ДЕРЖАВНИЙ ТОРГОВЕЛЬНО-ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
Кафедра комп'ютерних наук та інформаційних технологій

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
на тему:

«РОЗРОБКА АВТОМАТИЗОВАНОЇ СИСТЕМИ ПРОВЕДЕННЯ ІНСТРУКТАЖУ СПІВРОБІТНИКІВ ПІДПРИЄМСТВА»

Студента 4 курсу, 10 групи
Спеціальності 122
«Комп'ютерні науки» Карпенка Матвія Сергійовича

_____
підпис студента

Кандидат педагогічних наук, доцент Дивак Володимир Валерійович

_____
підпис керівника

Гарант освітньої програми
кандидат технічних наук, доцент Демідов Павло Георгійович

_____
підпис керівника

Київ 2023

Державний торговельно-економічний університет

Факультет інформаційних технологій
Кафедра комп'ютерних наук та систем
Спеціальність 122 «Комп'ютерні науки»

Затверджую
Зав. кафедри _____Пурський О.І.
«12» грудня 2023 р.

Завдання
на випускну кваліфікаційну роботу(проект) студенту

Карпенка Матвія Сергійовича
(прізвище, ім'я, по батькові)

1. Тема випускної кваліфікаційної роботи (проекту)
«Розробка автоматизованої системи проведення інструктажу співробітників підприємства»
Затверджена наказом ректора від *«09» грудня 2022 р. № 3332*
2. Строк здачі студентом закінченої роботи *30 травня 2023 року*
3. Цільова установка та вихідні дані до роботи
Мета роботи: розробити автоматизовану систему проведення інструктажу співробітників на підприємстві.
Об'єкт дослідження: розробка автоматизованої систему проведення інструктажу співробітників підприємства
Предмет дослідження: засоби розробки автоматизованої системи проведення інструктажу співробітників на підприємстві
4.Перелік графічного матеріалу_____
_____
_____
_____

5.Консультанти по роботі із зазначенням розділів, за якими здійснюється консультування:

Розділ	Консультант (прізвище, ініціали)	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Дивак В.В.	15.12.2022 р.	15.12.2022 р.
2	Дивак В.В.	15.12.2022 р.	15.12.2022 р.
3	Дивак В.В.	15.12.2022 р.	15.12.2022 р.

6. Зміст випускного кваліфікаційного проекту (перелік питань за кожним розділом)

ВСТУП

РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ ПРОЦЕСУ ІНСТРУКТАЖУ НА ПІДПРИЄМСТВІ

1.1. Аналіз існуючих проблем у проведенні інструктажу на підприємстві

1.2. Огляд сучасних рішень та методик автоматизації інструктування

1.3.Визначення вимог до автоматизованої системи проведення інструктажу

РОЗДІЛ 2. РОЗРОБКА АВТОМАТИЗОВАНОЇ СИСТЕМИ ПРОВЕДЕННЯ ІНСТРУКТАЖУ

2.1. Архітектура та функціональні вимоги до системи проведення інструктажу

2.2. Розробка бази даних для зберігання інформації про інструктажі та працівників

2.3. Проектування інтерфейсу користувача для зручного використання системи

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ АВТОМАТИЗОВАНОЇ СИСТЕМИ

3.1. Вибір технологій та інструментів для реалізації системи проведення інструктажу

3.2. Розробка та імплементація модулів системи проведення інструктажу

3.3. Тестування та валідація роботи системи

ВИСНОВКИ

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

№ Пор.	Назва етапів випускної кваліфікаційної роботи	Строк виконання етапів роботи	
		За планом	фактично
1	2	3	4
1	*Вибір теми випускної кваліфікаційної роботи*	*04.10.2020*	*04.10.2020*
2	*Розробка та затвердження завдання на випускну кваліфікаційну роботу*	*15.12.2022*	*15.12.2022*
3	*Вступ*	*03.02.2023*	*03.02.2023*
4	*РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ ПРОЦЕСУ ІНСТРУКТАЖУ НА ПІДПРИЄМСТВІ*	*28.02.2023*	*28.02.2023*
5	*РОЗДІЛ 2. РОЗРОБКА АВТОМАТИЗОВАНОЇ СИСТЕМИ ПРОВЕДЕННЯ ІНСТРУКТАЖУ*	*06.04.2023*	*06.04.2023*
6	*РОЗДІЛ 3. РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ АВТОМАТИЗОВАНОЇ СИСТЕМИ*	*12.05.2023*	*12.05.2023*
7	*Висновки*	*15.05.2023*	*15.05.2023*
8	*Здача випускної кваліфікаційної роботи на кафедру науковому керівнику*	*30.05.2023*	*30.05.2023*
9	*Попередній захист випускної кваліфікаційної роботи*	*31.05.2023-01.06.2023*	*31.05.2023-01.06.2023*
11	*Виправлення зауважень, зовнішнє рецензування випускної кваліфікаційної роботи*	*02.06.2023*	*02.06.2023*
12	*Представлення готової зшитої випускної кваліфікаційної роботи на кафедру*	*05.06.2023*	*05.06.2023*
13	*Публічний захист випускної кваліфікаційної роботи*	*За розкладом роботи ЕК*	*За розкладом роботи ЕК*

8. Дата видачі завдання «15» грудня 2022 р.

Керівник випускної кваліфікаційної роботи (проекту)
Дивак В. В.
(прізвище, ініціали, підпис)

Гарант освітньої програми Демідов П.Г.
(прізвище, ініціали, підпис)

Завдання прийняв студент-дипломник Карпенко М.С.

9. Відгук керівника випускної кваліфікаційної роботи (проекту)

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

Керівник випускної кваліфікаційної роботи (проекту)

_____ 30.05.2023 р.

(підпис, дата)

10. Висновок про випускну кваліфікаційну роботу

Випускна кваліфікаційна робота студента Карпенка М.С.

(прізвище, ініціали)

може бути допущена до захисту в екзаменаційній комісії.

Гарант освітньої програми _____Демідов П.Г.

(підпис, прізвище, ініціали)

Завідувач кафедри _____ Пурський О.І.

(підпис, прізвище, ініціали)

«_____»_____2023 р.

Анотація

Ця випускна кваліфікаційна робота присвячена розробці автоматизованої системи проведення інструктажу на підприємстві. Метою роботи є створення автоматизованого процесу інструктування, який забезпечить ефективне поширення інформації та полегшить організацію і контроль над інструктувальними заходами. Об'єктом дослідження є процеси інструктування на підприємстві, а предметом дослідження - засоби створення інформаційних систем та автоматизації процесів. Ця робота сприятиме покращенню ефективності та безпеки на робочому місці шляхом забезпечення належного інструктування працівників.

Наша ціль полягає в розробці автоматизованої системи проведення інструктажу, яка значно спростить і полегшить процеси організації та контролю над інструктуванням персоналу. Використовуючи інформаційні системи та автоматизацію процесів, забезпечимо ефективну комунікацію та передачу необхідної інформації всім співробітникам. Це не лише підвищить загальну ефективність роботи, але й зробить робоче місце більш безпечним для всіх. Отже, інструктування співробітників стане легшим, а організація та контроль за ним - більш ефективними.

Ключові слова: автоматизована система, інструктування, підприємство, інформаційна система, ефективність, безпека.

Abstract

This graduation thesis is devoted to the development of an automated system for conducting training at the enterprise. The goal of the work is to create an automated instructional process that will ensure effective dissemination of information and facilitate the organization and control of instructional activities. The object of the study is the process of instructing at the enterprise, and the subject of the study is the means of creating information systems and automating processes. This work will contribute to improved efficiency and safety in the workplace by ensuring that workers are properly trained.

Our goal is to develop an automated briefing system that will greatly simplify and facilitate the processes of organizing and controlling staff briefings. Using information systems and process automation, we will ensure effective communication and transfer of necessary information to all employees. This will not only increase overall work efficiency, but also make the workplace safer for everyone. Consequently, employee training will become easier, and its organization and control will become more efficient.

Keywords: automated system, instruction, enterprise, information system, efficiency, safety.

ЗМІСТ

ВСТУП

У сучасному світі безпека праці та здоров'я працівників на підприємствах є однією з найважливіших пріоритетів. Один зі способів забезпечення безпеки працівників - це проведення інструктажу, який дозволяє ознайомити співробітників з правилами та процедурами, пов'язаними з безпекою та працездатністю.

Актуальність даного дослідження полягає в необхідності покращення процесу проведення інструктажу співробітників на підприємствах. Розробка автоматизованої системи сприятиме підвищенню ефективності та якості інструктування, зменшенню ризиків для працівників та економії ресурсів підприємства.

Метою даної роботи є розробка автоматизованої системи проведення інструктажу співробітників на підприємстві. Ця система буде спрощувати та оптимізовувати процес проведення інструктажу, забезпечуючи ефективну комунікацію між керівництвом та співробітниками, а також забезпечуючи доступ до необхідної інформації та матеріалів.

Об'єктом дослідження є розробка автоматизованої системи проведення інструктажу співробітників на підприємстві.

Предметом дослідження є засоби розробки автоматизованої системи проведення інструктажу співробітників на підприємстві

Результати дослідження та розробка автоматизованої системи проведення інструктажу співробітників на підприємстві можуть бути використані практичною сферою для поліпшення безпеки та ефективності проведення інструктажу. Крім того, розроблена система може сприяти

оптимізації робочих процесів та підвищенню продуктивності працівників на підприємстві.

Для досягнення поставленої мети передусім потрібно вирішити наступні **завдання**:

1. Проаналізувати існуючі проблеми в процесі проведення інструктажу КОГО на підприємстві.

2. Вивчити сучасні рішення та методики у сфері автоматизації інструктування.

3. Визначити вимоги до автоматизованої системи проведення інструктажу.

