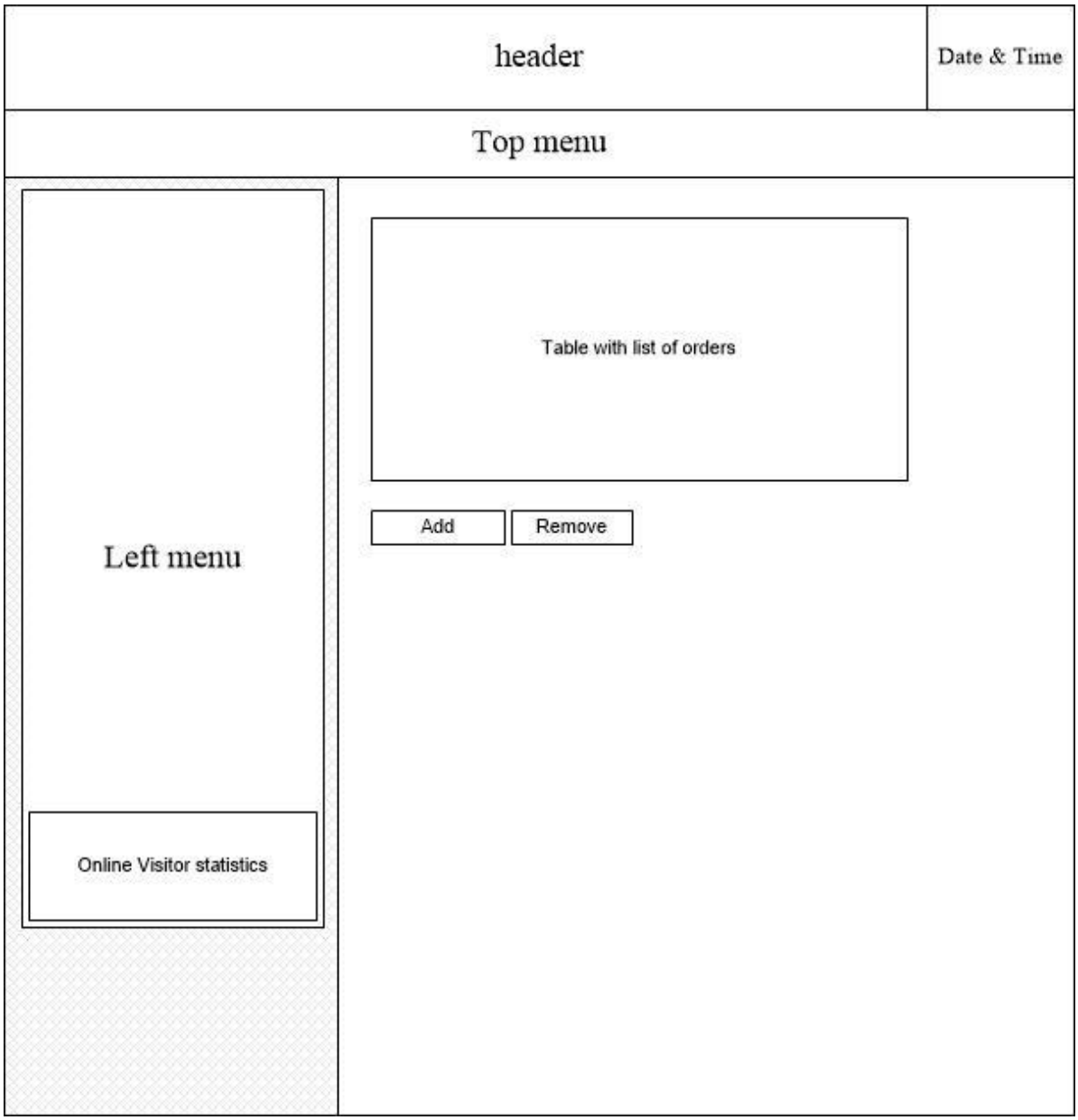


Рисунок 3.4 – Ескіз сторінки "Список заявок безробітних"

Розглянемо структуру сторінки "Список заявок безробітних" (рис. 3.4), головними відмінностями є те, що в області контенту відображається таблиця зі списком замовлень (ПІБ, Професія, Місце роботи, Стаж, Освіта тощо). Нижче знаходитимуться кнопки для додавання та видалення замовлень.

Розглянемо структуру сторінки "Додати заявку безробітного":

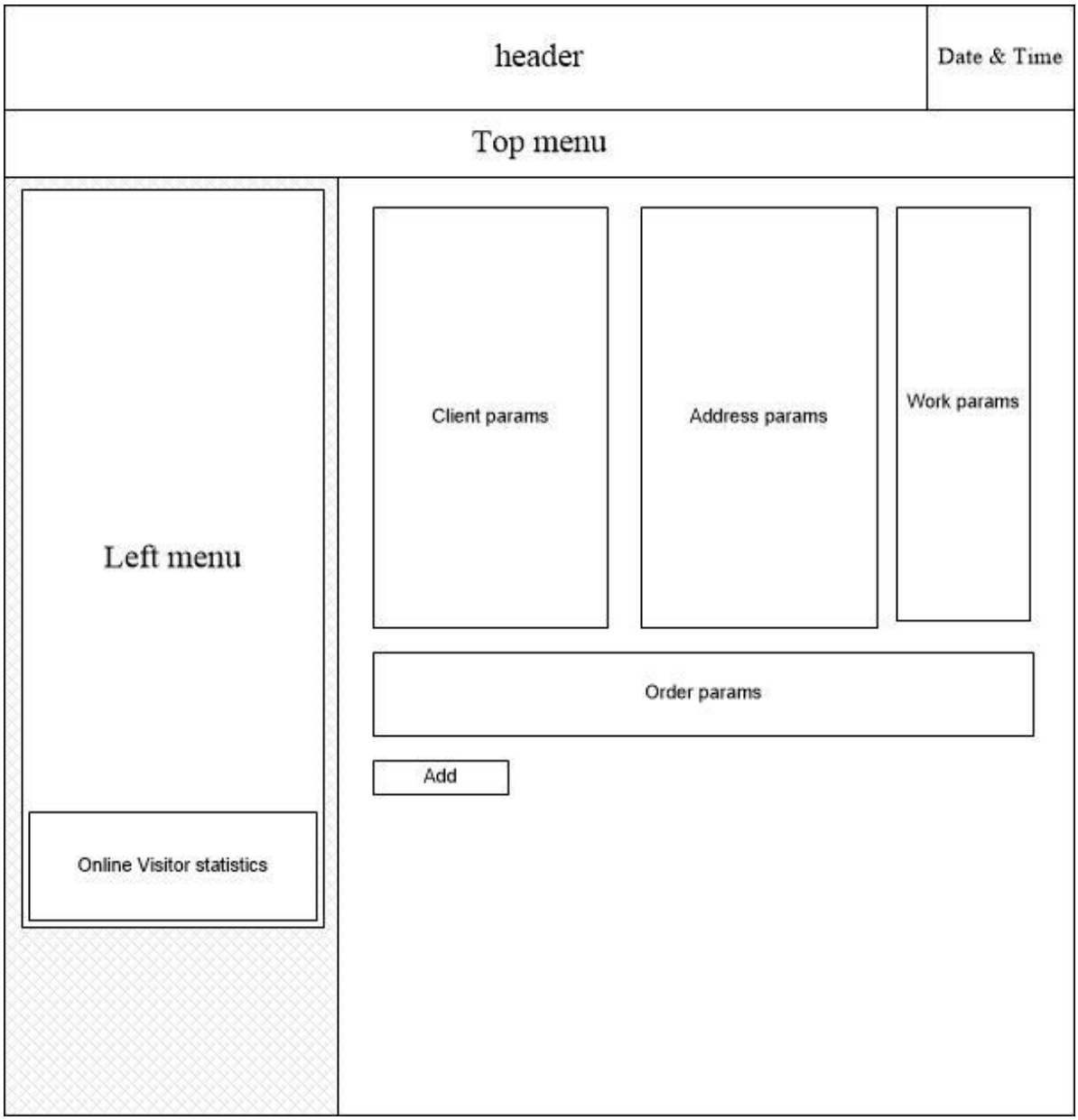


Рисунок 3.5 – Ескіз сторінки з інформацією про нову заявку

Розглянемо структуру сторінки "Додати заявку безробітного" (рис. 3.5), головними відмінностями є те, що в області контенту відображаються поля для введення параметрів заявки, безробітного, адреси, стажу, освіти тощо. Нижче знаходитиметься кнопка для додавання замовлення.

Розглянемо основні етапи створення сайту та подальшого його функціонування.

Основні етапи створення сайту:

проектування;

створення технічного завдання (ТЗ);

створення та затвердження графічного дизайну сайту;

перенесення дизайну у структуру веб-сторінок (верстка сайту);

програмування сайту (перетворення статичних веб-сторінок в інтерактивний додаток);

тестування;

запуск (перенесення сайту на сервер);

наповнення контентом (наповнення сайту тематичною інформацією);

пошукова оптимізація сайту (заходи для покращення позицій сайту у пошукових системах); розкрутка (залучення на сайт додаткового трафіку за рахунок

проведення рекламних кампаній або інших заходів що дають сайту додаткову кількість відвідувачів).

Не всі стадії створення сайту повинні бути проведені одноразово, деякі з них мають проводитися з певною періодичністю. Розглянемо коротко кожен із етапів.

Проектування – визначення основних цілей які мають бути досягнуті в результаті створення сайту. На цій стадії мають бути сплановані основні функції майбутнього сайту.

Створення технічного завдання – перелік та опис, у письмові формі, основних функцій та цілей сайту. Чим краще розтлумачені цілі, тим кращим буде кінцевий результат.

Створення та затвердження графічного дизайну сайту. Дана стадія може бути досить затяжною. Досить часто смаки дизайнера або дизайнерів, які будуть займатися створенням графічного дизайну, можуть не співпадати зі смаками замовника. Тільки після затвердження дизайну замовником, розробники сайту перейдуть до наступного етапу.

Перенесення дизайну у структуру веб-сторінок (верстка сайту). Даний етап не займає багато часу і носить суто технічний характер. Винятком можуть бути сайти у яких є велика кількість сторінок із різним дизайном.

Програмування сайту – це перетворення статичних веб-сторінок в інтерактивний додаток. Програмісти наповнюють сайт функціоналом, який був визначений у технічному завданні. По завершенні цього етапу, сайт перетворюється в інтерактивний веб-додаток і може бути перенесений на сервер.

Тестування – перевірка роботи усіх функцій сайту. Цей етап є дуже важливим, він дозволяє виправити усі помилки у роботі сайту до того, як сайт побачать реальні відвідувачі.

Запуск (перенесення сайту на сервер). Після перенесення сайту на ваш сервер (хостинг), сайт стає доступним з Інтернету.

Наповнення контентом (наповнення сайту тематичною інформацією). Для того щоб сайт став цікавим і потрібним для користувачів, на ньому має бути розміщена корисна інформація. Чим більше на сайті буде якісного контенту, тим кращими будуть результати роботи сайту.

Пошукова оптимізація сайту – це заходи для покращення позицій сайту у пошукових системах.

Розкрутка (залучення на сайт додаткового трафіку за рахунок проведення рекламних кампаній або інших заходів що дають сайту додаткову кількість відвідувачів). Розкрутка сайту проводиться з метою залучення якнайбільшої кількості відвідувачів на сайт. Для розкрутки сайту можна вдаватися до будь-якого виду реклами, від розклеювання оголошень до реклами по телебаченню.

Перш ніж починати створення сайту, необхідно вибрати тип нашого веб-сайту, для цього спершу розглянемо класифікацію існуючих типів сайтів.

В залежності від технології створення можна виділити наступні типи сайтів:

статичні сайти, що містять статичні HTML або XHTML сторінки.

Статичні веб-сторінки – це статичні файли (набір тексту, таблиць, малюнків і т.д.), які створюється за допомогою мови розмітки HTML (мають розширення.html або.htm) і зберігаються в готовому вигляді у файловій системі сервера;

динамічні сайти, в яких веб-сторінки генеруються або формуються (створюються динамічно) в процесі виконання запиту користувача. Динамічні сайти бувають двох типів. У першому типі сайтів, веб-сторінки генеруються або формуються з даних, які зберігаються на сервері в базі даних. У другому типі сайтів веб-сторінки генеруються на стороні клієнтського додатка (в браузері);

Flash-сайти – це інтерактивні програми, розроблені в середовищі Macromedia Flash. Основним інструментом розробки flash-програм є векторна графіка (інтерактивна векторна анімація для Web). Flash надає сайтам динамічність та інтерактивність;

комбіновані сайти, в яких використовуються вищевикладені технології створення сайтів.

Сайти по взаємодії користувача з ресурсами веб-сторінки можна розділити на пасивні та активні або інтерактивні.

Пасивні сайти – це сайти з пасивними веб-сторінками. У пасивних сайтах користувач має можливість тільки переглядати інформацію на веб-сторінках.

Інтерактивні сайти – це сайти з активними веб-сторінками. При роботі з інтерактивними веб-сторінками користувач має можливість обмінюватися даними з сервером, брати участь в інтерактивному діалозі.

Розглянемо статичні сайти з пасивними веб-сторінками. Технологія створення веб-сторінки статичних сайтів: мова HTML (Hyper Text Markup Language), яка є мовою розмітки гіпертексту та каскадні таблиці стилів CSS (Cascading Style Sheets). CSS використовується для оформлення та

форматування різних елементів веб-сторінок, в результаті чого значно знижують розміри веб-сторінок.

Створення веб-сторінок статичних сайтів – це трудомісткий процес. Статичні сайти з пасивними веб-сторінками створюються вручну, за допомогою редактора HTML у файловій системі комп'ютера, потім завантажуються на сайт. Створення нових веб-сторінок або редагування існуючих сторінок користувач виконує на ПК в редакторі, а потім знову завантажує на Web-сайт.

В основному статичні сайти з пасивними веб-сторінками застосовуються для створення невеликих і середніх сайтів з постійною структурою і зовнішнім виглядом сторінок (але кожна сторінка може мати свій шаблон оформлення), які можна розміщувати на будь-яких хостингах, у тому числі на безкоштовних, які не підтримують роботу скриптів.

Для створення сайту використовують різні засоби: редактори тексту типу "Блокнот", візуальні редактори типу "Microsoft FrontPage", "Macromedia Dreamweaver" і безліч інших редакторів, а також конструктори сайтів (дизайнери). Конструктори веб-сайтів розміщуються на деяких сайтах в мережі Інтернет.

Розглянемо статичні сайти з інтерактивними веб-сторінками. Для додання статичним веб-сторінкам інтерактивності і динамічності в веб-сторінку можна вставляти скрипти на мовах сценаріїв JavaScript і VBScript, виконуваних на стороні клієнта. Скрипти на JavaScript і VBScript можуть виконуватися або при наявності будь-яких дій користувача або автоматично під час завантаження веб-сторінки.

Крім того, в HTML документ можна вставляти елементи DHTML (динамічний HTML). DHTML – це спосіб створення інтерактивного веб-сайту. Динамічний HTML побудований на мові програмування JavaScript, каскадних таблицях стилів CSS і DOM (об'єктній моделі документа) [30].

У документ HTML можна вставляти флеш-фрагменти або Flash-ролики (swf-файли). У документ HTML можна вставляти Flash-форми аналогічні

HTML формам. Флеш забезпечує інтерактивність за рахунок інтерактивної векторної анімації для Web. Для створення Flash використовується мова сценаріїв ActionScript.

Для обміну даними між користувачем і сервером у веб-сторінку можна вставити веб-додаток, який називається HTML-формою (form). Форма – це частина веб-сторінки, в яку користувач може вводити свою інформацію і відправляти її на сервер. Запити обробляються на сервері, який генерує відповідну вихідну інформацію. Запити в формі можуть виконуватися методами GET або POST.

У зв'язку з тим, що скрипти, які виконуються на стороні клієнта, збільшують обсяг веб-сторінок, їх кількість і розмір на сторінці повинен бути обмеженим. Створення статичних сайтів з інтерактивними веб-сторінками доцільно виконувати в редакторі Macromedia Dreamweaver 8.

Розглянемо динамічні сайти, веб-сторінки яких генеруються або формуються з даних зберігаються на сервері в базі даних. В даний час для створення динамічних сайтів застосовуються різні веб-додатки. Для розробки веб-додатків застосовуються різні технології, що забезпечують створення динамічних веб-сторінок. Динамічні сайти здатні реагувати на введену користувачем інформацію, тобто можуть бути інтерактивними, тому динамічні сайти, як правило, є інтерактивними, але не завжди.

Для розробки веб-додатків використовують два підходи:

на основі компільованих модулів;

на основі інтерпретованих сценаріїв.

Компільовані модулі – це модулі типу CGI, які транслюються в виконувані файли і виконуються веб-сервером. Першими веб-додатками для створення динамічних сайтів були окремі модулі CGI (сценарії, створені в основному на мові Perl), які виконувалися на сервері. CGI-сценарії є звичайними програмами. Результатом виконання модуля є сторінка в форматі HTML.

Підхід на основі інтерпретованих сценаріїв – у цьому випадку для створення сайту застосовуються серверні скрипти, так звані мови сценаріїв. Код сценаріїв, як і HTML-код, інтерпретується кодом, тому HTML і сценарії можна комбінувати. Найбільш поширені мови серверних скриптів: Perl, ASP, JSP, PHP, Cold Fusion, Python.

Сценарії взаємодіють з об'єктами на сервері і генерують вихідну інформацію у форматі HTML. Тип серверного скрипта визначається з розширення імені файлу (.php,.asp,.aspx,.jsp,.cfm). Якщо Web-сервер отримує запит на сторінку такого типу, то він інтерпретує всі сценарії, які містяться в ній, в результаті чого генерується веб-сторінка у форматі HTML, яка передається назад браузеру.

Зважаючи на специфіку завдання, було з’ясовано, що сайт буде динамічним, веб-сторінки генеруються або формуються (створюються динамічно) в процесі виконання запиту користувача з даних, які зберігаються на сервері в базі даних.

Для створення веб-сайту обрано технологію ASP.NET. Дана технологія створення веб-додатків і веб-сервісів була розроблена компанією Microsoft. Вона є складовою частиною платформи Microsoft.NET і розвитком старішої технології Microsoft ASP. У цей час останньою версією цієї технології є ASP.NET 4.0.

ASP.NET зовні багато в чому зберігає схожість із старішою технологією ASP, що дозволяє розробникам відносно легко перейти на ASP.NET. У той же час внутрішній устрій ASP.NET істотно відрізняється від ASP, оскільки вона заснована на платформі.NET і, отже, використовує всі нові можливості, що надаються цією платформою.

Після випуску сервера Internet Information Services 4.0 в 1997 році, компанія Microsoft почала досліджувати можливість нової моделі веб-додатків, яка задовольнить скарги на ASP, зокрема пов'язані з відділенням оформлення від змісту, і яка дозволить писати "чистий" код. Робота з розробки такої моделі була доручена Марку Андерсу, менеджеру команди

IIS, і Скотту Гутрі, що прийшов на роботу в Microsoft в 1997. Андерс і Гутрі розробили початковий проект протягом двох місяців, і Гутрі написав код первісного прототипу під час різдвяних канікул 1997 року.

Для створення веб-додатку нашої інформаційної системи, необхідно у середовищі розробки MS Visual Studio 2008 натиснути "Файл/Створити" та обрати пункт "Веб-додаток ASP.NET"(рис 3.6, рис. 3.7):

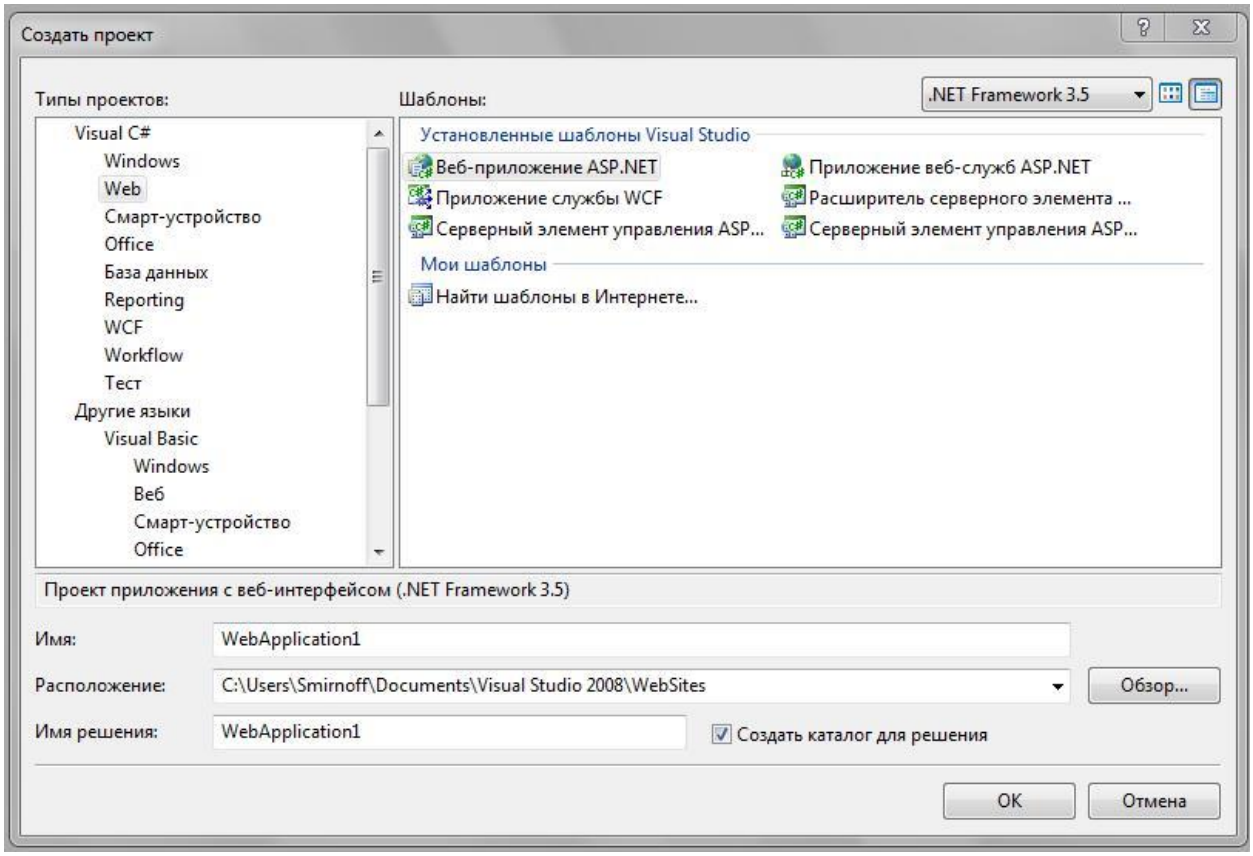


Рисунок 3.6 – Створення веб-додатку ASP.NET в середовищі розробки MS Visual Studio 2008

Одразу після натиску кнопки "OK", можемо спостерігати структуру новоствореного проекту у вікні "Оглядач рішень":

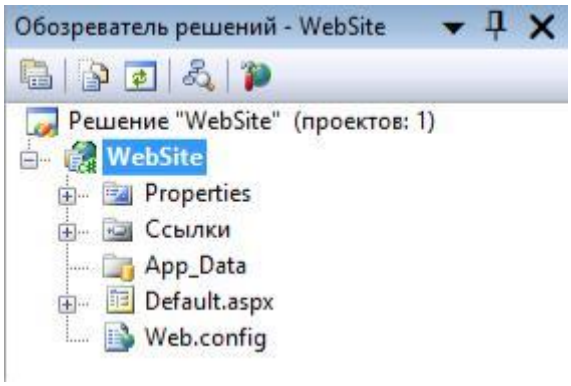


Рисунок 3.7 – Структура новоствореного проекту веб-додатку ASP.NET

Після чого можемо наповнювати наш веб-додаток інформацією та функціоналом.

Головні параметри нашого веб-додатку зберігаються у файлі "Web.config". Даний файл визначає параметри для ASP.NET web-додатків. Файл web.config – це XML-документ. У ньому зберігається інформація про параметри постачальників станів сеансів, членства, визначаються посилання на сторінки помилок. Також "web.config" містить рядки з'єднання з базами даних, засоби управління трасуванням тощо.

У кожному web-додатку повинен бути файл "web.config", що знаходиться в його кореневому каталозі. Однак дочірні каталоги можуть містити свої параметри конфігурації ASP.NET (які відрізняються від головного конфігураційного файлу відсутністю певних розділів).

Механізм успадкування файлів конфігурації ASP.NET наступний:

– спочатку застосовуються параметри з "machine.config"

– потім застосовуються параметри з файлу "web.config", що знаходиться в кореневому каталозі додатка.

– якщо в якомусь з каталогів додатка є файл "web.config", то застосовуються параметри з нього.

– повторюється п.3, поки файлів конфігурації не буде виявлено.

В нашому файлі "web.config" нам необхідно модифікувати рядок з’єднання з базою даних:

Розділ <customErrors> у файлі "Web.config" дозволяє налаштовувати параметри дій на випадок виникнення необроблених помилок під час виконання запиту. А саме, він дозволяє розробникам налаштовувати HTML-сторінки з повідомленнями про помилки, які будуть відображатися замість трасування стека помилок.

Master Pages (майстер-сторінки) – технологія Microsoft, що вперше з'явилася в ASP.NET 2.0. Дана технологія дозволяє створити зумовлену компоновку сторінок (наприклад, заголовок (шапку), меню, поля для відображення інформації, лічильники і т.д.) і багаторазово застосовувати її до різних сторінок додатку, наповнюючи елементи компоновки тим вмістом, якого вимагає контент сторінки.

Реалізацією технології майстер сторінок є два нових типи сторінок: майстер сторінка та сторінка вмісту.

Майстер-сторінка – це шаблон сторінки з деяким початковим фіксованим змістом. Вона може включати будь-яку комбінацію HTML елементів і коду, а також області, що можуть змінюватись – заповнювачі вмісту.

Сторінка вмісту (контентна сторінка) – сторінка, що посилається на одну майстер сторінку і отримує її компонування і заповнювачі. Крім того, сторінка вмісту може додавати дані в будь-який заповнювач.

Додамо майстер-сторінку до нашого рішення, для чого, в контекстному меню Провідника "Оглядач рішень" вузла рішення вибираємо пункт "Додання нового елементу".

Майстер-сторінка може містити HTML-код сторінки і може містити будь-які контроли, сценарії і т.п., тобто все, що може містити будь-яка веб-сторінка. Крім того, майстер-сторінка (і тільки вона і сторінки, для яких вона є батьківською) може містити (або не містити) контроли ContentPlaceHolder.

Даний елемент визначає область, яка може бути перевизначено сторінкою, що викликається з майстер-сторінки.