4. Розробити архітектуру та функціональні вимоги до системи проведення інструктажу.

5. Розробити базу даних для зберігання інформації про інструктажі та працівників.

6. Проектувати інтерфейс користувача для зручного використання системи.

7. Реалізувати систему проведення інструктажу та забезпечити її тестування та валідацію.

Для досягнення поставлених завдань будуть використовуватися наступні **методи дослідження:**

1. Аналіз літературних джерел, наукових публікацій та статистичних даних щодо проблеми інструктування на підприємствах.

2. Опитування та інтерв'ювання фахівців з питань безпеки праці та проведення інструктажу.

3. Моделювання та проектування інформаційної системи проведення інструктажу.

4. Використання програмних засобів для реалізації системи проведення інструктажу.

5. Тестування та валідація роботи системи на підприємстві.

Практична складова дослідження підтвердила, що розроблена автоматизована система є ефективним інструментом для покращення процесу проведення інструктажу на підприємстві. Вона сприяє залученню працівників до навчального процесу, поліпшує засвоєння інформації та забезпечує належний рівень безпеки. Застосування автоматизованої системи проведення інструктажу дозволяє ефективно використовувати ресурси підприємства та забезпечує підвищення продуктивності та якості роботи працівників.

Для впровадження системи було проведено тестування та валідацію функціональності, а також забезпечено навчання та підтримку користувачів. Після впровадження системи була здійснена оцінка її ефективності та впливу на процес проведення інструктажу, залучення працівників та забезпечення безпеки на робочому місці.

Структура та обсяг випускної кваліфікаційної роботи. Випускна кваліфікаційна робота складається із вступу, трьох розділів, висновків, списку використаних джерел із 41 найменувань, додатків і містить 45 сторінки основного тексту, 7 рисунків і 1 таблиці.

РОЗДІЛ 1.

АНАЛІЗ ПРОБЛЕМАТИКИ ПРОЦЕСУ ІНСТРУКТАЖУ НА ПІДПРИЄМСТВІ

1.1. Аналіз існуючих проблем у проведенні інструктажу на підприємстві

У даному підрозділі розглядається пошук та аналіз наявних проблем, які виникають в процесі проведення інструктажу на підприємстві. Для цього здійснюється комплексне дослідження, яке включає такі етапи:

1.1.1. Збір інформації про поточний процес проведення інструктажу

На цьому етапі здійснюється аналіз поточного стану проведення інструктажу на підприємстві. Збирається інформація про процедури, правила, методи та інструменти, що застосовуються під час проведення інструктажу. Розглядаються документи, які регулюють цей процес, наприклад, положення про проведення інструктажу, стандарти безпеки праці, інструкції для працівників. Виявляються основні етапи і взаємозв'язки між ними, а також залучені сторони.

1.1.2. Аналіз ідентифікованих проблем

На цьому етапі виявляються та аналізуються проблеми, які можуть виникати в процесі проведення інструктажу. Це можуть бути такі проблеми, як недостатня ефективність інструктування, нечіткість інструкцій, відсутність контролю над процесом, недостатня увага до безпеки праці, недоцільне використання ресурсів і т.д. Аналіз проводиться залежно від специфіки підприємства, його виду діяльності та особливостей робочого процесу.

1.1.3. Виявлення причин виникнення проблем

На цьому етапі проводиться ідентифікація причин, що лежать в основі виявлених проблем. Причини можуть бути різноманітними, включаючи недостатню кваліфікацію інструктувального персоналу, відсутність належних ресурсів, недостатню увагу до безпеки праці, неефективну організацію процесу, недостатню інформаційну підтримку та інші.

1.1.4. Оцінка впливу проблем на підприємство

На цьому етапі проводиться оцінка впливу виявлених проблем на роботу підприємства. Визначається, які проблеми мають найбільший негативний вплив на продуктивність, ефективність та безпеку праці. Оцінка проводиться з урахуванням рівня ризику, можливостей удосконалення та покращення інструктування.

1.1.5. Формулювання висновків та рекомендацій

На основі проведеного аналізу формулюються висновки про виявлені проблеми та їх вплив на підприємство. Також розробляються рекомендації щодо вирішення виявлених проблем та удосконалення процесу проведення інструктажу. Рекомендації можуть стосуватися внесення змін до організаційної структури, політики безпеки праці, методик проведення інструктажу, використання інформаційних технологій та інших аспектів.

1.1.6. Висновки підрозділу

Завершуючи перший підрозділ, формулюються загальні висновки щодо пошуку наявних проблем в процесі проведення інструктажу на підприємстві. Аналіз дозволяє зрозуміти, які аспекти інструктування потребують удосконалення та автоматизації для поліпшення ефективності, безпеки та продуктивності робочого процесу на підприємстві.

На підприємствах часто існують різні проблеми, пов'язані з процесом проведення інструктажу. Наприклад, недостатня увага до безпеки праці може

призводити до нещасних випадків та травмувань працівників. Нечіткі або недостатньо структуровані інструкції можуть призвести до неправильного виконання робочих завдань і зниження якості продукції або послуг. Також можуть виникати проблеми з ефективністю процесу інструктування, коли не вистачає ресурсів або відсутня систематична перевірка засвоєння інформації.

Метою дослідження є виявлення та аналіз проблем, що виникають в процесі проведення інструктажу на підприємстві, з метою їх подальшого вирішення та удосконалення. Завдання дослідження включають:

- Збір і аналіз інформації про поточний процес проведення інструктажу на підприємстві.

- Виявлення проблем, що виникають в процесі інструктування.

- Визначення причин виникнення проблем і їх оцінка.

- Розробка рекомендацій щодо вирішення виявлених проблем та удосконалення процесу проведення інструктажу.

Для досягнення цієї мети та виконання завдань дослідження будуть використовуватись такі методи і засоби дослідження:

- Аналіз документації, що регулює проведення інструктажу та безпеку праці.

- Спостереження за процесом проведення інструктажу та взаємодією працівників.

- Анкетування працівників та учасників процесу інструктування.

- Співбесіди з керівниками та фахівцями з безпеки праці.

- Використання методів статистичного аналізу та моделювання.

Актуальність дослідження полягає у тому, що покращення процесу проведення інструктажу на підприємстві може привести до зниження ризиків,

покращення безпеки праці, підвищення ефективності роботи та забезпечення якості продукції або послуг. Це має важливе значення для підприємств, оскільки недоліки в процесі інструктування можуть мати негативний вплив на репутацію, фінансовий стан та конкурентоспроможність підприємства.

Таким чином, перший підрозділ роботи спрямований на пошук наявних проблем в процесі проведення інструктажу на підприємстві. Це включає аналіз поточного стану, ідентифікацію проблем, оцінку їх впливу на підприємство та формулювання висновків і рекомендацій.

1.2. Огляд сучасних рішень та методик автоматизації інструктування

У цьому підрозділі роботи буде проведено огляд сучасних рішень та методик, що застосовуються для автоматизації процесу інструктування на підприємствах. Автоматизація інструктування є важливим напрямом розвитку у сфері безпеки праці та підвищення ефективності робочих процесів.

Для здійснення огляду будуть використані наступні джерела і дослідження:

1. Літературні джерела: наукові статті, книги та журнальні публікації, що стосуються автоматизації інструктування та безпеки праці.

2. Аналіз веб-ресурсів та відповідних інтернет-платформ, які спеціалізуються на розробці і впровадженні автоматизованих систем інструктування.

3. Дослідження конкретних випадків впровадження автоматизованих систем інструктування на підприємствах.

Під час огляду будуть визначені наступні аспекти:

1. Огляд існуючих програмних рішень та систем, що використовуються для автоматизації процесу інструктування. Будуть проаналізовані їх функціональні можливості, ефективність та принципи роботи.

2. Вивчення методик та підходів до автоматизації інструктування, зокрема використання інтерактивних електронних платформ, відеоінструкцій, онлайн-курсів та інших засобів.

3. Аналіз досвіду впровадження автоматизованих систем інструктування на різних підприємствах. Будуть розглянуті позитивні аспекти впровадження, досягнуті результати та можливі проблеми.

На основі отриманих даних і вивчених рішень буде здійснено аналіз їх ефективності, переваг та недоліків. Результати огляду допоможуть визначити найбільш перспективні підходи та методики автоматизації інструктування, які можуть бути використані при розробці автоматизованої системи проведення інструктажу для підприємства, що розробляється в рамках даної роботи.

Цей підрозділ спрямований на систематизацію наявних сучасних рішень та методик автоматизації інструктування, що дозволить отримати об'єктивну інформацію та базовий матеріал для подальшого розроблення автоматизованої системи проведення інструктажу на підприємстві.

У цьому підрозділі проведемо детальний огляд сучасних рішень та методик, що застосовуються для автоматизації процесу інструктування на підприємствах. Автоматизація інструктування є актуальним напрямом розвитку у сфері безпеки праці та підвищення ефективності робочих процесів. Для початку, будуть вивчені і аналізовані літературні джерела, наукові статті, книги та журнальні публікації, що стосуються автоматизації інструктування. Це дозволить ознайомитися з актуальними тенденціями, розробками та використовуваними методами у цій сфері. Далі, буде проведено аналіз веб-

ресурсів та інтернет-платформ, спеціалізованих у розробці та впровадженні автоматизованих систем інструктування. Перевіримо наявні програмні рішення, їх функціональні можливості, ефективність та способи взаємодії з користувачами.

Також проведемо дослідження конкретних випадків впровадження автоматизованих систем інструктування на підприємствах. Дослідження включатиме огляд реальних проектів, їх успішність, переваги та недоліки, а також думки та досвід фахівців, які брали участь у впровадженні таких систем.