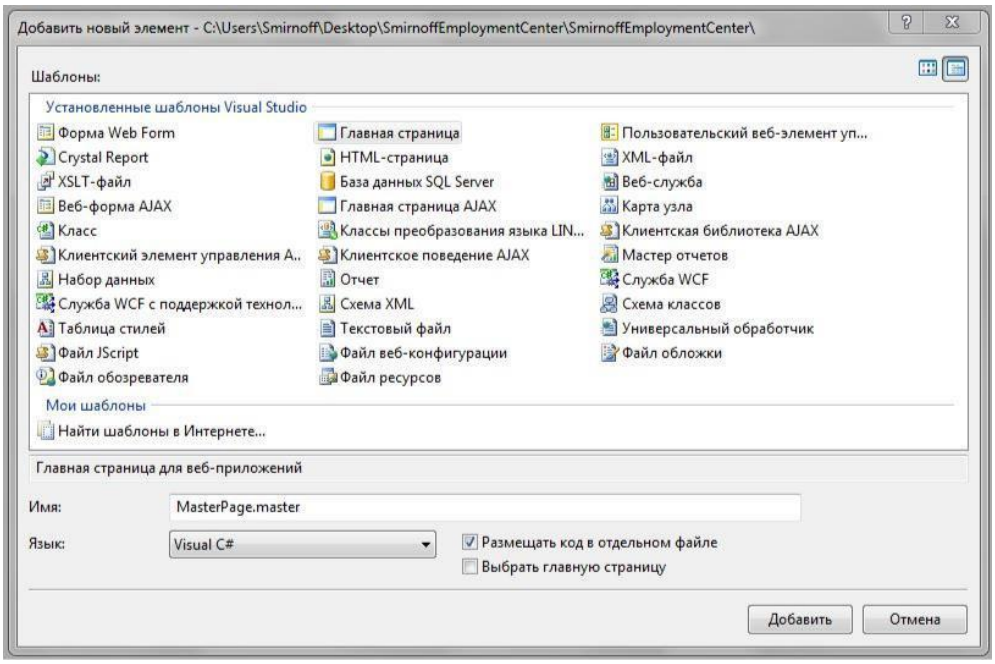


Рисунок 3.8 – Створення майстер-сторінки NET в середовищі розробки MS Visual Studio 2008

Код у файлі MasterPage.master має директиву "Майстер" (замість директиви "Page") – в іншому, він мало відрізняються від коду звичайної веб-сторінки. Новим може бути лише поява контрола "ContentPlaceHolder" – це елемент визначає область, яка може бути перевизначеною конкретною сторінкою вмісту.

Залишилося додати нашу майстер-сторінку до файлу рішення, для чого, у файлі "Main.aspx" додаємо параметр MasterPageFile:

Звернемо увагу, що значення параметра MasterPageFile починається з символів "~ /". Це означає, що майстер сторінка знаходиться не в наперед визначеній папці "MasterPages", а в кореневій директорії рішення (хоча, можна і створити таку папку). Причиною відсутності тегів <html>, <head>, <title>, <body> є те, що вони вже є в майстер-сторінці та в жодній іншій сторінці рішення бути не можуть.

Побудувавши майстер-сторінку отримали наступний її вигляд в конструкторі (рисунок 3.9) та відповідне відображення у вікні браузера (рисунок 3.10):

Рисунок 3.9 – Зображення вікна "Конструктор" майстер-сторінки

Рисунок 3.10 – Зображення головної сторінки сайту

Майстер сторінку можна додавати в рішення і через включення оголошення в web.config (<pages masterPageFile="MasterPage.master" />), але, в силу ряду недоліків і обмежень, цей шлях не рекомендований розробниками. HTML-код майстер-сторінки представлено в додатку А.

3.2 Структура та особливості реалізації алгоритмічного забезпечення

Додамо можливість навігації по сторінках веб-сайту за допомогою пунктів меню. Для цього у файлі "MasterPage.Master.cs" створимо необхідні нам події з перенаправленням на відповідні сторінки:

І відповідно прив’яжемо дані обробники подій до наших кнопок за допомогою властивості "OnClick":

Також, додамо відображення поточної дати та часу на сайті. Для цього в обробнику події "Page_Load" майстер-сторінки "MasterPage.Master" запишемо наступний код:

Додамо до нашого веб-додатку функціонал щодо визначення можливостей браузера і установка властивостей заголовка сторінки.

Нам необхідно програмно визначити наступні властивості браузера:

тип і версія;

платформа;

підтримка таблиць;

підтримка cookies;

підтримка скриптів.

Дані властивості потрібно показувати користувачам, щоб вони могли визначити, чи може їх веб-браузер підтримувати всі функціональні можливості, реалізовані на веб-сайті.

Для цього створимо окрему групу в меню "Служби веб-сайту" та підпункт "Діагностика", що посилатиметься на сторінку "Diagnostics.aspx".

При завантаження сторінки виникає подія "Page_Load", в обробнику, код якої ми й запишемо для визначення параметрів браузера:

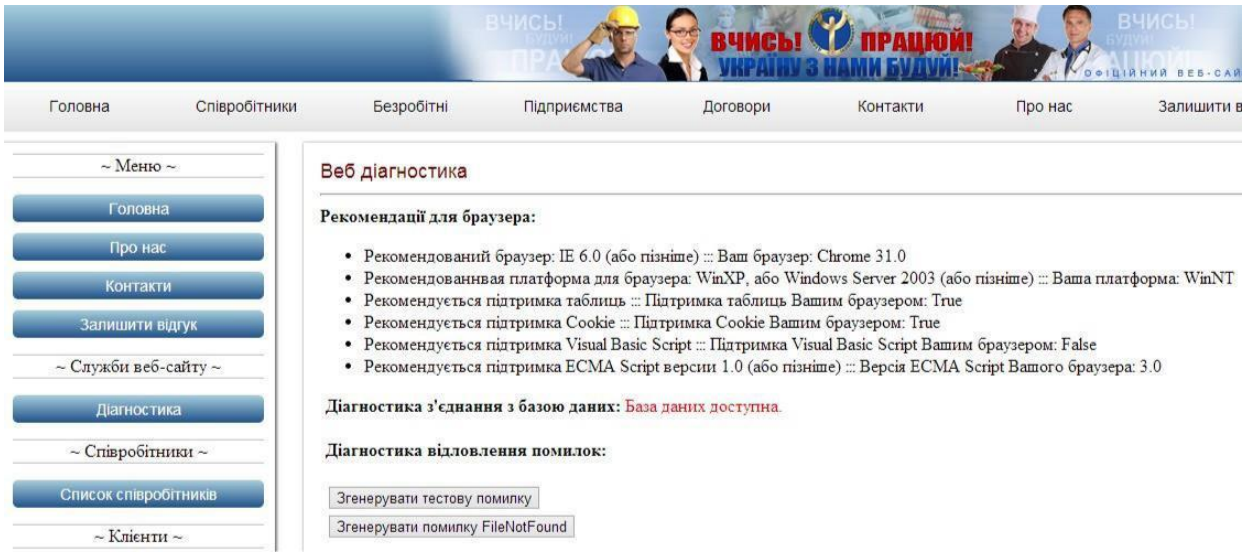


Рисунок 3.11 – Результат визначення параметрів веб-браузера користувача

Крім того необхідно програмно здійснити доступ до заголовка сторінки і змінити властивість "Title" об'єкта "Page.Header". Знову ж таки в обробнику події "Page_Load" запишемо наступний код:

Для організації обробки помилок на рівні додатків, змінити файл "Web.config" наступним чином:

вказати, що користувацькі помилки повинні бути дозволені;

встановити перенаправлення помилок за замовчуванням таким чином, щоб браузер перенаправляв необроблені помилки на "customErrorPage.aspx".

Для цього необхідно додати наступний рядок у файл "Web.config":

Важливою особливістю ASP.NET є використання серверних елементів управління на веб-сторінці (елементи WebForm), які є фактично тегами, зрозумілими веб-серверу. Ці елементи визначені в просторі імен System.Web.UI.WebControls.

Прийнято виділяти три типи серверних елементів управління:

серверні елементи управління HTML - звичайні HTML теги;

елементи управління веб-сервера - нові теги ASP.NET;

серверні елементи управління для перевірки даних– застосовуються для валідації вхідних даних від клієнтського додатка (зазвичай веб-браузера).

Переваги від використання таких елементів при розробці веб-додатків:

скорочується кількість коду, написаного вручну (що особливо помітно для складних елементів документа). Елемент просто "перетягується" з панелі інструментів, після чого виконується налаштування його параметрів у спеціальному вікні. При цьому всі зміни автоматично заносяться безпосередньо в *.aspx файл;

з програмної точки зору кожного з цих елементів управління відповідає певний клас у бібліотеці базових класів. NET, що дозволяє писати для них такий же код як і для будь-яких інших класів;

для будь-якого елементу управління WebForm визначений набір подій, оброблюваних на веб-сервері;

для будь-якого елементу управління WebForm надається можливість для перевірки вводу даних користувачем.

За замовчуванням такі елементи управління в ASP.NET файлах розглядаються як текст. Для їх програмування потрібне додавання атрибуту runat = "server" у відповідний HTML елемент. Крім того, всі серверні елементи управління HTML повинні бути розміщені всередині області дії тега <form>, також має атрибут runat = "server".

Таблиця 3.1 – Серверні елементи управління

№	Серверний елемент управління	HTML Опис
1	HtmlAnchor	Управління HTML елементом <a>
2	HtmlButton	Управління HTML елементом <button>

Продовження таблиці 3.1

3	HtmlForm	Управління HTML елементом <form>
4	HtmlGeneric	Управляє HTML елементами не описуваними як елементи керування HTML, наприклад, <body>, , та ін
5	HtmlImage	Управління HTML елементом <image>
6	HtmlInputButton	Управління HTML елементами <input type="button">, <input type="submit"> і <input type="reset">
7	HtmlInputCheckBox	Управління HTML елементом <input type="checkbox">
8	HtmlInputFile	Управління HTML елементом <input type="file">
9	HtmlInputHidden	Управління HTML елементом <input type="hidden">
10	HtmlInputImage	Управління HTML елементом <input type="image">
11	HtmlInputRadioButton	Управління HTML елементом <input type="radio">
12	HtmlInputText	Управління HTML елементами <input type="text"> і <input type="password">
13	HtmlSelect	Управління HTML елементом <select>
14	HtmlTable	Управління HTML елементом <table>
15	HtmlTableCell	Управління HTML елементами<td> і<th>

Продовження таблиці 3.1

16	HtmlTableRow	Управління HTML елементом <tr>
17	HtmlTextArea	Управління HTML елементом <textarea>

Одним із найбільш поширених підходів до розробки клієнтських прикладних програм для взаємодії з реляційними базами даних під ОС Windows є використання інструментарію Visual Studio корпорації Microsoft. MS Visual Studio – це інтегроване середовище розробки (Integrated Development Environment (IDE)) для створення, документування, запуску та коректування програм, написаних на сучасних мовах створення програмних систем. Це потужний інструмент професійної розробки складних прикладних програм, один з найкращих у світі.

Використаємо можливості застосування об’єктів технології ADO (Active Data Objects) для відображення даних, які зберігаються в персональній реляційній базі даних. Також, слід вивчити можливості використання технології ODBC (Open Database Connectivity) для підключення клієнтської прикладної програми до різних джерел даних без необхідності корегувань програмного коду та перекомпіляції.

Модель об’єктів ADO визначається набором класів, які інкапсулюють в собі більшість можливостей проведення операцій з базами даних, яка наслідує при створенні їх екземпляри – об’єкти. За операцію відображення даних відповідає об’єкт класу-наслідувача DataSet, в який завантажуються дані з представлень чи базових таблиць і потім передаються в інтерфейс програми без підтримки неперервного зв’язку з базою даних. Також, передбачений обернений процес передачі даних із DataSet у базові таблиці, в складі об’єкту DataSet є властивість, яка представляє собою колекцію об’єктів DataTable. Вони використовуються для представлення відповідних таблиць, тобто виступають таблицями, генерованими при отриманні результатів із бази даних. Повинен бути реалізований механізм взаємодії з

джерелом даних. Такий механізм забезпечує об’єкт типу DataAdapter.
Сама назва даного об’єкту вказує на перетворення.

Рекомендується застосовувати явне управління з'єднанням, так як воно робить більш чистий і зручний для читання програмний код, допомагає при корегуванні прикладної програми, є більш ефективним засобом організації роботи з використанням баз даних та з’єднуванням їх до програмних компонентів [25].

Технологія ODBC передбачає використання єдиного інтерфейсу для доступу до різних серверів реляційних баз даних. Цей інтерфейс забезпечує високу ступінь універсальності, в результаті якої один і той же запит може отримувати доступ до даних, які зберігаються в базах під управлінням різноманітних систем управління базами даних без необхідності внесення його програмного тексту. Таким чином розробники отримують інструмент для створення та поширення програм архітектури клієнт-сервер, які здатні працювати з широким спектром систем управління базами даних.

Зв’язати програми з будь-якою СУБД можна засобом використання відповідного ODBC драйвера. На сучасному етапі розвитку технологія ODBC фактично отримує значення галузевого стандарту. Основна причина популярності міститься в наявній у неї гнучкості, що пропонує розробникам такі переваги, як програма більше не зв’язана з прикладним API якої-небудь однієї системи управління базами даних, прикладна програма здатна ігнорувати особливості використовуваних протоколів у передачі даних, дані посилаються та доставляються в тому форматі, який є найбільш зручним для конкретного використання, на сьогоднішньому етапі існують драйвери ODBC для різноманітних типів самих різних систем управління базами даних.

Розглянемо прив’язку даних до GridView, який відображатиме інформацію про заявки безробітних. Помістивши GridView на сторінку, задамо йому джерело даних:

Рисунок 3.12 – Задання джерела даних для елементу GridView

Вибравши у випадаючому списку елемент "Нове джерело данних…", отримали вікно Майстра конфігурації джерела даних, в якому й вибираємо джерелом – базу даних та задаємо ідентифікатор для нього.

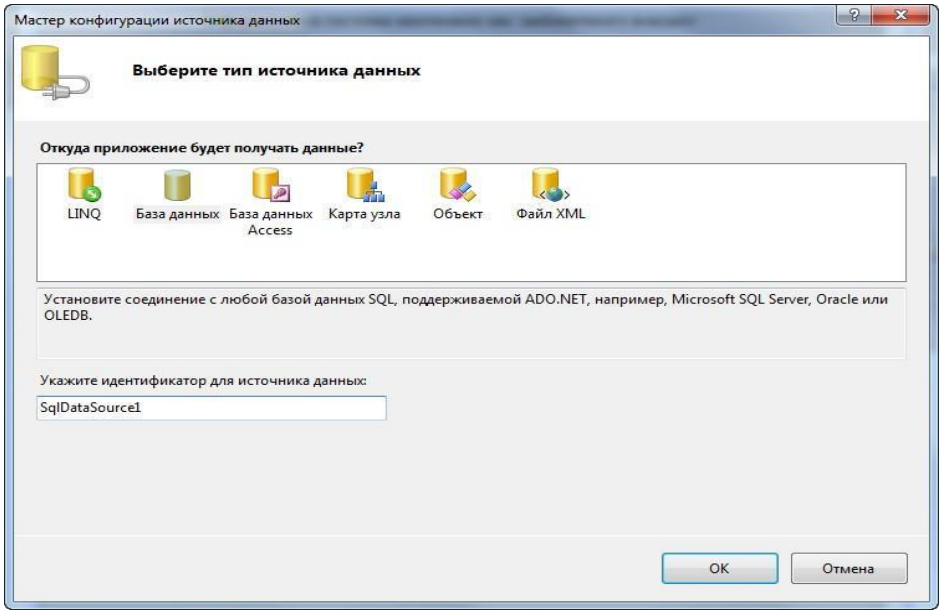


Рисунок 3.13 – Вибір типу джерела даних

Наступним кроком буде обрання нашої БД. Обравши базу даних, необхідно налаштувати оператор "Select або обрати вже наявну таблицю чи представлення.