На основі проведеного огляду буде здійснений аналіз ефективності та переваг сучасних рішень та методик автоматизації інструктування. Будуть визначені найбільш перспективні підходи та методики, які можна застосувати при розробці автоматизованої системи проведення інструктажу на підприємстві.

Отримані результати огляду та аналізу сучасних рішень та методик будуть використані як основа для розробки власної автоматизованої системи проведення інструктажу підприємства. При цьому будуть враховані найкращі практики, унікальні рішення та можливості, які можуть принести значний внесок у покращення процесу інструктування та безпеки праці на підприємстві.

1.3. Визначення вимог до автоматизованої системи проведення інструктажу

У цьому підрозділі розглянемо процес визначення вимог до автоматизованої системи проведення інструктажу на підприємстві. Вимоги

виступають як основні критерії, що мають бути враховані при розробці системи з метою досягнення поставлених цілей та задач дослідження.

1.3.1. Функціональні вимоги

Розглянемо функціональні вимоги до автоматизованої системи проведення інструктажу. Вони визначають функції, які повинна виконувати система для задоволення потреб користувачів та досягнення мети дослідження. До таких вимог можуть належати:

- Можливість створення та редагування інструкцій та матеріалів для проведення інструктажу.

- Забезпечення доступу до інструкцій для працівників підприємства з можливістю ознайомлення та виконання завдань.

- Запис результатів інструктажу та збереження їх для подальшого аналізу та звітності.

1.3.2. Нефункціональні вимоги

Окрім функціональних вимог, ми також розглянемо нефункціональні вимоги, які визначають характеристики системи, її надійність, швидкодію, безпеку та інші аспекти. До нефункціональних вимог можуть належати:

- Інтуїтивно зрозумілий та легкий у використанні інтерфейс для користувачів системи.

- Забезпечення безпеки та конфіденційності інформації, що стосується інструктажу.

- Висока швидкодія та доступність системи для користувачів.

1.3.3. Технічні вимоги

Також важливо визначити технічні вимоги до автоматизованої системи проведення інструктажу. Ці вимоги стосуються апаратного та програмного забезпечення, необхідного для реалізації системи. До технічних вимог можуть належати:

- Підтримка певних операційних систем та мережевих протоколів.

- Наявність відповідних серверних та баз даних для зберігання інформації про інструктаж.

- Відповідність вимогам до безпеки, надійності та продуктивності обладнання.

Визначення вимог є критичним етапом у розробці системи проведення інструктажу, оскільки це впливає на подальші кроки розробки, вибір технологій та інфраструктури системи.

Під час аналізу вимог до системи основний акцент був зроблений на визначенні того, що потрібно зробити, без врахування способу його виконання. Під час розробки системи вирішуються питання, як здійснити рішення, прийняті на етапі аналізу. [32].

Спочатку проводиться розробка загальної структури (архітектури) системи. Архітектура системи визначає структуру системи, розподілена на модулі, і встановлює контекст для прийняття проектних рішень на кожному етапі розробки. Це означає, що вже на початкових етапах розробки вирішуються питання про організацію системи, взаємозв'язки між компонентами, інтерфейси, а також встановлюються принципи, які будуть

керувати роботою системи. Прийняття правильних архітектурних рішень впливає на подальшу розробку і успішне функціонування системи. [6].

Для планування архітектури системи необхідно вибрати відповідну систему управління базами даних (СУБД). В залежності від способу доступу до баз даних, існують різні типи СУБД. [7]:

- клієнт-серверні;
- файл-серверні;
- вбудовані.

Клієнт-серверна архітектура передбачає наявність двох взаємодіючих незалежних процесів - клієнта і сервера, які можуть працювати на різних комп'ютерах та обмінюватись даними по мережі. Ця архітектура може бути використана для побудови систем обробки даних на основі СУБД, поштових систем та інших.

У файл-серверній архітектурі дані зберігаються на файловому сервері (наприклад, NetWare Novell або Windows NT Server), а їх обробка відбувається на робочих станціях, на яких, зазвичай, використовується одна з так званих "настільних СУБД" - Access, FoxPro, Paradox і інші.

Додаток на робочій станції відповідає за всі аспекти роботи - формування інтерфейсу для користувача, логічну обробку даних і безпосереднє маніпулювання даними. Файловий сервер надає лише найнижчий рівень послуг - відкриття, закриття і модифікацію файлів, а не бази даних. База даних існує тільки на "мозку" робочої станції.

Маніпулювання даними здійснюється кількома незалежними та непромовленими процесами. Крім того, для будь-якої обробки дані потрібно передавати по мережі з сервера на робочу станцію.

У клієнт-серверній системи присутні (як мінімум) два додатки - клієнт та сервер, які розділяють між собою функціональні обов'язки, які у файл-серверній архітектурі зазвичай виконує додаток на робочій станції. Зберігання та безпосереднє маніпулювання даними здійснюється сервером баз даних, яким можуть бути Microsoft SQL Server, IBM DB2, Caché, Sybase, Firebird, Interbase, Informix, Oracle, PostgreSQL, MySQL, Лінтера та інші [9].

Вибір системи управління баз даних (СУБД) є складним завданням і представляє собою важливий етап у розробці додатків баз даних. Обране програмне рішення повинно задовольняти поточні та майбутні потреби підприємства, враховуючи при цьому фінансові витрати. Забезпечення відповідності загальним технічним вимогам є основною пріоритетною вимогою до інформаційної системи. [2].

Таблиця 1.1 - Характеристики інформаційної системи

Характеристики	**Коментарі**
Єдина база даних забезпечує розраховану на багато користувачів роботу.	Рекомендується використовувати централізовану базу даних, яка базується на потужних промислових системах управління базами даних, таких як MS SQL Server, Oracle, Informix або DB2.
Відсутність обмежень за кількістю об'єктів (максимальна кількість вимірювань, записів, звітів, число одночасно працюючих користувачів і т.д.).	Якщо існують будь-які обмеження на кількість об'єктів, з якими операційна система може працювати, це свідчить про недосконаліть технічного рішення даного програмного продукту. Навіть якщо ці обмеження наразі можуть виглядати задовільними, іх наявність

	може стати перешкодою для майбутнього розширення бюджетної моделі.
Інтеграція з суміжними автоматизованими системами.	Система повинна мати можливості для повноцінного імпорту та експорту даних, включаючи можливість попередньої обробки даних з різних облікових систем, якщо це необхідно. Бажано, щоб система забезпечувала двосторонній зв'язок з наявними обліковими системами. Використання стандартних систем управління базами даних також полегшує процес інтеграції.
Можливості доопрацювання системи на роботодавця.	Необхідно з'ясувати чи має постачальник програмного забезпечення можливість його доопрацювання на рівні програмного коду.

Розглянемо більш докладно найбільш поширені СУБД.

1) Oracle - одна з найпотужніших СУБД сучасності, створена для корпоративних баз даних з високими вимогами до сервера. Вона сумісна з багатьма операційними системами, такими як Windows-NT, 2000, Linux, UNIX, AIX, Nowell Netware [38].

Використання Oracle як СУБД надає можливість обирати мову програмування. Зазвичай для цього використовується мова PL/SQL, але також можна використовувати потужну мову програмування Java.

Oracle пропонує потужні та зручні засоби адміністрування як для одного сервера, так і для групи серверів, розташованих у різних регіонах світу.

Oracle має кілька ключових переваг, серед яких варто виділити підтримку баз даних великого обсягу (до 64 ГБ), потужні засоби для розробки та адміністрування, підтримку багатопроцесорних систем та наявність двох мовних середовищ. Крім того, Oracle забезпечує можливість інтеграції з Web, що дозволяє легко взаємодіяти з інтернет-середовищем і розробляти веб-додатки. Усі ці функціональні можливості сприяють ефективному управлінню базами даних та розробці потужних додатків на основі Oracle. Водночас, ця програма має високі апаратні вимоги та високу вартість.

2) СУБД Borland Interbase розроблена спеціально для малого та середнього бізнесу [10] і включає всі необхідні функції для задоволення їхніх потреб. Починаючи з версії 6.0, програма стала доступною безкоштовно, що значно полегшує її використання різними компаніями. Вона не вимагає високих апаратних вимог і підтримується на платформах Windows, Linux, UNIX, NetBSD і FreeBSD.

Borland надає популярні мови програмування, такі як Delphi, Kylix і C++ Builder, разом з компонентами, які спеціально розроблені для роботи з СУБД Borland Interbase. Це дозволяє розробникам створювати дуже швидкі програми, оскільки вони мають доступ до оптимізованих компонентів, спеціально призначених для роботи з цією СУБД.

3) СУБД MySQL здобула широке поширення як інструмент для роботи з базами даних в Інтернеті. Ця програма є абсолютно невимогливою до ресурсів сервера, на якому вона працює, дуже швидкою і, крім того, абсолютно безкоштовною. На сайті в Інтернеті можна знайти вихідні коди для різних платформ. Спочатку програма була розроблена для операційної системи Linux, але тепер вона доступна у версіях і для інших операційних систем, таких як Windows, UNIX, NetBSD, FreeBSD і AIX. Це означає, що користувачі цих

операційних систем також можуть використовувати цю програму і насолоджуватися її функціональністю та можливостями. Розширення підтримки до різних платформ робить програму більш універсальною і забезпечує доступність для широкого кола користувачів. Останнім часом програма набирає популярність серед користувачів Macintosh, які використовують операційну систему Mac OSX. [4].

5) MS Access є системою управління базами даних, яка використовується для вирішення локальних офісних завдань з невеликим обсягом даних. Вона дозволяє створювати звіти на основі результатів роботи, які можуть бути представлені у стандартному форматі, зрозумілому для багатьох офісних додатків.

MS Access - це програмне середовище, яке комбінує можливості розробки на двох мовах програмування (Visual Basic і спрощений діалект SQL), а також функції CASE-засобу і створення звітів. Це дозволяє розробникам створювати програми, працювати з базами даних та створювати звіти в зручному та наочному способі. [20].

За допомогою програмного забезпечення можна створювати програми, які складаються з одного файлу, включаючи текст програми і складну структуру реляційної бази даних. Access має зручну інтеграцію з іншими продуктами від Microsoft, що дає змогу використовувати його як клієнтську частину інформаційної системи. Особливо в поєднанні з MS SQL Server, який виконує роль серверної частини. Ця інтеграція дозволяє зручно обмінюватися даними між Access і SQL Server, використовуючи їх у великому комплексі інформаційних рішень. Таке поєднання забезпечує зручну та ефективну роботу з базами даних та доступ до різноманітної функціональності.

6) MS SQL Server є відомим своєю надійністю, безпекою, високою продуктивністю і зручністю в роботі протягом багатьох років. Ця сучасна СУБД є потужним програмним комплексом, який забезпечує можливість створення додатків будь-якої складності. Вона має широкі функціональні можливості, що дозволяють ефективно управляти базами даних та розробляти рішення для різних потреб користувачів. MS SQL Server став популярним вибором для багатьох організацій і розробників завдяки своїм передовим можливостям і надійності. Основою цього комплексу є база даних, яка може безмежно масштабуватися залежно від доступних ресурсів. Інформацію, що зберігається в цій базі даних, можуть одночасно обробляти будь-яка кількість користувачів з високою ефективністю, не спостерігаючи зниження продуктивності системи при збільшенні їх числа.

СУБД MS SQL Server мають вдосконалені механізми масштабування, які дозволяють збільшувати продуктивність і швидкість роботи сервера MS SQL Server і пов'язаних з ним додатків. Ці механізми дозволяють легко розширювати обсяги обробки даних і підвищувати продуктивність системи, щоб відповідати зростаючим потребам і завданням.

Однією з ключових переваг СУБД MS SQL Server є її універсальність, оскільки вона підтримує практично всі операційні системи, які наявні на сьогоднішній день. Це означає, що компаніям, які вибирають продукти MS SQL Server, не потрібно змінювати вже наявну мережеву інфраструктуру. Хоча є деякі незначні відмінності в роботі з СУБД, зумовлені особливостями конкретних операційних систем, в цілому MS SQL Server завжди залишається безпечним, надійним і зручним рішенням.

Слід зазначити раціональну міграційну стратегію, яку використовує MS SQL Server. Розуміючи, що перехід від старіших версій СУБД на новіші є

складним процесом, пов'язаним з тестуванням функціональності існуючих додатків у новому середовищі, MS SQL Server активно працює над забезпеченням сумісності вгору-вниз. Це робить процес міграції практично безболісним. Останні версії СУБД MS SQL Server мають спрощений процес встановлення і початкового налаштування. Крім того, збільшилися можливості для спеціалізованого налаштування СУБД під конкретні завдання. [9].

При виборі сервера СУБД MS SQL Server було враховано кілька факторів.

По-перше, технічні характеристики, які повністю відповідали вимогам. По-друге, доступність даної СУБД також мала значення. Серед усіх переваг MS SQL Server можна виділити наступні:

- простота і зручність адміністрування;
- невибагливість і мінімальні системні вимоги;
- ефективність і швидкодія;
- високий ступінь інтеграції в середовище розробки;
- висока надійність;
- можливість розширення бази даних;
- наявність універсальних засобів захисту інформації;
- орієнтованість на Інтернет-технології;
- порівняно низька ціна.

На підставі вищезазначених переваг MS SQL Server було обрано як оптимальне рішення для реалізації поставленого завдання створення СУБД .

Реінжиніринг означає глибоке переглядання та ревізію бізнес-процесів з метою досягнення значних поліпшень у фінансових та кількісних показниках організації.

Майкл Хаммер оголосив революційні на той час принципи: “Реконструюйте роботи, не автоматизовуючи, а спрощуючи або знищуючи”, “Використовуйте комп’ютери не тільки для автоматизації, а й для реконструкції існуючих бізнес-процесів”[41].

Реінжиніринг бізнес-процесів (РБП) передбачає створення цілком нових і більш ефективних бізнес процесів для досягнення вагомих покращень. В сьогоднішніх ринкових умовах, де постійні лише зміни, не завжди можливо утримувати свої позиції на ринку, процеси можуть перетворитися на “непридатні”, і не буде сенсу налагоджувати або поліпшувати їх; краще створити щось нове із самого початку, відносно поточної ситуації. Отже, суть реінжинірингу побудована на системі докорінних перетворень в організації. Спочатку моделюємо організацію, а потім змінюємо цю модель під рішення конкретних поточних та перспективних завдань, частіше за все шляхом рішучого відрубання нераціональних ланок та функцій. Концепція покращення бізнес-процесів ґрунтується на чотирьох підходах:

- Методика швидкого аналізу рішення.
- Бенчмаркінг процесу.
- Перепроектування процесу.
- Реінжиніринг процесу.

Методика швидкого аналізу рішення - це інноваційний підхід, який зосереджує увагу групи на конкретному процесі під час одно- або дводенної наради з метою визначення способів поліпшення цього процесу протягом наступних 90 днів.

Бенчмаркінг процесу - це систематичний метод, спрямований на визначення, розуміння та творчий розвиток товарів, проектів, послуг, обладнання, процесів та процедур вищої якості для покращення поточних діяльностей організації.

Цей метод полягає у вивченні того, як різні організації виконують подібні або схожі операції. Зазвичай, в результаті застосування бенчмаркінгу процесу вдається знизити витрати, скоротити тривалість циклу та зменшити кількість помилок на 20-50%.

Перепроектування процесу – концентрує зусилля на вдосконаленні існуючого процесу, як правило використовуються для тих процесів, які вже є досить успішні.

Реінжиніринг бізнес-процесів – найбільш радикальний із усіх чотирьох підходів до покращення бізнес-процесів. Реінжиніринг БП знижує витрати та тривалість циклу на 60-90% і рівень помилок на 40-70%. Часто цей процес стимулює команду, що стає справжнім проривом[29].

РОЗДІЛ 2. РОЗРОБКА АВТОМАТИЗОВАНОЇ СИСТЕМИ ПРОВЕДЕННЯ ІНСТРУКТАЖУ

2.1. Архітектура та функціональні вимоги до системи проведення інструктажу

У цьому розділі ми проведемо аналіз вимог до автоматизованої системи для проведення інструктажів на підприємстві. Наша мета - врахувати потреби та очікування користувачів, а також врахувати вимоги до функціональності, ефективності, надійності, безпеки та інших аспектів системи.

Спробуємо визначити функціональні вимоги, які повинна задовольняти система, тобто які конкретні функції та можливості вона повинна мати для задоволення потреб користувачів. Розглянемо можливості створення користувацького інтерфейсу для нашої системи, який буде зручним у

використанні для працівників та адміністраторів системи. Вивчимо можливості створення та збереження інструкцій, матеріалів та інших даних, пов'язаних з проведенням інструктажу. Також врахуємо нефункціональні вимоги до системи, такі як швидкодія, масштабованість, безпека даних, можливість резервного копіювання та відновлення, сумісність з існуючими системами та інші.

В даному підрозділі розробимо архітектуру автоматизованої системи проведення інструктажу. Визначимо основні компоненти системи, їх взаємозв'язок та взаємодію. Розглянемо можливість використання клієнт-серверної архітектури, веб-орієнтованої архітектури або інших підходів в залежності від вимог та потреб підприємства. Визначимо ролі та повноваження користувачів системи. Розглянемо можливості реєстрації користувачів, аутентифікації та авторизації. Розробимо модуль для управління користувачами та розподілу ролей, який забезпечить доступ до системи тільки авторизованим користувачам з відповідними правами. Також розробимо модуль для створення та збереження інструкцій та матеріалів, пов'язаних з проведенням інструктажу. Врахуємо можливість структурування інформації, пошуку, редагування та видалення даних. Розглянемо можливість надання доступу до інструкцій відповідно до ролей та потреб користувачів.

У даному підрозділі зосередимося на реалізації розробленої архітектури системи проведення інструктажу. Використовуючи відповідні технології та інструменти програмування, створимо необхідні модулі та функціонал системи. Після реалізації проведемо тестування системи для перевірки його функціональності, відповідності вимогам та виявлення можливих помилок. Виконаємо тестування різних сценаріїв використання системи, перевіримо

коректність збереження та відображення інформації, швидкодію системи та інші аспекти.

Крім того, проведемо оцінку ефективності та корисності системи на підприємстві. Зіберемо відгуки та повідомлення від користувачів системи, оцінимо їх задоволеність та переваги, які система надає у порівнянні з попередніми методами проведення інструктажу.

2.2. Проектування інтерфейсу користувача для зручного використання системи

У цьому підрозділі проведемо опис створення та функціонування баз даних, як основного засобу для реалізації моделі програми лояльності. База даних є необхідною складовою для зберігання, організації та обробки інформації, пов'язаної з програмою лояльності. Почнемо з огляду сучасних підходів та технологій у створенні баз даних. Вивчимо різноманітні моделі баз даних, такі як реляційні, об'єктно-орієнтовані, розподілені та інші, та порівняємо їх особливості, переваги та недоліки в контексті створення моделі програми лояльності. Розглянемо також різні системи управління базами даних (СУБД) та їх можливості.

Далі, визначимо основні елементи моделі програми лояльності, які будуть зберігатись у базі даних. Це можуть бути дані про клієнтів, їхні покупки, бонуси, знижки, історія транзакцій та інші. Опишемо структуру цих елементів і зв'язки між ними.