Оберемо реалізоване представлення "СписокЗаявокКлієнтів":

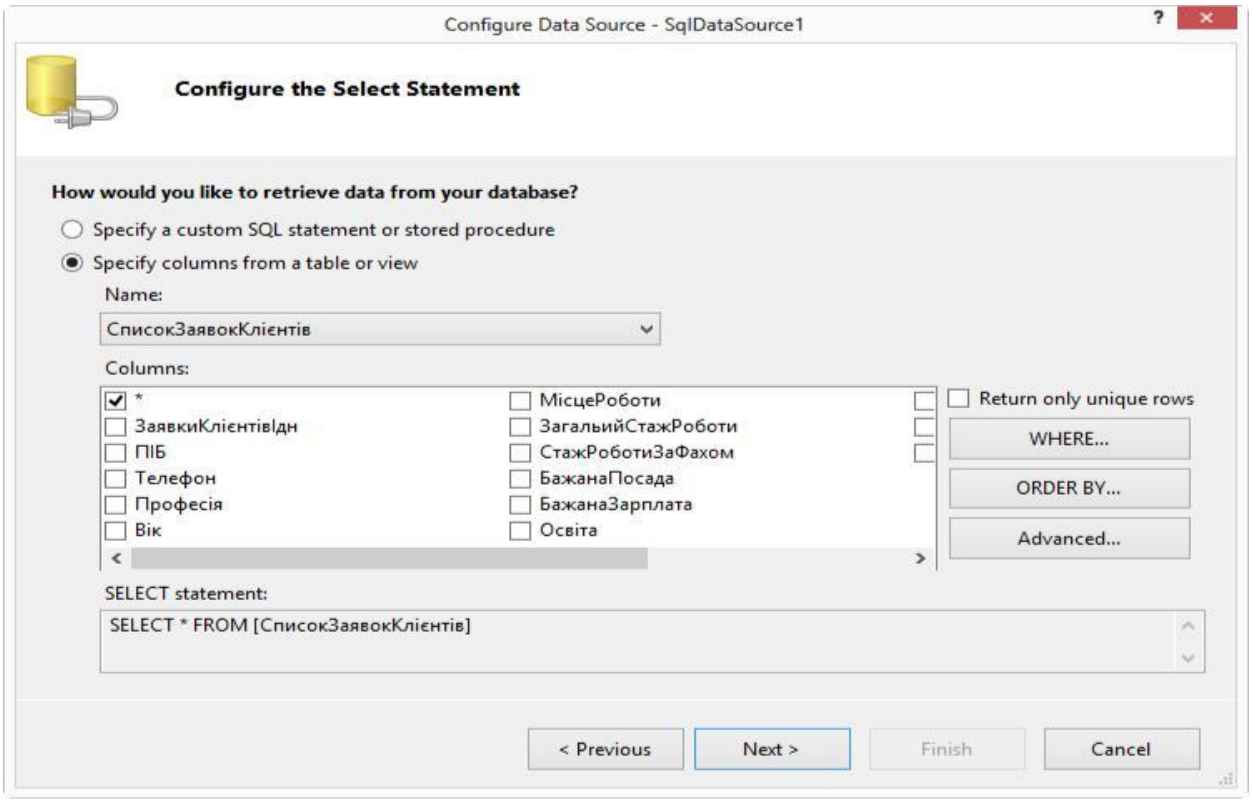


Рисунок 3.13 – Налаштування оператору "Select"

Після чого натискаємо кнопки "Далі" та "Готово". В результаті зв'язано до елемента GridView результати представлення "СписокЗаявокКлієнтів", що вибирає з БД інформацію про безробітного, стаж роботи, освіту, бажану посаду, володіння мовами, посвідчення водія тощо.

3.3 Реалізація прототипу веб-орієнтованої інформаційної системи центру зайнятості

Розглянемо механізм маніпулювання даними в складних реляційних структурах на основі запуску з Веб-додатка збережених процедур бази даних, які призначені для вставки, модифікації і видалення даних в декількох зв'язаних таблицях бази даних.

На сторінці "Список заявок клієнтів", також маємо можливість додавати заявку натиснувши кнопку "Додати", після чого переходимо на сторінку введення параметрів нової заявки.

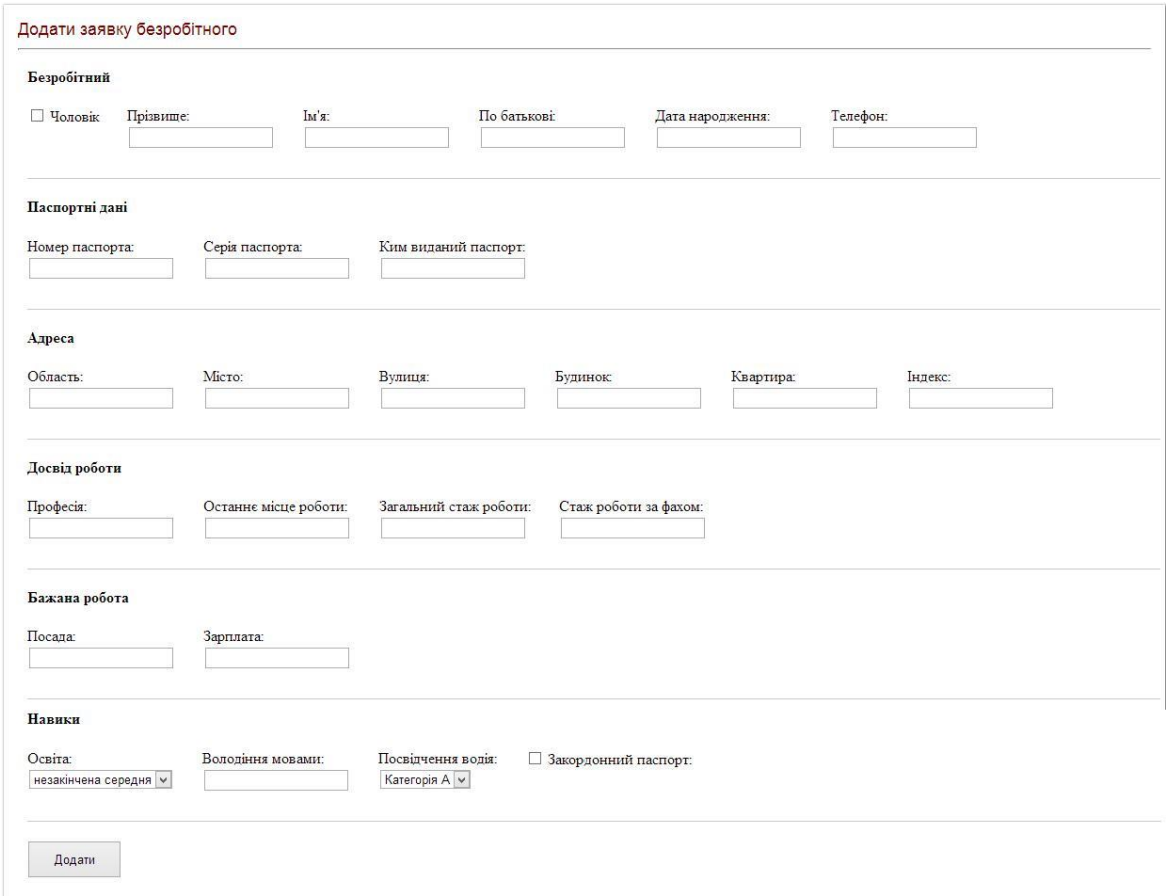
Заявки безробітних

№	ПІБ	Телефон	Професія	Вік	Місце роботи	Загальний стаж	Стаж за фахом	Бажана посада	Бажана зарплата	Освіта	Володіння мовами	Посвідчення водія
1000000000	Пахомова Мілана Володимирівна	805073441245	Програміст	26	Сумнасосенергомаш	5	3	Адміністратор БД	7000.000	вища	українська, англійська, німецька	Категорія В
1000000001	Дмитров Денис Юрійович	відсутній	Електрик	53	ЕКО Маркет	30	20	Електрик	3200.000	вища	українська, російська	Категорія В
1000000002	Свирська Тетяна Іванівна	806623542902	Менеджер	33	Мобілочка	8	2	Менеджер-консультант	3200.000	незакінчена вища	українська, англійська	Відсутнє
1000000003	Латутько Валентина Василівна	809543565073	Вчитель математики	42	Шосткинська школа №1	20	20	Вчитель математики	3200.000	вища	українська, російська	Категорія В
1000000004	Усик Іван Васильович	809923497005	Вантажник	36	VICOTEC	20	5	Вантажник	3500.000	незакінчена середня	українська, російська	Категорія В
1000000005	Коваль Володимир Ігорович	80678562383	Дизайнер	22	не визначено	1	1	Арт-дизайнер	4000.000	незакінчена вища	українська, російська, англійська	Відсутнє
1000000006	Слабоспицька Ірина Петрівна	806745572999	Продавець	25	ЕЛЕКТРОМАРКЕТ	2	2	Продавець-консультант	3900.000	незакінчена вища	українська, англійська	Відсутнє
1000000007	Чеховський Віктор Олексійович	806739941182	Охоронець	51	УАБС НБУ	30	10	Охоронець	3200.000	середня	українська, російська	Категорія В
1000000008	Домарєва Інна Іванівна	809932977173	Лікар	32	Сумська дитяча поліклініка	8	6	Отоларинголог	4100.000	вища	українська, російська	Відсутнє
1000000009	Хаблак Григорій Андрійович	806799967854	Сантехнік	37	ВАТ "Приват Банк"	11	5	Сантехнік	3500.000	середня	українська, російська	Категорія В

1 2

Додати

Рисунок 3.14 – Зображення таблиці зі списком заявок від безробітних



Додати заявку безробітного

Безробітний

☐ Чоловік Прізвище: Ім'я: По батькові: Дата народження: Телефон:

Паспортні дані

Номер паспорта: Серія паспорта: Ким виданий паспорт:

Адреса

Область: Місто: Вулиця: Будинок: Квартира: Індекс:

Досвід роботи

Професія: Останнє місце роботи: Загальний стаж роботи: Стаж роботи за фахом:

Бажана робота

Посада: Зарплата:

Навики

Освіта: незакінчена середня Володіння мовами: Посвідчення водія: Категорія А ☐ Закордонний паспорт:

Додати

Рисунок 3.15 – Зображення сторінки «Додати заявку безробітного»

В додатку Е представлено HTML-код елементів для відображення поточної сторінки, а також програмний код для додавання нової заявки

Проведемо тестування нашого функціоналу. Для початку спробуємо додати нову заявку:

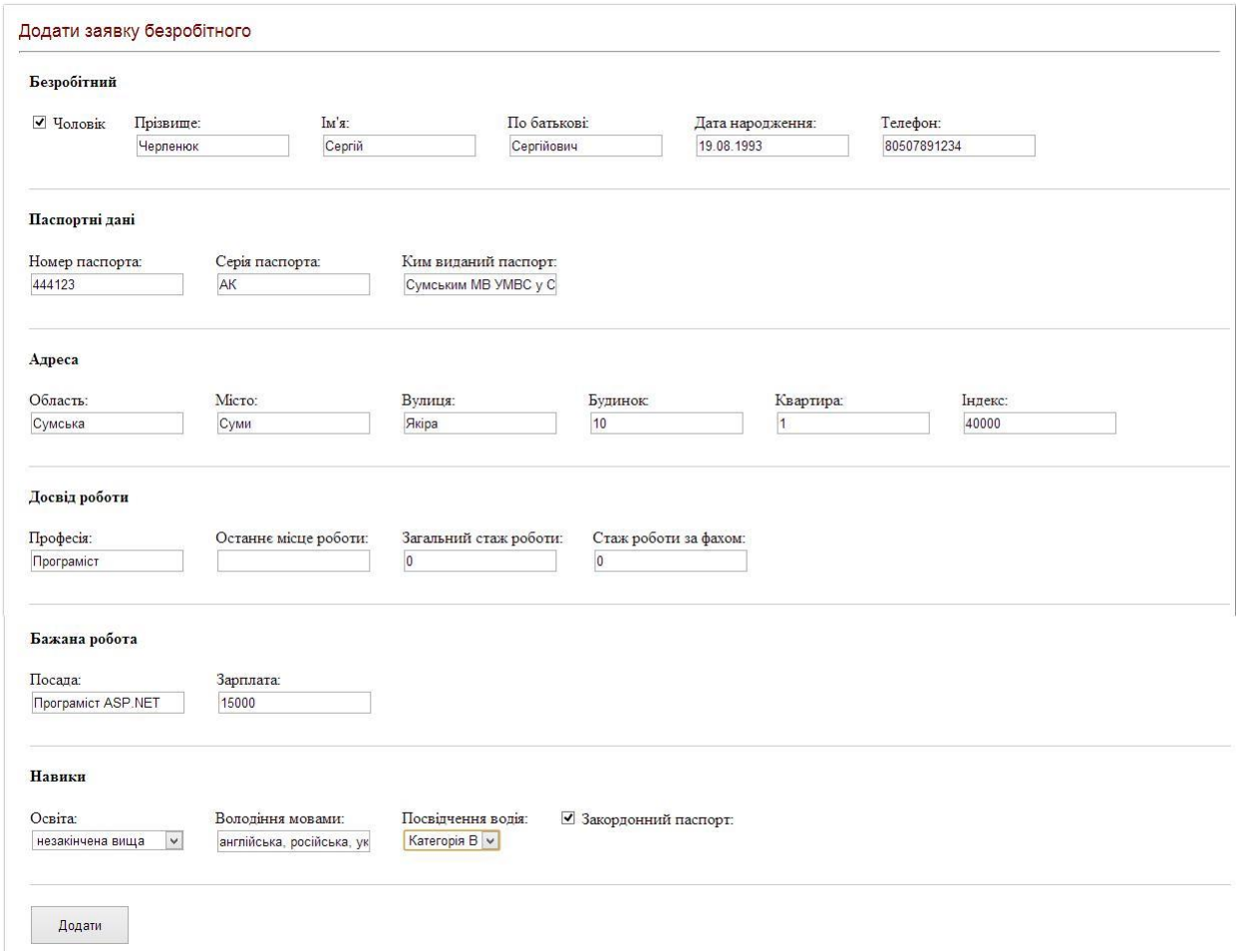


Рисунок 3.16 – Додавання нової заявки

Після натиснення на кнопку «Додати», можемо спостерігати, що нова заявка відображається в списку:

Заявки безробітних

№	ПІБ	Телефон	Професія	Вік	Місце роботи	Загальний стаж	Стаж за фахом	Бажана посада	Бажана зарплата	Освіта	Володіння мовами	Посвідчення водія
1000000010	Голенков Ігор Леонідович	805073722372	Кухар	34	Ресторан "Рішельє"	15	15	Кухар	2000,0000	вища	українська, російська, англійська, німецька	Категорія В
1000000011	Сергій Черленюк Сергійович	507891234	Програміст	20		0	0	Програміст ASP.NET	15000,0000	незакінчена вища	англійська, російська, українська	Категорія В

1 2

Додати

Рисунок 3.17 – Зображення таблиці наявних заявок безробітних з новододаною заявкою

Так само на сторінці "Список заявок роботодавців", також маємо можливість додавати заявку натиснувши кнопку "Додати", після чого переходимо на сторінку введення параметрів нової заявки.