Важливим аспектом є забезпечення надійності та безпеки бази даних програми лояльності. Розглянемо методи інтеграції заходів безпеки, такі як шифрування, аутентифікація, авторизація та механізми резервного копіювання та відновлення даних. Звернемо увагу на важливість захисту персональних даних клієнтів та відповідність вимогам законодавства про захист персональних даних. Для реалізації моделі програми лояльності можуть бути використані різні інструменти та технології. Розглянемо можливість використання мов програмування, фреймворків та бібліотек для розробки логіки бази даних та інтерфейсу користувача. Описаної моделі, яка базується на базі даних, для її реалізації можуть бути використані такі засоби, як SQL-запити, ORM-технології, REST API та інші.

Результатом цього підрозділу буде детальний опис структури та функціональності бази даних, яка використовується для створення моделі програми лояльності. Також будуть виявлені ключові аспекти безпеки та надійності бази даних.

База даних "автосервіс" розробляється з метою обліку різних сутностей, таких як клієнти, співробітники, постачальники і виконані роботи. Ця база даних створена для використання працівниками автосервісу, які виконують свої обов'язки і взаємодіють з цією системою. Вона дозволяє зберігати і відстежувати інформацію про клієнтів, співробітників, постачальників і здійснені роботи для ефективного управління автосервісом.

Реляційна БД - основний тип сучасних баз даних. Складається з таблиць, між якими можуть існувати зв'язки з ключових значень [38].

Таблиця бази даних (table) - регулярна структура, яка складається з однотипних рядків (записів, records), розбитих на стовпці (поля, fields).

В теорії реляційних баз даних синонім таблиці - відношення (relation), в якому рядок називається кортежом, а стовпець називається атрибутом.

У концептуальній моделі реляційної БД аналогом таблиці є сутність (entity), з певним набором властивостей - атрибутів, здатних приймати певні значення (набір допустимих значень - домен).

Ключовий елемент таблиці (ключ, regular key) - таке її поле (простий ключ) або рядковий вираз, утворений із значень декількох полів (складових ключа), за яким можна визначити значення інших полів для однієї або декількох записів таблиці. На практиці для використання ключів створюються індекси - службова інформація, що містить впорядковані відомості про ключові значення. У реляційній теорії і концептуальній моделі поняття "ключ" застосовується для атрибутів відносин або сутності.

Первинний ключ (primary key) - головний ключовий елемент, однозначно ідентифікує рядок в таблиці. Можуть також існувати альтернативний (candidate key) і унікальний (unique key) ключі, слугує також для ідентифікації рядків в таблиці.

У реляційній теорії первинний ключ - мінімальний набір атрибутів, однозначно ідентифікує кортеж у відношенні.

У концептуальній моделі первинний ключ - мінімальний набір атрибутів сутності, однозначно ідентифікує екземпляр сутності [39].

Зв'язок (relation) - функціональна залежність між об'єктами. У реляційних базах даних між таблицями встановлюються зв'язки по ключах, один з яких в головній (parent, батьківської) таблиці - первинний, другий – зовнішній ключ - у зовнішній (child, дочірньої) таблиці, як правило, первинним не є і утворює зв'язок "один до багатьох" (1: N). У разі первинного зовнішнього

ключа зв'язок між таблицями має тип "один до одного" (1: 1). Інформація про зв'язки зберігається в базі даних.

Зовнішній ключ (foreign key) - така підмножина атрибутів дочірнього відношення, що для будь-якого його непорожнього значення обов'язково знайдеться рівне значення первинного ключа головного відношення.

Реляційна модель:

- Клієнт (код клієнта, найменування, контакти)
- Автомобіль (код авто, марка, модель, реєстраційний номер)
- Виконавець інструктажу (код замовлення, код співробітника, відсоток участі)
- Зовнішні ключі: код співробітника, посилається на таблицю «співробітник».
- Співробітник (код співробітника, ПІБ, код посади, контакти)
- Зовнішні ключі: код посади, посилається на таблицю «посаду».
- Робота (код роботи, найменування, код одиниці виміру, контакти)
- Зовнішні ключі: код одиниці виміру, посилається на таблицю «одиниці виміру».
- Інструктаж (код замовлення, дата, код клієнта, код авто, причина, стан)
- Зовнішні ключі: код клієнта, код авто, посилаються на таблиці «клієнт», «автомобіль».
- Посада (код посади, найменування, оклад)
- Повноваження (код посади, об'єкт доступу, читання, зміна, видалення)
- Одиниці виміру (код одиниці виміру, найменування)

- Запаси (код запасів, номер за каталогом, найменування, виробник, код одиниці виміру, ціна відпускна)
- Зовнішні ключі: код одиниці виміру, посилається на таблицю «одиниці виміру»
- Постачальники (код постачальника, найменування, реквізити, контакти)
- Місце зберігання (код місця, найменування

2.3. Розробка бази даних для зберігання інформації про інструктажі та працівників

Базуючись на концептуальній і реляційній моделях, були створені таблиці в базі даних MS SQL.

На рисунку 2.2 показана таблиця, яка містить дані про клієнтів. Для проведення інструктажу адміністратору спочатку потрібно внести контактні дані клієнта до цього довідника, а потім, в формі інструктажу, вибрати клієнта зі списку. Такий підхід дозволяє уникнути втрати контактної інформації клієнтів і помилок при введенні їх прізвища, імені та по батькові. Це забезпечує точність та зручність обробки даних про клієнтів під час проведення інструктажу.

Посади | Повноваження | Працівники | Одиниці виміру | Місця зберігання | Запаси | Постачальник | Роботи | Клієнти

КодКлієнта	Найменування	Контакти	Примітка
1	Сидоров Сергій Петрович	(063) 555 55 55	
2	ТОВ Спецперевезення	(095) 125 55 89	знижка 10% на всі види робіт

Рис. 2.2. Таблиця «працівники»

На рисунку 2.3 показана таблиця "працівники", яка містить дані про співробітників. У цій таблиці зберігаються логіни і зашифровані паролі, які використовуються для входу в інформаційну систему. Для збереження паролів використовується алгоритм хешування MD5, що дозволяє перетворити пароль на непередбачуваний хеш-код. Такий підхід забезпечує підвищену безпеку паролів, оскільки навіть в базі даних вони зберігаються у зашифрованому вигляді.

Рис. 2.3. Таблиця « проінструктовані працівники»

На рис. 2.4 представлена таблиця з марками автомобілів, дані марки будуть обиратися адміністратором зі списку при оформленні замовлення.

На рисунку 2.5 представлена таблиця, яка відображає посади в автосервісі та відповідні оклади для кожної посади. Ця таблиця містить інформацію про різні робочі посади, які існують в автосервісі, і визначає фіксований оклад, який пов'язаний з кожною конкретною посадою. Це дозволяє зручно вести облік заробітної плати співробітників в залежності від їхніх посад і забезпечує систематизацію та структурування даних про оклади.

Рис. 2.5. Таблиця «посади»

На рис. 2.6 показана таблиця повноважень для кожної посади, ці повноваження визначають доступ до інформаційної системи.

Посади | Повноваження | Працівники | Одиниці Виміру | Місця зберігання | Запаси | Постачальник | Роботи | Клієнт

Обрати посаду: Адміністратор

Посада	Об'єкт доступу	Читання	Зміна	Видалення
Адміністратор	Результати	1	0	0
Адміністратор	ДовідникКлієнтів	1	1	0
Адміністратор	ДовідникАвто	1	1	0
Адміністратор	ДовідникПрацівників	1	1	0
Адміністратор	ДовідникПосад	0	0	0
Адміністратор	ДовідникМісцьЗберігання	1	1	1
Адміністратор	ДовідникОдиниціВимірювання	1	1	1
Адміністратор	ДовідникЗапасів	1	1	0
Адміністратор	ДовідникРобіт	1	1	0
Адміністратор	ДовідникПостачальників	1	1	0
Адміністратор	ЗамовленняОсновнаІнформація	1	1	0
Адміністратор	ЗамовленняТовари	1	1	1
Адміністратор	ЗамовленняРоботи	1	1	1
Адміністратор	ЗамовленняВиконавці	1	1	1
Адміністратор	ПрихідЗапасів	1	1	0
Адміністратор	МатрицяПовноважень	0	0	0

Рис. 2.6. Таблиця «повноваження»

Під час проектування бази даних важливо враховувати аспекти цілісності даних. Це означає, що потрібно визначити правильну структуру таблиць, щоб запобігти порушенню зв'язків між даними і введенню некоректних значень. Для забезпечення цілісності даних необхідно визначити оптимальний підхід. Цілісність даних базується на стабільності і точності збережених даних у базі даних.

У базі даних було застосовано цілісність полів для таблиць. Особлива увага приділяється полям, таким як найменування, код замовлення, номер документа, де важливо уникати наявності нульових значень. Це важливо, оскільки наявність нульових значень може призвести до втрати необхідних даних і порушити цілісність бази даних.

У базі даних була забезпечена цілісність таблиці шляхом використання первинного ключа. У кожній таблиці бази даних кожен рядок має свій унікальний ідентифікатор, який називається *первинним ключем*. Цей первинний ключ дозволяє ідентифікувати і розрізняти кожен окремий рядок в таблиці. При видаленні будь-якої посади з бази даних, важливо позначити цю посаду як "на видалення". Це забезпечує збереження всіх даних, пов'язаних з цією видаленою посадою. Такий підхід дозволяє уникнути втрати інших даних, які можуть бути залежні від цієї посади, і забезпечує цілісність бази даних.

Крім того дотримана цілісність посилань. Відносини між первинним ключем (таблиці, на яку посилаються) і зовнішнім ключем (таблиці, яка посилається на іншу) завжди захищені. Якщо в таблиці існує зв'язок між рядками, то рядок основної таблиці, на який посилаються, не може бути видалений або його первинний ключ не може бути змінений, доки існує зв'язок з вторинним ключем. Це обмеження забезпечує цілісність даних і запобігає порушенню зв'язку між таблицями. Якщо спробувати видалити або змінити рядок основної таблиці, на який є посилання в другій таблиці, без попереднього видалення зв'язку, то зв'язок буде порушений. Відновлення такого зв'язку в подальшому може стати проблематичним.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ АВТОМАТИЗОВАНОЇ СИСТЕМИ

3.1. Вибір технологій та інструментів для реалізації системи проведення інструктажу

У цьому підрозділі розглянемо різні методи та засоби автоматизації, які можуть бути використані для ефективного проведення інструктажу на підприємстві. Ознайомимося з існуючими програмними рішеннями, системами та технологіями, які спрощують та поліпшують процес проведення інструктажу.

Розглянемо можливість використання електронних платформ та веб-додатків для проведення інструктажу. Дослідимо їх функціонал, можливості і переваги, які вони надають. Вивчимо можливість створення електронних інструктажів з використанням мультимедійних матеріалів, відео, анімації та інтерактивних елементів.

Також розглянемо можливість використання мобільних додатків для проведення інструктажу. Вивчимо їх можливості в контексті забезпечення мобільності та доступності для працівників підприємства. Розглянемо можливість створення персоналізованих інструктажів та можливість відстеження їх виконання через мобільні додатки.

Для автоматизації процесу проведення інструктажу також можуть бути використані спеціалізовані системи управління навчанням (Learning Management Systems). Дослідимо їх функціонал, можливості та переваги. Розглянемо можливість створення інтерактивних навчальних матеріалів, тестування та оцінювання знань працівників через ці системи.

В результаті цього підрозділу будуть визначені найбільш підходящі методи та засоби автоматизації для процесу проведення інструктажу на

підприємстві. Будуть обґрунтовані їх вибір та переваги в контексті задач та потреб підприємства.

3.2. Розробка та імплементація модулів системи проведення інструктажу

Вхідними даними повинні бути:

- за заявками - номер, дата заявки, клієнт, вид послуг, що надаються, ;
- по клієнтах – ПІБ клієнта, контакти, урахування знижки;
- по автомобілях – код, марка, модель, VIN, регіональний номер;
- за видами робіт - найменування роботи, працівник, що виконується даний вид ремонту;

Вихідними даними повинні бути:

- сформований звіт за даний інструктаж;

3.3. Тестування та валідація роботи системи

В процесі роботи над дипломною роботою була створена інформаційна система, яка забезпечує:

- підвищення ефективності управління;
- оптимізацію процесів збору, обробки, обліку та контролю інформації;
- підвищення якості обслуговування клієнтів, скорочення рутинної роботи;
- оперативність доступу до інформації для всіх підрозділів.

Далі вибирається класифікація ремонту. Потім кожному класу ремонту підбираються відповідно роботи і деталі. Після всього цього список всіх робіт

передається механіку або автослюсарю, а список необхідних деталей комірникові.

Рис. 3.1. Схема формування замовлення-наряду

Для забезпечення простоти інтерфейсу та підвищення безпеки даних, було прийнято рішення про розмежування доступу до бази даних. Адміністратор баз даних створив логіни і паролі для користувачів. Оскільки дані, які зберігаються в базі даних, мають фінансову значимість, було вирішено застосувати алгоритми шифрування для збереження паролів [8]. При запуску програми з'являється вікно, де користувач може обрати свій профіль.

Рис. 3.2. Форма авторизації

Для початку роботи необхідно обрати співробітника і ввести його пароль. Після натискання кнопки "Ок", пароль обробляється за допомогою хеш-функції, і отриманий хеш-значення порівнюється з відповідними даними в базі даних. Якщо авторизація пройде успішно і введений пароль збігається зі збереженим хешем, то робоча форма співробітника, відповідна його статусу, буде завантажена для подальшої роботи.

Подивитися код програми можна в Додатку.

Генеральний директор має можливість переглядати всю інформацію, пов'язану з автосервісом, за допомогою свого інтерфейсу, але доступна вона тільки для читання.

На формі, яка доступна, є можливість додавання та редагування виконуваних робіт та пов'язаних з ними запчастин. Цю можливість можна використовувати шляхом взаємодії з додатковими формами.

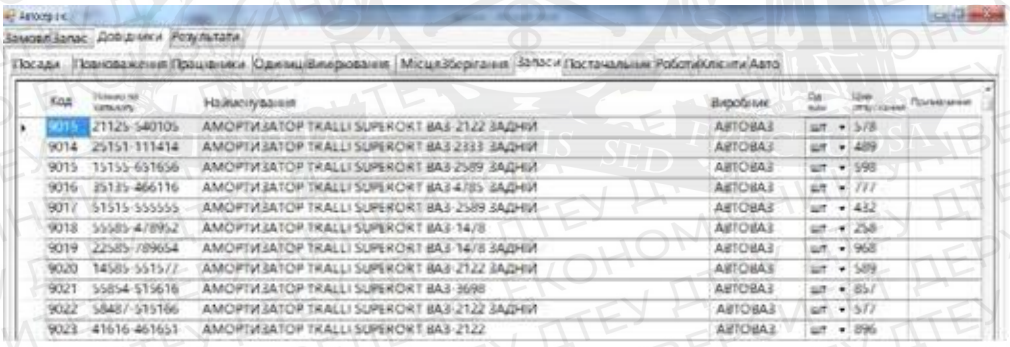

Рис. 3.3. Інтерфейс генерального директора

В адміністративному інтерфейсі є можливість додавання нових клієнтів та проведення інструктажу. Початково дані про клієнтів записуються в довідник клієнтів, щоб зберегти контактну інформацію та уникнути її втрати. Після цього зі списку доступних клієнтів можна вибрати того, хто пройшов інструктаж.

Розроблена інформаційна система дозволяє співробітникам вести облік запасів на складі, ведення списку клієнтів, відстеження стану робіт, управління списком співробітників, а також облік витрат і доходів від виконаних робіт.

ВИСНОВОК

У даній дипломній роботі була розроблена Автоматизована Система Проведення Інструктажу Співробітників для підприємства. Метою роботи було розробити автоматизований процес проведення інструктажу на підприємстві у формі ІС.

Висновки дослідження:

1. Було проведено аналіз наявних проблем в процесі проведення інструктажу на підприємстві. Виявлено, що існують певні недоліки, такі як недостатня систематизація інформації, низька доступність матеріалів для інструктування, втрата документації тощо.

2. Був проведений огляд сучасних рішень та методик автоматизації інструктування. Виявлено, що існують різноманітні підходи до автоматизації цього процесу, включаючи використання спеціалізованого програмного забезпечення, веб-порталів та мобільних додатків.

3. Були сформульовані функціональні, нефункціональні та технічні вимоги до автоматизованої системи проведення інструктажу. Враховано основні вимоги щодо забезпечення зручного інтерфейсу, доступності інформації, безпеки даних та інтеграції з існуючими системами.

Загальним результатом дипломної роботи є розробка концепції Автоматизованої Системи Проведення Інструктажу, яка враховує виявлені проблеми, використовує переваги сучасних рішень та відповідає визначеним вимогам. Розробка цієї системи може позитивно вплинути на ефективність інструктування на підприємстві, сприяти поліпшенню безпеки праці та зниженню ризиків для підприємства.

Отже, розробка Автоматизованої Системи Проведення Інструктажу є актуальною та необхідною для вирішення проблем, пов'язаних з проведенням інструктажу на підприємстві. Використання автоматизованої системи дозволить збільшити ефективність проведення інструктажу, забезпечити достовірність інформації та знизити ризик виникнення негативних наслідків для підприємства.

Результати даної дипломної роботи можуть бути використані як підгрунтя для подальшої розробки та впровадження Автоматизованої Системи Проведення Інструктажу на підприємстві.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Адміністрування Microsoft SQL Server 2000. Навчальний курс MCSA / MCSE, MCDBA

2. А. П. Сиротинська, І. Д. Лазаришина. Інформаційні системи підприємств малого бізнесу, 2008. – 155 с.

3. Гайдамакин Н. А., Автоматизовані інформаційні системи, бази і банки даних. Вступний курс: Навчальний посібник. - М .: Геліос АРВ, 2002. - 368 с.

4. Автоматизовані інформаційні системи і технології: навчальний посібник / В. Є. Юринець, Р. В. Юринець; М-во освіти і науки, молоді та спорту України, Львів. нац. ун-т ім. І. Франка. Л. : [ЛНУ ім. І. Франка], 2014.

5. Сайт з SQL і клієнт / серверної технології [Електронний ресурс]. - Режим доступу: http://www.sql.ru/, вільний.

6. Бусленко Н.П. Моделювання складних систем / Н.П. Бусленко. К. : [б.в.], 2013. – 165с.

7. Галіцина О.А. Бази даних: навчальний посібник / О. А. Галіцина. - 4-е вид., Перероб. і доп. - Юрайт, - 2014. - 78с .;

8. Денісова О. О. Автоматизоване проектування інформаційних систем: навчальний посібник / О.О. Денісова ; Міністерство освіти і науки, молоді та спорту України, Державний вищий навчальний заклад "Київський національний економічний університет імені Вадима Гетьмана". - Київ : КНЕУ, 2014.

9. Бекаревич Ю.Б., Пушкіна Н.В., Смирнова О.Ю. Управління базами даних. [Текст] / Бакаревіч Ю.Б., Пушкіна Н.В, 2009.

10. Жерновий Ю. В. Сучасні інформаційні системи та технології / Ю. В. Жерновий : підручник. – Львів: Видавничий центр ЛНУ імені Івана Франка, 2013. – 259с.

11. Джон Шарп. Microsoft Visual C #. Детальний опис, 8-е вид., 2017. – 655 с.