Заявки роботодавців

№	Назва фірми	Посада	Оплата	Вимоги	Телефон	Адреса
100000001	ПАТ "Альтком"	Програміст	8000	вища освіта	80952836432	Комсомольська, 27
100000002	ПАТ "Samsung Electronics"	Менеджер	12000	вища освіта	80632564523	Жовтнева, 25
100000003	ПАТ "Мироннівпродукт"	Бухгалтер	4000	вища освіта	80503871014	Заливная, 7
100000004	ПАТ "Оптима Фарм"	Журналіст	6000	вища освіта	80994568912	Академическая, 13
100000005	ПАТ "Авангард"	Ветеринар	5320	вища освіта	80951237898	Парижской Коммуны, 70
100000006	ПАТ "Auchan Україна"	Водій	3200	незакінчена вища освіта	80637541595	Лукьяновская, 62
100000007	ПАТ "Керамет"	Охоронець	3200	незакінчена вища освіта	80508774383	Лебединская, 15
100000008	ПАТ "Паралель"	Сантехнік	3300	незакінчена вища освіта	80507845123	Д. Коротченка, 7
100000009	ПАТ " Нова лінія"	Аналітик	15000	вища освіта	80631234598	Прокофьева, 9

1 2

Додати

Рисунок 3.18 – Зображення таблиці зі списком заявок від роботодавців

Таким чином реалізовано мету дослідження, а саме побудувано веб-орієнтовану інформаційну систему центру зайнятості.

3.4 Оцінка очікуваного ефекту від впровадження системи автоматизації

В даному підрозділі кваліфікаційної роботи розраховуються значення основних та додаткових показників економічної ефективності розробленої автоматизованої інформаційної системи.

Для з'ясування економічної ефективності розробки веб-сайту центру зайнятості необхідно розрахувати наступні показники:

– визначити річний ефект від впровадження організаційно-економічної автоматизованої системи управління;

– визначити коефіцієнт ефективності капітальних вкладень;

– визначити термін окупності витрат (То) на впровадження системи;

– при якому терміні окупності річний ефект буде дорівнювати 0?

Загальним критерієм доцільності автоматизації рішення економічних задач є досягнення підвищення ефективності виробництва за допомогою вдосконалення системи управління. З цієї точки зору зниження витрат на обробку інформації не є головним чинником. Проте ці витрати входять у витрати виробництва і між ними і загальною ефективністю автоматизації існує пряма пропорційна залежність. Тому вибір більш економічних варіантів автоматизації є важливою умовою проектування машинного рішення економічних задач. Необхідно вибрати методику, яка б дозволила зіставити показники, що характеризують автоматизоване рішення задачі з показниками рішення задачі за технологією базового варіанту, тобто технологією, за якою здійснювалось рішення до впровадження автоматизації.

Позначення, які необхідні для розрахунку ефективності ІС наведені в таблиці 3.2.

Таблиця 3.2 – Основні позначення розрахунку ефективності системи

№	Позначення	Пояснення
1	Впроект	Витрати на проектування комплексу задач
2	Впрогр	Витрати на програмування комплексу задач
3	Вн	Витрати на програмування в період налагодження ПЗ комплексу задач
4	Ввпров	Витрати на впровадження
5	Вб	Приведені до одного року витрати на обробку інформації при базовому варіанті організації обробки
6	Вп	Приведені до одного року витрати на обробку інформації при впроваджуваному (пропонованому) варіанті організації системи обробки
7	Еу	Річний економічний ефект
8	S	Річна економія, яку отримає підприємство в результаті впровадження сайту
9	С	Капітальні вкладення, які було витрачено в результаті впровадження сайту
10	rn	Нормативний коефіцієнт окупності капітальних вкладень, узятий для конкретної галузі
11	Р	Термін окупності витрат на впровадження проекту машинної обробки

Для оцінки ефективності системи розрахуємо наступні показники:

– суму капітальних витрат, які включають витрати на проектування і упровадження проекту, придбання необхідного устаткування і його установку, програмування комплексу задач та його налагодження;

– суму річної економії, яка розраховується як різниця між річною вартісною оцінкою результатів використання СОЕІ об'єктом управління і приведеними до даного року витратами;

– річний економічний ефект від розробки і упровадження АІС визначається, як різниця між сумою річної економії (річний приріст прибутку);

– термін окупності витрат на впровадження проекту машинної обробки інформації, який розраховується як відношення капітальних витрат на розробку і впровадження системи обробки до річної економії від впровадження.

Таблиця 3.3 – Розрахунок ефективності системи

№	Формула	Опис формули	Розрахунок
1	С = Впроект + Впрограм + Вн + Ввпров	Сума капітальних витрат	11865+12340+272+444=24921 (грн.)
2	S=Вб–Вп	Сума річної економії	12045 – 444 = 11601 (грн.)
3	E S C r	Річний ефект від впровадження	11601 – 8753 = 2848 (грн.)
4	P = C/S = 1/Rce	Термін окупності	24921 / 11601 = 2,15 (р.)

Звертаючи увагу на результати обчислень отримали наступні значення показників:

– сума капітальних витрат дорівнює 24921 грн.;

– сума річної економії складає 11601 грн.;

– річний ефект від провадження веб-сайту дорівнює 2848 грн.

– термін окупності впровадженої веб-орієнтованої інформаційної системи за проведеними розрахунками складає 26 місяців.

ВИСНОВКИ

Одним з факторів розвитку економіки Україні є максимальне задоволення потреб підприємств у трудових ресурсах. Для забезпечення потреб підприємств робочою силою існують державні центри зайнятості. В усіх обласних центрах України створено служби зайнятості, основним завданням яких є підбір персоналу для трудових потреб регіону. Все це обумовлює актуальність проведеного дослідження – розробку веб-орієнтованої інформаційної системи центру зайнятості.

Відповідно до мети кваліфікаційної роботи в проведеному дослідженні:

– розглянуто стан ДУ "Сумський міський центр зайнятості", та сформовано вимоги до системи автоматизації;

– описано функціонал веб-сайту, що розробляється;

– спроектовано систему автоматизації центру зайнятості;

– побудовано моделі бізнес-процесів центру зайнятості;

– для реалізації проекту обрано трирівневу клієнт-серверну архітектуру;

– для створення веб-сторінки Сумського міського центру зайнятості використано програмний інструментарій середовища розробки Visual Studio, а саме технологію ASP.NET;

– описані програмні та апаратні вимоги до системи;

– побудовано структуру веб-сайту та наведені особливості реалізації інформаційного забезпечення;

– реалізована веб-орієнтована інформаційна система обслуговування клієнтів центру зайнятості.

Подальшим розвитком системи може бути розширення функціоналу розділом правових консультацій

СПИСОК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. Архитектура клиент-сервер [Електронний ресурс]. – Режим доступу:

2. Братушка С.М.Системний аналіз, навч.-метод.посібник [Електронний ресурс]. – Режим доступу: http://ek-portal.com/materials /category/sa.

3. Бітрікс24 [Електронний ресурс]. – Режим доступу: https://www. bitrix24.ua/articles/crm_client_base.php.

4. Галузинський Г.П. Перспективні технологічні засоби оброблення інформації, навч.-метод. посібник / Г.П.Галузинський, І.В. Гордієнко. – К. : КНЕУ, 2009. – 280 с. – ISBN 966-574-359-7.

5. Державна служба зайнятості [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/67424.

6. Динаміка використання Інтернет в Україні [Електронний ресурс]. – Режим доступу: http://www.kiis.com.ua/?lang=ukr&cat=reports&id=621.

7. Діренко І. С. Система управління вмістом веб-ресурсів на основі онтологічно-документного моделювання / І. С. Діренко, О. В. Олецький. – Теоретичні та прикладні аспекти побудови програмних систем. Матеріали міжнародної конференції TAAPSD'2007. / Київ, грудень 2007 р. – С.171-176.

8. Інформаційна система "EIAS NET" [Електронний ресурс]. – Режим доступу: http://prog.bobrodobro.ru/40364.

9. Інформаційні архітектрури [Електронний ресурс]. – Режим доступу: http://elchaninowa-natalja.narod.ru/index/0-2.

10. Пономаренко В.С. Інформаційні системи і технології в економіці: Навчальний посібник/ В.С. Пономаренко, Р.К. Бутова, І.В. Журавльова, Г.Н. Назарова та ін.; за ред. В.С. Пономаренка. – Київ: Академія, 2008. – 542с.

11. Інформаційні системи та їх роль в управлінні економікою [Електронний ресурс]. – Режим доступу: http://www.uzhnu.edu.ua/uk/ infocentre/get/6742.

12. Калина И.Т. Новые информационные технологии – математическая основа / И.Т. Калина. – Одеса: Одеська нац. акад. зв'язку, 2009, № 2, С. 108−111.

13. Литвин І.С. Інформаційні технології в економіці / І.С. Литвин. – Тернопіль: навч. посібник "Економічна думка", 2011. – 296 с. – ISBN 5-7763-0459.

14. Огляд CMS. Сайт про системи управління сайтом. [Електронний ресурс]. – Режим доступу: http://www.cmslist.ru.

15. Олійник А. В. , Інформаційні системи і технології у фінансових установах : навч. посіб. / А. В. Олійник , В. М. Шацька. – Л.: Новий Світ-2006, 2008. – 436 с.

16. Організаційна структура центру зайнятості [Електронний ресурс]. – Режим доступу: http://studcon.org/organizaciyna-struktura-i-funkciyi-centru-zaynyatosti?page=1.

17. Особенности клиент-сервер [Електронний ресурс]. – Режим доступу: http://citforum.ck.ua/programming/application/builder_cs3.shtml.

18. Офіційний сайт компанії Microsoft: [Електронний ресурс]. – Режим доступу: http://support.microsoft.com/.

19. Про державну службу зайнятості [Електронний ресурс] : Постанова Кабінету Міністрів України від 24.06.91 № 47 [зі змін. та доп.]. – Режим доступу : http://zakon.rada.gov.ua/cgi-bin/laws/main.cgi?nreg=47-14.

20. Розробка ІС [Електронний ресурс]. – Режим доступу: https://xreferat.com/33/6794-3-razrabotka-informacionnoiy-sistemy-sluzhba-zanyatosti.html.

21. Сендзюк М. А. Інформаційні системи в державному управлінні : навч. посіб. / М. А. Сендзюк. – К. : КНЕУ, 2010. – 352 с.

22. Сивец С.Д. Непрерывное образование: концепция и ее реализация [Електронний ресурс]: Центр дистанционного образования Элитариум. Санкт-Петербург, 2009. – Режим доступу : http://www.elitarium.ru /2009/09/13/ nepreryvnoe_obrazovanie_koncepcija_i_ee_realizacija.html.

23. CMSогляд[Електроннийресурс].–Режимдоступу:

24. Топ-5 наиболее популярных CMS [Електронний ресурс]. – Режим доступу: https://ru/post/151879/.

25. Хансен Г.Дж. Базы данных: разработка и управление / Г.Дж. Хансен. – М. : БИНОМ, 2010. – 704 с. – ISBN 5-7989-0015-0.

26. Чмирь И.А. Объектно-ориентированное моделирование / И. А. Чмирь, А. Ф. Верлань. – Одесса: НАДУ. – 2009. – 243 с.

27. Троелсен Э. Язык программирования C# 2005 и платформа .NET 2.0 – 3-е изд. / Э. Троелсен. – М. : "Вильямс", 2008. – 1168 с.

28. Brusilovsky, P. Adaptive and intelligent Web-based educational systems / P. Brusilovsky, C. Peylo / Pittsburgh: International Journal of Artificial Intelligence in Education. Special Issue on Adaptive and Intelligent Web-based Educational Systems, 2011. – № 13 (2-4). – P. 159-172.

29. Brusilovsky, P. Domain, Task, and User Models for an Adaptive Hypermedia Performance Support System / P. Brusilovsky, D. W. Cooper. / San Francisco, CA: Proc. of 2009 International Conference on Intelligent User Interfaces – ACM Press. 2009. – P.23-30.

30. Buchanan B. G., Shortliffe E. H., The MYCIN Experiments of the Stanford Heuristic Programming Project. / B.G. Buchanan, E. H. Shortliffe. / MA: Addison-Wesley, 2008. – 769 p.