12. Мартін Р., Мартін М. Принципи, патерни і методики гнучкої розробки на мові C #, - 2011.

13. Віссер Дж. Розробка обслуговуючих програм на мові C #. Вид. ДМК Пресс, - 2016.

14. Марков О. Д. Організація автосервіса. Львів: Оріяна Нова, 1998. – 332 с.

15. Лудченко О. А. Технічна експлуатація і обслуговування автомобілів. Технологія – К.: Знання, 2009. – 478 с.

16. Канарчук В. Є., Лудченко О. А., Чигринець А. Д. Основи технічного обслуговування і ремонту автомобілів – К.: Вища шк., 2008.

17. Ким С. О., Пушкін П. С., Овчинніков С. І. Організація і планування промислового виробництва, - Мінськ: «Вища школа», - 1980.

18. Кожекін Г. Я., Синиця Л. М. Організація виробництва, - Мінськ: ПП «Екоперспектива», - 2008.

19. Інформаційні системи і технології: / Н.Б. Чорней, Р.К. Чорней ; Міжрегіональна академія управління персоналом. - Київ : [МАУП], 2015. - 187с.

20. Ю. Бекаревич, Н. Пушкіна. Самовчитель MS Office Access 2016, - 2017.

21. Анісімов А. П. Організація і планування автотранспортних підприємств. – М.: Транспорт, - 1982.

22. Будрій А. Г., Коновалов Г. А. Економіка автомобільного транспорту. – Київ: видавницький центр «Академія», - 2012.

23. Семенов Г. А., Станчевський В. К. Організація і планування на підприємстві. – Київ «Центр учбової літератури», - 2006.

24. Польшиков В. І., Сахно Є. Ю. Економіка, організація та управління технічним обслуговуванням і ремонтом машин. – Київ «Центр навчальної літератури», - 2010.

25. НДІАТ Короткий автомобільний довідник.

26. Норми витрат на матеріали і запасні частини.

27. Онищенко В. О., Старовірець А. С., Редкін О. В., Чевганова В. Я. Організація виробництва.

28. Польшаков В. І., Сахно Є. Ю. Економіка, організація та управління технічним обслуговуванням і ремонтом машин.

29. Гончарова О. М., Ефективна економіка, КНУ ім. Тараса Шевченка

30. Ризун Н. О. Сучасні інструменти аналізу складних економічних систем / Н. О. Ризун, Є. Г. Холод // Академічних огляд. – 2008. - № 1.

31. Трубілін І. Т. Автоматизовані інформаційні технології в економіці, 1999. – 416 с.

32. Ендрю Троелсен, Філіпп Джепікс. Мова програмування C # 7 і платформи .NET і .NET Core. Том 2, 2020.

33. Дж. Ріхтер. CLR via C #. Програмування на платформі Microsoft.NET Framework 4.5 мовою C #, 2016.

34. Брайан Нойєс. Прив'язка даних у Windows Forms, 2009.

35. Таха Х. Введення у дослідження операцій. Пер. з англ. [Текст, електронний ресурс] / Хемді Таха. – М.: Вільямс, 2005.

36. Томашевський В. Моделювання систем [Текст] / В. М. Томашевський. – К.: БПУ, 2005.

37. Методи моделювання бізнес-процесів підприємства засобами системного аналізу [Електронний ресурс]. – Режим доступу: http://vuz-lib.com/content/view/403/84/.

38. Чекалов А. Бази даних: від проектування до розробки додатків [Текст, електронний ресурс] / А. Чекалов 2003.

39. Янг Б. Об'єктно-орієнтовний аналіз і проектування с прикладами додатків. 3-є видання [Текст, електронний ресурс] / Боббі Янг, Джим Коннален, Граді Буч. – М.: Вільямс, 2008. – 720 с.

40. Підлипний, Ю. В. Розробка моделей, методів, та алгоритмів діагностики технологічних процесів виготовлення ІС : автореферат дисертації на здобуття наукового ступеня кандидата технічних наук / Ю.В. Підлипний. - Львів : 2013. – 166с.

41. М.Я. Гвоздь, Національний університет “Львівська Політехніка” – 106с.