31. De Bra P. Web-based educational hypermedia / P. De Bra / Book chapter in: Data Mining / [edited by C. Romero and S. Ventura]. – Universidad de Cordoba, Spain: WIT Press., 2010. – P. 3-17.– ISBN 1-84564-152.

32. Marshall B. Convergence of Knowledge Management : the GetSmart Experience [Electronic resource]. – Houston: 2011. – Access mode: http://ai.bpa.arizona.edu/go/intranet/ Publication/JCDL-2011-Marshall.pdf.

33. McArthur D. The roles of artificial intelligence in education: Current progress and future prospects / D. McArthur, M. Lewis, M. Bishay. / RAND, Santa Monica: 2008 CA: DRU-472-NSF.

34. Microsoft Windows 7 Professional Edition. System requirements. [Electronic resource]. – Access mode: http://www.microsoft.com/en-us/smb/ products/os/winxp/requirements.mspx

35. Murray T. R. Authoring Intelligent Tutoring Systems: An Analysis of the State of the Art / T.R. Murray. / New York: International Journal of Artificial Intelligence in Education: 2008.– № 10 – P. 98-129.

36. Pathak, S. Assessing Student Programming Knowledge with Webbased Dynamic Parameterized Quizzes / S. Pathak, P. Brusilovsky / ED-MEDIA'2009 - World Conference on Educational Multimedia, Hypermedia and Telecommunications, Denver, CO, June 24-29, 2009 [Barker P and Rebelsky S (eds)]. – AACE, 2009. – P. 1548-1553.

37. Shute, V. Large-scale evaluation of an intelligent discovery world: SMITHTOWN / V. Shute, R. Glaser / Interactive Learning Environments. – 2008. – № 1. – P. 51-77.

38. Stankov, S. Ontology as a Foundation for Knowledge Evaluation in Intelligent E-learning Systems / S. Stankov, B. Žitko, A. Grubišić. / Amsterdam, Netherlands: International Workshop on Applications of Semantic Web Technologies for E-Learning (SW-EL'05) in conjunction with 12th International Conference on Artificial Intelligence in Education (AI-ED 2012), 2012. – P. 81-84.

39. Stash, N. Authoring of Learning Styles in Adaptive Hypermedia: Problems and Solutions / N. Stash, A. Cristea, P. De Bra / Proc. of The 13th International World Wide Web Conference (Alternate track papers and posters). – ACM Press, 2011. – P. 114-123.

40. Vassileva, J. A task-centered approach for user modeling in a hypermedia office documentation system / J. Vassileva. / User Modeling and User-Adapted Interaction , 2009 – № 6 (2-3). – P. 185-223.

41. Visual Studio [Electronic resource]. – Access mode: https://www. visualstudio.com/en-us/productinfo/vs2008-sysrequirements-vs.

ДОДАТКИ

Додаток А

SUMMARY

Kharchenko V.I. Development of the prototype of information system for employment centers’ clients servicing. – Qualification master's work. Sumy State University, Sumy, 2018

The paper deals with automation of the employment centers' activities. The activity of the Sumy City Employment Center has been analyzed, business process models have been developed, and a web-based information system for servicing the employment center has been designed and implemented.

Keywords: automation, web-oriented information system, ASP.NET, Visual Studio, employment center, business processes.

АНОТАЦІЯ

Харченко В.І. Розробка прототипу інформаційної системи обслуговування клієнтів центру зайнятості. – Кваліфікаційна магістерська робота. Сумський державний університет, Суми, 2018 р.

У роботі досліджені питання автоматизації діяльності центрів зайнятості. Проаналізована діяльність ДУ "Сумський міський центр зайнятості", побудовано моделі бізнес-процесів, спроектована та реалізована веб-орієнтована інформаційна система обслуговування клієнтів центру зайнятості.

Ключові слова: автоматизація, веб-орієнтована інформаційна система, ASP.NET, Visual Studio, центр зайнятості, бізнес-процеси.

Додаток Б

Лістинг Б.1 – Код Transact-SQL створення бази даних, таблиць та звязків між ними

```
Create table Адрес
(
АдресИдн int IDENTITY(1,1) not null
Constraint ПК_АдресИдн
primary key clustered,
СтранаИдн int not null,
Улица nvarchar(50) not null,
Дом int not null,
Город nvarchar(50) not null,
Область nvarchar(50)
)
CREATE TABLE Работодатель
(
РаботодательИдн int not null
Constraint ПК_РаботодательИдн
primary key clustered,
НазваниеФирмы varchar(MAX) not null,
Телефон int not null,
АдресИдн int not null

CONSTRAINT FK_Работодатель_Адрес
FOREIGN KEY (АдресИдн)
REFERENCES Адрес(АдресИдн)
)
CREATE TABLE Трудоустраиваемый
(
ТрудоустраиваемыйИдн int not null
Constraint ПК_ТрудоустраиваемыйИдн
primary key clustered,
Фамилия nvarchar(50) Not null,
Имя nvarchar(50) Not null,
Отчество nvarchar(50),
Професия varchar(max) not null,
ДатаРождения date not null,
Телефон int not null,
МестоРаботы varchar(max),
ОбщийСтаж int,
СтажЗаФахом int,
ЖелаемаяДолжность varchar(max) not null,
ЖелаемаяЗарплата money not null,
Образование varchar(max) not null,
ВладениеЯзыками varchar(max),
ВодительскиеПрава varchar(max),
АдресИдн int not null

CONSTRAINT FK_Трудоустраиваемый_адрес
FOREIGN KEY (АдресИдн)
REFERENCES Адрес(АдресИдн)
```

```
)
CREATE TABLE Картотека
(
НомерИдн int not null
Constraint ПК_НомерИдн
primary key clustered,
ДатаЗавершения date not null
)

CREATE TABLE ЗаявкаТрудоустраиваемого
(
ЗаявкаТрудоустраиваемогоИдн int not null
Constraint ПК_ЗаявкаТрудоустраиваемогоИдн
primary key clustered,
ДатаЗаявкиТрудоустраиваемого date not null
DEFAULT GETDATE(),
Оплата money not null,
ГрафикРаботы varchar(50) not null,
ТрудоустраиваемыйИдн int not null,
НомерИдн int not null

CONSTRAINT FK_ЗаявкаТрудоустраиваемого_Картотека
FOREIGN KEY (НомерИдн)
REFERENCES Картотека(НомерИдн),

CONSTRAINT FK_ЗаявкаТрудоустраиваемого_Трудоустраиваемый
FOREIGN KEY (ТрудоустраиваемыйИдн)
REFERENCES Трудоустраиваемый(ТрудоустраиваемыйИдн)
)

CREATE TABLE ЗаявкаРаботодателя
(
ЗаявкаРаботодателяИдн int not null
Constraint ПК_ЗаявкаРаботодателяИдн
primary key clustered,
ДатаЗаявкиРаботодателя date not null
DEFAULT GETDATE(),
Должность varchar(max) not null,
Требования varchar(max) not null,
Пол bit,
Возраст int,
ОпытРаботы varchar(max),
Оплата money not null,
РаботодательИдн int,
НомерИдн int not null

CONSTRAINT FK_ЗаявкаРаботодателя_Картотека
FOREIGN KEY (НомерИдн)
REFERENCES Картотека(НомерИдн),

CONSTRAINT FK_ЗаявкаРаботодателя_Работодатель
FOREIGN KEY (РаботодательИдн)
REFERENCES Работодатель(РаботодательИдн)
)
```

Додаток В

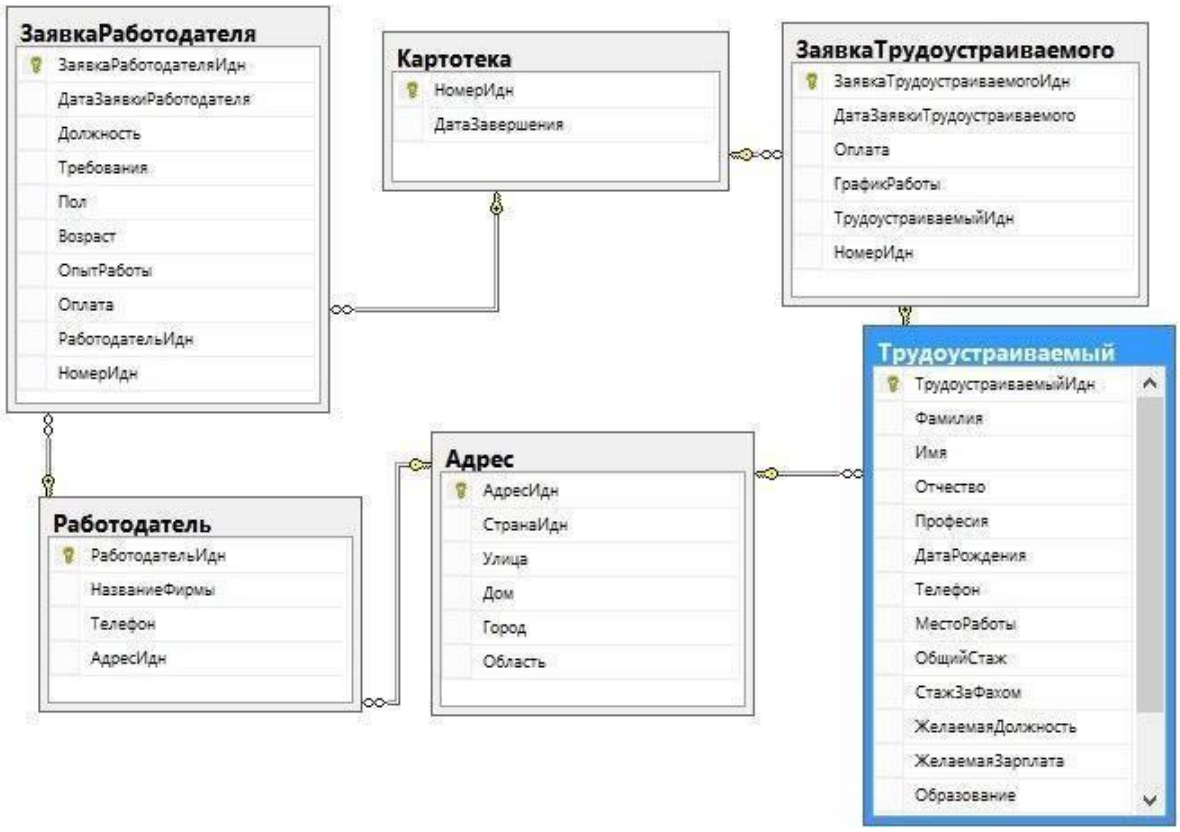


Рисунок В.1 – Діаграма зв’язків між базовими таблицями

Додаток Г

Лістинг Г.1 – HTML-код майстер-сторінки

```
<%@ Master Language="C#" AutoEventWireup="true"
CodeBehind="MasterPage.Master.cs" Inherits="WebSite.MasterPage" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >

<head id="Head1" runat="server">
    <link id="Link1" rel="stylesheet" type="text/css"
href="style.css" runat="server" />
    <title>Центр зайнятості</title>
</head>

<body>
    <form id="form1" runat="server">
        <div class="shadow" id="header">
            <div id="datetime">
                <asp:Label ID="lblTime" runat="server"
Text="datetime"></asp:Label>
            </div>
            <div id="title"><img src="Images/img1.png" alt=""
height="100%" width="auto"/></div>
        </div><!-- /#header-->

        <div id="topmenu">
            <asp:Menu ID="Menu1" runat="server" Width="480px"
                DataSourceID="SiteMapDataSource1"
                Orientation="Horizontal" StaticDisplayLevels="2">
                <StaticMenuItemStyle CssClass="staticMenuItemStyle" />
                <DynamicMenuItemStyle CssClass="dynamicMenuItemStyle" />
            </asp:Menu>
        </div><!-- /#topmenu-->

        <div id="main">
            <div class="shadow" id="leftmenu">
                <div class="menucat">~ Меню ~</div>
                <asp:Button ID="Main" CssClass="btnmenu"
Text="Головна" OnClick="MenuButton_Click" runat="server"/>
                <asp:Button ID="About" CssClass="btnmenu" Text="Про
нас" OnClick="MenuButton_Click" runat="server"/>
                <asp:Button ID="Contacts" CssClass="btnmenu"
Text="Контакти" OnClick="MenuButton_Click" runat="server"/>
                <asp:Button ID="FeedBack" CssClass="btnmenu"
Text="Залишити відгук" OnClick="MenuButton_Click" runat="server"/>

                <div class="menucat">~ Служби веб-сайту ~</div>
                <asp:Button ID="Diagnostics" CssClass="btnmenu"
Text="Діагностика" OnClick="MenuButton_Click" runat="server"/>

                <div class="menucat">~ Співробітники ~</div>
                <asp:Button ID="Employee" CssClass="btnmenu"
Text="Список співробітників" OnClick="MenuButton_Click" runat="server"/>

                <div class="menucat">~ Клієнти ~</div>
```

```
                <asp:Button ID="ClientOrders" CssClass="btnmenu"
Text="Заявки безробітних" OnClick="MenuButton_Click" runat="server"/>
                <asp:Button ID="EnterpriseOrders" CssClass="btnmenu"
Text="Заявки підприємства" OnClick="MenuButton_Click" runat="server"/>
                <asp:Button ID="Contracts" CssClass="btnmenu" Text="Договори
з працевлаштування" OnClick="MenuButton_Click" runat="server"/>

                <div class="menucat">~ Довідники ~</div>
                <asp:Button ID="DriverLicense" CssClass="btnmenu"
Text="Посвідчення водія" OnClick="MenuButton_Click" runat="server"/>

                <div id="userstat">
                    <asp:Label ID="lblUsers" runat="server"
Text="Відвідувачі:" Font-Bold="true"></asp:Label><br/>
                    <asp:Label ID="lblUserInfo" runat="server"
Text="Label" style="font-size: 13px;"></asp:Label>
                </div>
            </div><!-- /#leftmenu-->
            <div class="shadow" id="content">
                <asp:ContentPlaceHolder ID="PageContent"
runat="server"></asp:ContentPlaceHolder>
                <div id="foother">
                    Copyrights © 2013 Центр зайнятості
                </div><!-- /#foother-->
            </div><!-- /#content-->
        </div><!-- /#main-->
        <asp:SiteMapDataSource ID="SiteMapDataSource1" runat="server"
    /> </form>
</body>
</html>
```

Лістинг Г.2 – Програмний код майстер-сторінки

```
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Xml.Linq;

namespace WebSite
{
    public partial class MasterPage :
    System.Web.UI.MasterPage {
        protected void Page_Load(object sender, EventArgs e)
        {
            lblTime.Text = "Дата: " + DateTime.Now.ToLongDateString() +
"<br/>Час: " + DateTime.Now.ToLongTimeString();

            string sUserCount = "На сайті " +
 Application["UserCount"].ToString() + " відвідувача(ів).";
            string sUserNumber = "Ви відвідувач № "
+ Session["UserNumber"].ToString();
            lblUserInfo.Text = sUserCount + "<br/>" + sUserNumber;
```

```
        }

        protected void MenuButton_Click(object sender, EventArgs e)
        {
            Button btn = (Button)sender;
            Response.Redirect(btn.ID + ".aspx");
        }
    }
}
```

ДОДАТОК Д

Лістинг Д.1 – HTML-код сторінки Diagnostics.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="Diagnostics.aspx.cs" MasterPageFile="~/MasterPage.Master"
Title="Діагностика" Inherits="WebSite.Diagnostics" %>

<asp:Content ID="DefaultContent"
ContentPlaceHolderID="PageContent" runat="server">
<h4>Веб діагностика</h4>
<hr/>
<br/>
<p style="color: #fff; font-weight: bold;">Рекомендації для браузера:</p>

<asp:BulletedList ID="blBrowserReqs" runat="server"></asp:BulletedList>

<div style="margin: 15px 5px;">
    <p style="color: #fff; font-weight: bold;">Діагностика з'єднання з
базою даних:
        <asp:Label ID="LabelConnectionValue" runat="server"
Text="none" style="color: Red;"></asp:Label>
    </p>

    <p style="color: #fff; font-weight: bold;">Діагностика відловлення
помилок:</p>
    <asp:Button ID="TestError" Text="Згенерувати тестову
помилку" OnClick="TestError_Click" runat="server"/><br/>
    <asp:Button ID="Button1" Text="Згенерувати помилку
FileNotFound" OnClick="TestError404_Click" runat="server"/> </div>


</asp:Content>
```

Лістинг Д.2 – Програмний код сторінки діагностики

```
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Xml.Linq;

namespace WebSite
{
    public partial class Diagnostics :
    System.Web.UI.Page {
        protected void Page_Load(object sender, EventArgs e)
        {
            Page.Header.Title = "Загальні рекомендації для браузера";

            HttpBrowserCapabilities currentBrowser = Request.Browser;
```

```
            ListItem browserName = new ListItem
                ("Рекомендований браузер: IE 6.0 (або пізніше) ::: Ваш
браузер: " + currentBrowser.Browser + " " +
            currentBrowser.Version); ListItem browserPlatform = new
            ListItem
                ("Рекомендованнвая платформа для браузера: WinXP, або Windows
Server 2003 (або пізніше) ::: Ваша платформа: " + currentBrowser.Platform);
            ListItem browserTables = new ListItem
                ("Рекомендується підтримка таблиць ::: Підтримка таблиць
Вашим браузером: " + currentBrowser.Tables.ToString());
            ListItem browserCookies = new ListItem
                ("Рекомендується підтримка Cookie ::: Підтримка Cookie Вашим
браузером: " + currentBrowser.Cookies.ToString());
            ListItem browserVBS = new ListItem
                ("Рекомендується підтримка Visual Basic Script ::: Підтримка
Visual Basic Script Вашим браузером: " + currentBrowser.VBScript.ToString());
            ListItem browserJS = new ListItem
                ("Рекомендується підтримка ECMA Script версии 1.0
(або пізніше) ::: Версія ECMA Script Вашого браузера: " +
currentBrowser.EcmaScriptVersion.ToString());

            blBrowserReqs.Items.Add(browserName);
            blBrowserReqs.Items.Add(browserPlatform);
            blBrowserReqs.Items.Add(browserTables);
            blBrowserReqs.Items.Add(browserCookies);
            blBrowserReqs.Items.Add(browserVBS);
            blBrowserReqs.Items.Add(browserJS);

            string DBMsg = "";
            try
            {
                string dbStatus =
ConfigurationManager.ConnectionStrings["WebSiteDB"].ConnectionString;
                System.Data.SqlClient.SqlConnection sqlConn =
new System.Data.SqlClient.SqlConnection(dbStatus);

                sqlConn.Open();

                if (sqlConn.State.ToString() == "Open")
                    DBMsg = "База даних доступна.";

                sqlConn.Close();
            }
            catch (Exception ex)
            {
                Console.WriteLine("Error: " + ex.ToString());
                DBMsg = "База даних тимчасово недоступна.";
            }
            finally
            {
                LabelConnectionValue.Text = DBMsg;
            }
        }

        protected void TestError_Click(object sender, EventArgs e)
        {
            Exception customException = new Exception("Test
            Error"); throw customException;
        }

        protected void TestError404_Click(object sender, EventArgs e)
        {
            Response.Redirect("Test.aspx");
        }
    }
}
```

ДОДАТОК Е

Лістинг Е.1 – HTML-код сторінки DriverLicense.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="DriverLicense.aspx.cs" MasterPageFile="~/MasterPage.Master"
Inherits="WebSite.DriverLicense" %>

<asp:Content ID="DefaultContent"
ContentPlaceHolderID="PageContent" runat="server">
<div class="pagetitle">
    <p>Посвідчення водія</p>
    <hr/>
</div>

    <div>
        <asp:Label ID="lblDriverLicenseType" runat="server"
Text="Види посвідчень:"></asp:Label><br/><br/>
        <asp:GridView ID="gvDriverLicenseType" runat="server"
AllowPaging="True"
            AllowSorting="True" AutoGenerateColumns="False" CellPadding="4"
            DataKeyNames="ПосвідченняВодіяІдн"
DataSourceID="SqlDriverLicenseDataSource"
            ForeColor="#333333" Width="50%"
            RowStyle-HorizontalAlign="Center"

onselectedindexchanged="gvDriverLicenseType_SelectedIndexChanged"
> <RowStyle HorizontalAlign="Center" BackColor="#FFFBD6"
ForeColor="#333333"></RowStyle>
            <Columns>
                <asp:BoundField DataField="ПосвідченняВодіяІдн"
HeaderText="№"
                    InsertVisible="False" ReadOnly="True"
SortExpression="ПосвідченняВодіяІдн" />
                <asp:BoundField DataField="Категорія" HeaderText="Категорія"
                    SortExpression="Категорія" />
            </Columns>
            <FooterStyle BackColor="#990000" ForeColor="White"
Font-Bold="True" />
            <PagerStyle BackColor="#FFCC66" ForeColor="#333333"
HorizontalAlign="Center" />
            <SelectedRowStyle BackColor="#FFCC66" Font-
Bold="True" ForeColor="Navy" />
            <HeaderStyle BackColor="#990000" Font-Bold="True"
ForeColor="White" />
            <AlternatingRowStyle BackColor="White"
        /> </asp:GridView>

        <asp:Panel ID="PanelNewDriverLicenseType"
Visible="false" style="margin:30px 0;" runat="server">
            <asp:Label ID="lblNewDriverLicenseType"
runat="server" Text="Назва категорії:"></asp:Label><br />
            <input id="tbNewDriverLicenseType" type="text" runat="server" />
            <asp:Button ID="btnAddChangeDriverLicenseType" runat="server"
Text="Додати"
        onclick="AddChage_DriverLicenseType_Click"/>
        </asp:Panel>
    </div>
    <asp:Panel ID="PanelButtons" style="width: 100%; margin-top: 20px;
height: 150px;" runat="server">
```

```
        <asp:Button ID="btnAddDriverLicenseType" runat="server" Text="Додати"
style="width: 100px; height: 40px;" onclick="Add_DriverLicenseType_Click"/>
        <asp:Button ID="btnChangeDriverLicenseType"
runat="server" Text="Змінити" style="width: 100px; height: 40px;"
onclick="Change_DriverLicenseType_Click"/>
        <asp:Button ID="btnRemoveDriverLicenseType"
runat="server" Text="Видалити" style="width: 100px; height:
40px;" onclick="Remove_DriverLicenseType_Click"/>
    </asp:Panel>

    <asp:SqlDataSource ID="SqlDriverLicenseDataSource" runat="server"
        ConnectionString="<%$ ConnectionStrings:WebSiteDB %>"
        SelectCommand="SELECT * FROM [ПосвідченняВодія]"></asp:SqlDataSource>
</asp:Content>
```

Лістинг Е.2 – Програмний код сторінки DriverLicense.aspx.cs

```
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Xml.Linq;
using System.Data.Sql;
using System.Data.SqlClient;


namespace WebSite
{
    public partial class DriverLicense :
    System.Web.UI.Page {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void Add_DriverLicenseType_Click(object sender, EventArgs
e)
        {
            Session["ChangeState"] = 1; // Add
            PanelNewDriverLicenseType.Visible =
            true; PanelButtons.Visible = false;
            tbNewDriverLicenseType.Focus();
        }

        protected void Change_DriverLicenseType_Click(object
sender, EventArgs e)
        {
            Session["ChangeState"] = 2; // Change
            gvDriverLicenseType.AutoGenerateSelectButton = true;
        }

        protected void Remove_DriverLicenseType_Click(object
sender, EventArgs e)
```

```
        {
            Session["ChangeState"] = 3; // Remove
            gvDriverLicenseType.AutoGenerateSelectButton = true;
        }

        protected void AddChage_DriverLicenseType_Click(object
sender, EventArgs e)
        {
            SqlConnection connection = new
            SqlConnection(); connection.ConnectionString =
System.Configuration.ConfigurationManager.ConnectionStrings["WebSiteDB"].Conn
ectionString;

            string id = "";
            if (Session["ChangeState"].ToString() == "2")
                id = Session["ChangeID"].ToString();

            string sqlCmd = (Session["ChangeState"].ToString() == "2"
? "ЗмінитиПосвідченняВодія @ПосвідченняВодіяІдн = " + id + "," :
"ДодатиПосвідченняВодія");

            try
            {
                connection.Open();
                SqlCommand cmd = new SqlCommand(@"" + sqlCmd + " @Назва =
'" + tbNewDriverLicenseType.Value.ToString() + "'", connection);

                cmd.ExecuteNonQuery();
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex);
            }
            finally
            {
                Response.Redirect("DriverLicense.aspx");
                connection.Close();
            }
        }

        protected void
gvDriverLicenseType_SelectedIndexChanged(object sender, EventArgs e)
        {
            gvDriverLicenseType.AutoGenerateSelectButton =
            false; int id =
int.Parse(gvDriverLicenseType.Rows[gvDriverLicenseType.SelectedIndex].Cells[1 ].Text);


            Session["ChangeID"] = id;
            if (Session["ChangeState"].ToString() == "1")
            {
                lblNewDriverLicenseType.Text = "Назва
                категорії:"; btnAddChangeDriverLicenseType.Text =
                "Додати"; tbNewDriverLicenseType.Focus();
                PanelNewDriverLicenseType.Visible = true;
                PanelButtons.Visible = false;
            }
            else
                if (Session["ChangeState"].ToString() == "2")
                {
                    lblNewDriverLicenseType.Text = "Нова назва
                    категорії:"; btnAddChangeDriverLicenseType.Text =
                    "Змінити"; tbNewDriverLicenseType.Value =
gvDriverLicenseType.Rows[gvDriverLicenseType.SelectedIndex].Cells[2].Text;
```

```
                    tbNewDriverLicenseType.Focus();
                    PanelNewDriverLicenseType.Visible =
                    true; PanelButtons.Visible = false;
                }
                else
                {
                    SqlConnection connection = new
                    SqlConnection(); connection.ConnectionString =
System.Configuration.ConfigurationManager.ConnectionStrings["WebSiteDB"].Conn
ectionString;

                    try
                    {
                        connection.Open();
                        SqlCommand cmd = new
SqlCommand(@"ВидалитиПосвідченняВодія @ПосвідченняВодіяІдн = "
+ id.ToString(), connection);

                        cmd.ExecuteNonQuery();
                    }
                    catch (Exception ex)
                    {

                    }
                    finally
                    {
                        connection.Close();
                    }
                }
            }
        }
    }
```

ДОДАТОК Ж

Лістинг Ж.1 – HTML-код сторінки AddClientOrder.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="AddClientOrder.aspx.cs" MasterPageFile="~/MasterPage.Master"
Inherits="WebSite.AddClientOrder" %>

<asp:Content ID="DefaultContent" ContentPlaceHolderID="PageContent"
runat="server">
<style>.horsec{margin:10px;width:100%;border-bottom: solid 1px #ccc;padding-
bottom: 30px;float:left;clear:both;} .horitem{float:left;margin-right:30px;}
h4{margin-top: 0;}</style>
<div class="pagetitle">
    <p>Додати заявку безробітного</p>
    <hr/>
</div>

<div class="pagetext">
      <div id="HumanParams" class="horsec">
            <h4>Безробітний</h4>
            <div class="horitem">
                  <asp:CheckBox ID="cbIsMen" runat="server"
                  /> <asp:Label ID="lblIsMen" runat="server"
Text="Чоловік"></asp:Label>
            </div>

            <div class="horitem">
                  <asp:Label ID="lblFirstName" runat="server"
Text="Прізвище:"></asp:Label><br />
                  <input id="tbFirstName" type="text" runat="server"
            /> </div>

            <div class="horitem">
                  <asp:Label ID="lblLastName" runat="server"
Text="Ім'я:"></asp:Label><br />
                  <input id="tbLastName" type="text" runat="server"
            /> </div>

            <div class="horitem">
                  <asp:Label ID="lblMiddleName" runat="server" Text="По
батькові:"></asp:Label><br />
                  <input id="tbMiddleName" type="text" runat="server"
            /> </div>

            <div class="horitem">
                  <asp:Label ID="lblBornDate" runat="server" Text="Дата
народження:"></asp:Label><br />
                  <input id="tbBornDate" type="text"
            runat="server"/> </div>

            <div class="horitem">
                  <asp:Label ID="lblPhoneNumber" runat="server"
Text="Телефон:"></asp:Label><br/>
                  <input id="tbPhoneNumber" type="text"
            runat="server"/> </div>
      </div>

      <div id="PassportParams" class="horsec">
```

```
            <h4>Паспортні дані</h4>
            <div class="horitem">
                  <asp:Label ID="lblPassportNum" runat="server" Text="Номер
паспорта:"></asp:Label><br />
                  <input id="tbPassportNum" type="text" runat="server"
            /> </div>

            <div class="horitem">
                  <asp:Label ID="lblPassportSer" runat="server" Text="Серія
паспорта:"></asp:Label><br />
                  <input id="tbPassportSer" type="text" runat="server"
            /> </div>

            <div class="horitem">
                  <asp:Label ID="lblPassportVud" runat="server" Text="Ким
виданий паспорт:"></asp:Label><br />
                  <input id="tbPassportVud" type="text" runat="server"
            /> </div>
      </div>

      <div id="AddressParams" class="horsec">
            <h4>Адреса</h4>
            <div class="horitem">
                  <asp:Label ID="lblRegion" runat="server"
Text="Область:"></asp:Label><br />
                  <input id="tbRegion" type="text" runat="server"/>
            </div>

            <div class="horitem">
                  <asp:Label ID="lblCity" runat="server"
Text="Місто:"></asp:Label><br />
                  <input id="tblCity" type="text" runat="server"/>
            </div>

            <div class="horitem">
                  <asp:Label ID="lblAddress" runat="server"
Text="Вулиця:"></asp:Label><br />
                  <input id="tblAddress" type="text" runat="server"/>
            </div>

            <div class="horitem">
                  <asp:Label ID="lblBuilding" runat="server"
Text="Будинок:"></asp:Label><br />
                  <input id="tbBuilding" type="text" runat="server" />
            </div>

            <div class="horitem">
                  <asp:Label ID="lblFlat" runat="server"
Text="Квартира:"></asp:Label><br />
                  <input id="tbFlat" type="text" runat="server"/>
            </div>

            <div class="horitem">
                  <asp:Label ID="lblIndex" runat="server"
Text="Індекс:"></asp:Label><br />
                  <input id="tbIndex" type="text"
            runat="server"/> </div>
      </div>

      <div id="WorkParams" class="horsec">
            <h4>Досвід роботи</h4>
            <div class="horitem">
                  <asp:Label ID="lblPost" runat="server"
Text="Професія:"></asp:Label><br />
```

```
                  <input id="tbPost" type="text" runat="server"/>
            </div>

            <div class="horitem">
                  <asp:Label ID="lblPlaceOfWork" runat="server" Text="Останнє
місце роботи:"></asp:Label><br />
                  <input id="tbPlaceOfWork" type="text" runat="server"/>
            </div>

            <div class="horitem">
                  <asp:Label ID="lblTotalExp" runat="server" Text="Загальний
стаж роботи:"></asp:Label><br />
                  <input id="tbTotalExp" type="text" runat="server"/>
            </div>

            <div class="horitem">
                  <asp:Label ID="lblProfExp" runat="server" Text="Стаж роботи
за фахом:"></asp:Label><br />
                  <input id="tbProfExp" type="text" runat="server"/>
            </div>
      </div>

      <div id="FutureWorkParams" class="horsec">
            <h4>Бажана робота</h4>
            <div class="horitem">
                  <asp:Label ID="lblDesiredPost" runat="server"
Text="Посада:"></asp:Label><br />
                  <input id="tbDesiredPost" type="text" runat="server"/>
            </div>

            <div class="horitem">
                  <asp:Label ID="lblDesiredSalary"
runat="server" Text="Зарплата:"></asp:Label><br />
                  <input id="tbDesiredSalary" type="text" runat="server"/>
            </div>
      </div>

      <div id="SkillsParams" class="horsec">
            <h4>Навики</h4>
            <div class="horitem">
                  <asp:Label ID="lblEducation" runat="server"
Text="Освіта:"></asp:Label><br />
                  <asp:DropDownList ID="ddlEducation" runat="server"
                        DataSourceID="SqlEducationDataSource"
DataTextField="Форма"
                        DataValueField="ОсвітаІдн"> </asp:DropDownList>
                  <asp:SqlDataSource ID="SqlEducationDataSource"
runat="server"
                        ConnectionString="<%$ ConnectionStrings:WebSiteDB %>"
                        SelectCommand="SELECT * FROM
 [Освіта]"></asp:SqlDataSource>
            </div>

            <div class="horitem">
                  <asp:Label ID="lblLang" runat="server" Text="Володіння
мовами:"></asp:Label><br />
                  <input id="tbLang" type="text" runat="server"/>
            </div>

            <div class="horitem">
                  <asp:Label ID="lblDriver" runat="server" Text="Посвідчення
водія:"></asp:Label><br />
                  <asp:DropDownList ID="ddlDriver" runat="server"
```

```
                        DataSourceID="SqlDriverDataSource"
DataTextField="Категорія"

      DataValueField="ПосвідченняВодіяІдн"></asp:DropDownList>
                  <asp:SqlDataSource ID="SqlDriverDataSource" runat="server"
                        ConnectionString="<%$ ConnectionStrings:WebSiteDB %>"
                        SelectCommand="SELECT * FROM
[ПосвідченняВодія]"></asp:SqlDataSource>
            </div>

            <div class="horitem">
                  <asp:CheckBox ID="cbForeignPassport" runat="server"
                  /> <asp:Label ID="lblForeignPassport" runat="server"
Text="Закордонний паспорт:"></asp:Label>
            </div>
      </div>

      <asp:Panel ID="PanelButtons" class="horsec"
style="height:150px;border:none;" runat="server">
        <asp:Button ID="btnAddClientOrder" runat="server" Text="Додати"
style="width: 100px; height: 40px;" onclick="Add_ClientOrder_Click"/>
    </asp:Panel>
</div>
</asp:Content>
```

Лістинг Ж.2 – Програмний код сторінки AddClientOrder.aspx.cs

```
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Xml.Linq;
using System.Data.Sql;
using System.Data.SqlClient;

namespace WebSite
{
    public partial class AddClientOrder :
    System.Web.UI.Page {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void Add_ClientOrder_Click(object sender, EventArgs e)
        {
            SqlConnection connection = new
            SqlConnection(); connection.ConnectionString =
System.Configuration.ConfigurationManager.ConnectionStrings["WebSiteDB"].Conn
ectionString;

            try
            {
```

```
                connection.Open();
                SqlCommand cmd = new SqlCommand(@"ДодатиКлієнта @Прізвище
= '" + tbLastName.Value.ToString() +
                                                "', @Ім_я = '" +
tbFirstName.Value.ToString() +
                                                "', @Побатькові = '" +
tbMiddleName.Value.ToString() +
                                                "', @ДатаНародження = '" +
tbBornDate.Value.ToString() +
                                                "', @НомерТелефону = " +
tbPhoneNumber.Value.ToString() +
                                                ", @НомерПаспорту = " +
tbPassportNum.Value.ToString() +
                                                ", @СеріяПаспорту = '" +
tbPassportSer.Value.ToString() +
                                                "', @КимВиданий = '" +
tbPassportVud.Value.ToString() +
                                                "', @Область = '" +
tbRegion.Value.ToString() +
                                                "', @Місто = '" +
tblCity.Value.ToString() +
                                                "', @Вулиця = '" +
tblAddress.Value.ToString() +
                                                "', @Будинок = " +
tbBuilding.Value.ToString() +
                                                ", @Квартира = " +
tbFlat.Value.ToString() +
                                                ", @Індекс = " +
tbIndex.Value.ToString() +
                                                ", @СтатьЧоловіча = " +
((cbIsMen.Checked) ? "1" : "0") +
                                                  ", @Професія = '" +
tbPost.Value.ToString() +
                                                "', @МісцеРоботи = '" +
tbPlaceOfWork.Value.ToString() +
                                                "', @ЗагальнийСтажРоботи = " +
int.Parse(tbTotalExp.Value.ToString()) +
                                                ", @СтажРоботиЗаФахом = " +
int.Parse(tbProfExp.Value.ToString()) +
                                                ", @БажанаПосада = '" +
tbDesiredPost.Value.ToString() +
                                                "', @БажанаЗарплата = " +
tbDesiredSalary.Value.ToString() +
                                                ", @ДипломПроОсвітуІдн = " +
int.Parse(ddlEducation.SelectedItem.Value.ToString()) +
                                                ", @ВолодінняМовами = '" +
tbLang.Value.ToString() +
                                                "', @ПосвідченняВодіяІдн = "
+ int.Parse(ddlDriver.SelectedItem.Value.ToString()) +
                                                ",
@НаявністьЗакордонногоПаспорту = " + ((cbForeignPassport.Checked) ? "1" :
"0"), connection);

                cmd.ExecuteNonQuery();
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex);
            }
            finally
            {
                connection.Close();
                Response.Redirect("ClientOrders.aspx");
            }
        }

    }
}
```

Додаток З

Лістинг 3.1 – HTML-код сторінки AddEmployerOrder.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="AddEmployerOrder.aspx.cs"
MasterPageFile="~/MasterPage.Master" Inherits="WebSite.AddClientOrder" %>

<asp:Content ID="DefaultContent" ContentPlaceHolderID="PageContent"
runat="server">
<style>.horsec{margin:10px;width:100%;border-bottom: solid 1px #ccc;padding-
bottom: 30px;float:left;clear:both;} .horitem{float:left;margin-right:30px;}
h4{margin-top: 0;}</style>
<div class="pagetitle">
    <p>Додати заявку роботодавця</p>
    <hr/>
</div>

<div class="pagetext">
      <div id="HumanParams" class="horsec">
            <h4>Роботодавець</h4>
            <div class="horitem">
                  <asp:CheckBox ID="cbIsMen" runat="server"
                  /> <asp:Label ID="lblIsMen" runat="server"
Text="Назва фірми:"></asp:Label><br />
                  <input id="tbFirstName" type="text" runat="server"
            /> </div>

            <div class="horitem">
                  <asp:Label ID="lblLastName" runat="server"
Text="Телефон:"></asp:Label><br/>
                  <input id="tbPhoneNumber" type="text"
            runat="server"/> </div>
      </div>

      <div id="PassportParams" class="horsec">

      <div id="AddressParams" class="horsec">
            <h4>Адреса</h4>
            <div class="horitem">
                  <asp:Label ID="lblRegion" runat="server"
Text="Область:"></asp:Label><br />
                  <input id="tbRegion" type="text" runat="server"/>
            </div>

            <div class="horitem">
                  <asp:Label ID="lblCity" runat="server"
Text="Місто:"></asp:Label><br />
                  <input id="tblCity" type="text" runat="server"/>
            </div>

            <div class="horitem">
                  <asp:Label ID="lblAddress" runat="server"
Text="Вулиця:"></asp:Label><br />
                  <input id="tblAddress" type="text" runat="server"/>
            </div>

            <div class="horitem">
```

```
                  <asp:Label ID="lblBuilding" runat="server"
Text="Будинок:"></asp:Label><br />
                  <input id="tbBuilding" type="text" runat="server"
            /> </div>

            <div class="horitem">
                  <asp:Label ID="lblFlat" runat="server"
Text="Квартира:"></asp:Label><br />
                  <input id="tbFlat" type="text" runat="server"/>
            </div>

            <div class="horitem">
                  <asp:Label ID="lblIndex" runat="server"
Text="Індекс:"></asp:Label><br />
                  <input id="tbIndex" type="text"
            runat="server"/> </div>
      </div>

      <div id="WorkParams" class="horsec">

      <asp:Panel ID="PanelButtons" class="horsec"
style="height:150px;border:none;" runat="server">
        <asp:Button ID="btnAddClientOrder" runat="server" Text="Додати"
style="width: 100px; height: 40px;" onclick="Add_ClientOrder_Click"/>
    </asp:Panel>
</div>
</asp:Content>
```

Лістинг 3.2 – Програмний код сторінки AddEmployerOrder.aspx.cs

```
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Xml.Linq;
using  System.Data.Sql;
using System.Data.SqlClient;

namespace WebSite
{
    public partial class AddEmployerOrder :
    System.Web.UI.Page {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void Add_EmployerOrder_Click(object sender, EventArgs
        e) {
            SqlConnection connection = new
            SqlConnection(); connection.ConnectionString =
System.Configuration.ConfigurationManager.ConnectionStrings["WebSiteDB"].Conn
ectionString;
```

```
            try
            {
                connection.Open();
                SqlCommand cmd = new SqlCommand(@"ДодатиФірму @НазваФірми
= '" + tbName.Value.ToString() +
                                                "', @НомерТелефону = " +
tbPhoneNumber.Value.ToString() +
                                                "', @Область = '" +
tbRegion.Value.ToString() +
                                                "', @Місто = '" +
tblCity.Value.ToString() +
                                                "', @Вулиця = '" +
tblAddress.Value.ToString() +
                                                "', @Будинок = " +
tbBuilding.Value.ToString() +
                                                ", @Квартира = " +
tbFlat.Value.ToString() +
                                                ", @Індекс = " +
tblCity.Value.ToString() +
                                                "', @Вимоги = '" +
tblNeeds.Value.ToString() +
                                                "', @Стать = '" +
tblSt.Value.ToString() +
                                                "', @Вік = '" +
tblOld.Value.ToString() +
                                                "', @Оплата = '" +
tbPay.Value.ToString() +
                                                ", @Посада = '" +
tbStat.Value.ToString() +
                                                "', @ЗагальнийСтажРоботи = " +
int.Parse(tbTotalExp.Value.ToString()) +

                cmd.ExecuteNonQuery();
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex);
            }
            finally
            {
                connection.Close();
                Response.Redirect("ClientOrders.aspx");
            }
        }

    }
}
```