ДОДАТОК

Програмний код реалізації додатку

```
///Represents the strongly named DataTable class.
///</summary>
[global::System.Serializable()]
[global::System.Xml.Serialization.XmlSchemaProviderAttribute("GetTypedTableSc
hema")]
public partial class КлієнтиDataTable :
global::System.Data.TypedTableBase<КлієнтиRow> {
private global::System.Data.DataColumn columnКодКлієнта;
private global::System.Data.DataColumn columnНайменування;
private global::System.Data.DataColumn columnКонтакти;
private global::System.Data.DataColumn columnПримітка;
[global::System.Diagnostics.DebuggerNonUserCodettribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public КлієнтиDataTable() {
this.TableName = "Клієнти";
this.BeginInit();
this.InitClass();
this.EndInit();
}
```

```
[global::System.Diagnostics.DebuggerNonUserCodettribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
internal КлієнтиDataTable(global::System.Data.DataTable table) {
this.TableName = table.TableName;
if ((table.CaseSensitive != table.DataSet.CaseSensitive)) {
 this.CaseSensitive = table.CaseSensitive;
}
if ((table.locale.ToString() != table.DataSet.locale.ToString())) {
 this.locale = table.locale;
}
if ((table.Namespace != table.DataSet.Namespace)) {
 this.Namespace = table.Namespace;
}
this.Prefix = table.Prefix;
this.MinimumCapacity = table.MinimumCapacity;
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
protected
КлієнтиDataTable(global::System.Runtime.Serialization.SerializationInfo info,
global::System.Runtime.Serialization.StreamingContext context) :
 base(info, context) {
```

```
this.InitVars();
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public global::System.Data.DataColumn КодКлієнтаColumn {
get {
 return this.columnКодКлієнта;
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public global::System.Data.DataColumn НайменуванняColumn {
get {
 return this.columnНайменування;
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public global::System.Data.DataColumn КонтактиColumn {
get {
 return this.columnКонтакти;
```

```
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public global::System.Data.DataColumn ПримітkаColumn {
get {
 return this.columnПримітка;
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
[global::System.ComponentModel.Browsable(false)]
public int Count {
get {
 return this.Rows.Count;
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public КлієнтиRow this[int index] {
get {
```

```
 return ((КлієнтиRow)(this.Rows[index]));
}
}
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public event КлієнтиRowChangeEventHandler КлієнтиRowChanging;
[global::System.CodeDom.Compiler.GeneratedCodeAttribute
"System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]
public event КлієнтиRowChangeEventHandler КлієнтиRowChanged;
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public event КлієнтиRowChangeEventHandler КлієнтиRowDeleting;
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public event КлієнтиRowChangeEventHandler КлієнтиRowDeleted;
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public void AddКлієнтиRow(КлієнтиRow row) {
this.Rows.Add(row);
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
```

```
public КлієнтиRow AddКлієнтиRow(string Найменування, string Контакти, string
Примітка) {
КлієнтиRow rowКлієнтиRow = ((КлієнтиRow)(this.NewRow()));
object[] columnValuesArray = new object[] {
 null,
 Найменування,
 Контакти,
 Примітка};
rowКлієнтиRow.ItemArray = columnValuesArray;
this.Rows.Add(rowКлієнтиRow);
return rowКлієнтиRow;
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public КлієнтиRow FindByКодКлієнта(int КодКлієнта) {
return ((КлієнтиRow)(this.Rows.Find(new object[] {
 КодКлієнта})));
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public override global::System.Data.DataTable Clone() {
КлієнтиDataTable cln = ((КлієнтиDataTable)(base.Clone()));
```

```
cln.InitVars();
return cln;
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
protected override global::System.Data.DataTable CreateInstance() {
return new КлієнтиDataTable();
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
internal void InitVars() {
this.columnКодКлієнта = base.Columns["КодКлієнта"];
this.columnНайменування = base.Columns["Найменування"];
this.columnКонтакти = base.Columns["Контакти"];
this.columnПримітка = base.Columns["Примітка"];
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
private void InitClass() {
this.columnКодКлієнта = new global::System.Data.DataColumn "КодКлієнта",
typeof(int), null, global::System.Data.MappingType.Element);
```

```
base.Columns.Add(this.columnКодКлієнта);
this.columnНайменування = new global::System.Data.DataColumn
"Найменування", typeof(string), null, global::
ystem.Data.MappingType.Element);
base.Columns.Add(this.columnНайменування);
this.columnКонтакти = new global::System.Data.DataColumn "Контакти",
typeof(string), null, global::System.Data.MappingType.Element);
base.Columns.Add(this.columnКонтакти);
this.columnПримітка = new global::System.Data.DataColumn "Примітка",
typeof(string), null, global::System.Data.MappingType.Element);
base.Columns.Add(this.columnПримітка);
this.Constraints.Add(new global::System.Data.UniqueConstraint "Constraint1",
new global::System.Data.DataColumn[] {
 this.columnКодКлієнта}, true));
this.columnКодКлієнта.AutoIncrement = true;
this.columnКодКлієнта.AutoIncrementSeed = -1;
this.columnКодКлієнта.AutoIncrementStep = -1;
this.columnКодКлієнта.AllowDBNull = false;
this.columnКодКлієнта.ReadOnly = true;
this.columnКодКлієнта.Unique = true;
this.columnНайменування.AllowDBNull = false;
this.columnНайменування.MaxLength = 100;
this.columnКонтакти.MaxLength = 200;
this.columnПримітка.MaxLength = 200;
```

```
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public КлієнтиRow NewКлієнтиRow() {
return ((КлієнтиRow)(this.NewRow()));
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
protected override global::System.Data.DataRow
NewRowFromBuilder(global::System.Data.DataRowBuilder builder) {
return new КлієнтиRow(builder);
}
 [global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
protected override global::System.Type GetRowType() {
return typeof(КлієнтиRow);
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
```

```
protected override void
OnRowChanged(global::System.Data.DataRowChangeEventArgs e) {
base.OnRowChanged(e);
if ((this.КлієнтиRowChanged != null)) {
 this. КлієнтиRowChanged(this, new
КлієнтиRowChangeEvent(((КлієнтиRow)(e.Row)), e.Action));
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
protected override void
OnRowChanging(global::System.Data.DataRowChangeEventArgs e) {
base.OnRowChanging(e);
if ((this. КлієнтиRowChanging != null)) {
 this. КлієнтиRowChanging(this, new
КлієнтиRowChangeEvent(((КлієнтиRow)(e.Row)), e.Action));
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
protected override void
OnRowDeleted(global::System.Data.DataRowChangeEventArgs e) {
```

```
base.OnRowDeleted(e);
if ((this. КлієнтиRowDeleted != null)) {
 this. КлієнтиRowDeleted(this, new
КлієнтиRowChangeEvent(((КлієнтиRow)(e.Row)), e.Action));
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
protected override void
OnRowDeleting(global::System.Data.DataRowChangeEventArgs e) {
base.OnRowDeleting(e);
if ((this. КлієнтиRowDeleting != null)) {
 this. КлієнтиRowDeleting(this, new
КлієнтиRowChangeEvent(((КлієнтиRow)(e.Row)), e.Action));
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public static global::System.Xml.Schema.XmlSchemaComplexType
GetTypedTableSchema(global::System.Xml.Schema.XmlSchemaSet xs) {
global::System.Xml.Schema.XmlSchemaComplexType type = new
global::System.Xml.Schema.XmlSchemaComplexType();
```

```
global::System.Xml.Schema.XmlSchemaSequence sequence = new
global::System.Xml.Schema.XmlSchemaSequence();
MyDataSet ds = new MyDataSet();
global::System.Xml.Schema.XmlSchemaAny any1 = new
global::System.Xml.Schema.XmlSchemaAny();
any1.Namespace = "http://www.w3.org/2001/XMLSchema";
any1.MinOccurs = new decimal(0);
any1.MaxOccurs = decimal.MaxValue;
any1.ProcessContents =
global::System.Xml.Schema.XmlSchemaContentProcessing.Lax;
sequence.Items.Add(any1);
global::System.Xml.Schema.XmlSchemaAny any2 = new
global::System.Xml.Schema.XmlSchemaAny();
any2.Namespace = "urn:schemas-microsoft-com:xml-diffgram-v1";
any2.MinOccurs = new decimal(1);
any2.ProcessContents =
global::System.Xml.Schema.XmlSchemaContentProcessing.Lax;
sequence.Items.Add(any2);
global::System.Xml.Schema.XmlSchemaAttribute attribute1 = new
global::System.Xml.Schema.XmlSchemaAttribute();
attribute1.Name = "namespace";
attribute1.FixedValue = ds.Namespace;
type.Attributes.Add(attribute1);
```

```
global::System.Xml.Schema.XmlSchemaAttribute attribute2 = new
global::System.Xml.Schema.XmlSchemaAttribute();
attribute2.Name = "tableTypeName";
attribute2.FixedValue = "КлієнтиDataTable";
type.Attributes.Add(attribute2);
type.Particle = sequence;
global::System.Xml.Schema.XmlSchema dsSchema = ds.GetSchemaSerializable();
if (xs.Contains(dsSchema.TargetNamespace)) {
 global::System.IO.MemoryStream s1 = new global::System.IO.MemoryStream();
 global::System.IO.MemoryStream s2 = new global::System.IO.MemoryStream();
 try {
 global::System.Xml.Schema.XmlSchema schema = null;
 dsSchema.Write(s1);
 for (global::System.Collections.IEnumerator schemas =
xs.Schemas(dsSchema.TargetNamespace).GetEnumerator();
schemas.MoveNext(); ) {
 schema = ((global::System.Xml.Schema.XmlSchema)(schemas.Current));
 s2.SetLength(0);
 schema.Write(s2);
 if ((s1.length == s2.length)) {
 s1.Position = 0;
 s2.Position = 0;
 for (; ((s1.Position != s1.length)
  && (s1.ReadByte() == s2.ReadByte())); ) {
```

```
 ;
 }
 if ((s1.Position == s1.length)) {
  return type;
 }
///Represents the strongly named DataTable class.
///</summary>
[global::System.Serializable()]
[global::System.Xml.Serialization.XmlSchemaProviderAttribute("GetTypedTableSc
hema")]
public partial class МісцяЗберіганняDataTable :
global::System.Data.TypedTableBase<МісцяЗберіганняRow> {
private global::System.Data.DataColumn columnКодМісцяЗберігання;
private global::System.Data.DataColumn columnНайменування;
private global::System.Data.DataColumn columnПримітка;
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public МісцяЗберіганняDataTable() {
this.TableName = "МісцяЗберігання";
this.BeginInit();
this.InitClass();
this.EndInit();
}
```

```
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
internal МісцяЗберіганняDataTable(global::System.Data.DataTable table) {
this.TableName = table.TableName;
if ((table.CaseSensitive != table.DataSet.CaseSensitive)) {
 this.CaseSensitive = table.CaseSensitive;
}
if ((table.Locale.ToString() != table.DataSet.Locale.ToString())) {
 this.Locale = table.Locale;
}
if ((table.Namespace != table.DataSet.Namespace)) {
 this.Namespace = table.Namespace;
}
this.Prefix = table.Prefix;
this.MinimumCapacity = table.MinimumCapacity;
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
protected
МісцяЗберіганняDataTable(global::System.Runtime.Serialization.SerializationInfo
info, global::System.Runtime.Serialization.StreamingContext context) :
 base(info, context) {
```

```
this.InitVars();
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public global::System.Data.DataColumn КодМісцяЗберіганняColumn {
get {
 return this.columnКодМісцяЗберігання;
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public global::System.Data.DataColumn НайменуванняColumn {
get {
 return this.columnНайменування;
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public global::System.Data.DataColumn ПримітkaColumn {
get {
 return this.columnПримітка;
}
```

```
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
[global::System.ComponentModel.Browsable(false)]
public int Count {
get {
 return this.Rows.Count;
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public МісцяЗберіганняRow this[int index] {
get {
 return ((МісцяЗберіганняRow)(this.Rows[index]));
}
}
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public event МісцяЗберіганняRowChangeEventHandler
МісцяЗберіганняRowChanging;
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
```

```
public event МісцяЗберіганняRowChangeEventHandler
МісцяЗберіганняRowChanged;
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public event МісцяЗберіганняRowChangeEventHandler
МісцяЗберіганняRowDeleting;
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public event МісцяЗберіганняRowChangeEventHandler
МісцяЗберіганняRowDeleted;
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public void AddМісцяЗберіганняRow(МісцяЗберіганняRow row) {
this.Rows.Add(row);
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public МісцяЗберіганняRow AddМісцяЗберіганняRow(string Найменування,
string Примітка) {
МісцяЗберіганняRow rowМісцяЗберіганняRow =
((МісцяЗберіганняRow)(this.NewRow()));
object[] columnValuesArray = new object[] {
```

```
 null,
 Найменування,
 Примітка};
rowМісцяЗберіганняRow.ItemArray = columnValuesArray;
this.Rows.Add(rowМісцяЗберіганняRow);
return rowМісцяЗберіганняRow;
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public МісцяЗберіганняRow FindByКодМісцяЗберігання (int
КодМісцяЗберігання) {
return ((МісцяЗберіганняRow)(this.Rows.Find(new object[] {
 КодМісцяЗберігання})));
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public override global::System.Data.DataTable Clone() {
МісцяЗберіганняDataTable cln = ((МісцяЗберіганняDataTable)(base.Clone()));
cln.InitVars();
return cln;
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
```

```
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
protected override global::System.Data.DataTable CreateInstance() {
return new МісцяЗберіганняDataTable();
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
internal void InitVars() {
this.columnКодМісцяЗберігання = base.Columns["КодМісцяЗберігання"];
this.columnНайменування = base.Columns["Найменування"];
this.columnПримітка = base.Columns["Примітка"];
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
private void InitClass() {
this.columnКодМісцяЗберігання = new global::System.Data.DataColumn
"КодМісцяЗберігання", typeof(int), null,
global::System.Data.MappingType.Element);
base.Columns.Add(this.columnКодМісцяЗберігання);
this.columnНайменування = new global::System.Data.DataColumn
("Найменування", typeof(string), null,
global::System.Data.MappingType.Element);
```

```
base.Columns.Add(this.columnНайменування);
this.columnПримітка = new global::System.Data.DataColumn("Примітка",
typeof(string), null, global::System.Data.MappingType.Element);
base.Columns.Add(this.columnПримітка);
this.Constraints.Add(new global::System.Data.UniqueConstraint("Constraint1",
new global::System.Data.DataColumn[] {
 this.columnКодМісцяЗберігання}, true));
this.columnКодМісцяЗберігання.AutoIncrement = true;
this.columnКодМісцяЗберігання.AutoIncrementSeed = -1;
this.columnКодМісцяЗберігання.AutoIncrementStep = -1;
this.columnКодМісцяЗберігання.AllowDBNull = false;
this.columnКодМісцяЗберігання.ReadOnly = true;
this.columnКодМісцяЗберігання.Unique = true;
this.columnНайменування.AllowDBNull = false;
this.columnНайменування.MaxLength = 100;
this.columnПримітка.MaxLength = 100;
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public МісцяЗберіганняRow NewМісцяЗберіганняRow() {
return ((МісцяЗберіганняRow)(this.NewRow()));
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
```

```
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
protected override global::System.Data.DataRow
NewRowFromBuilder(global::System.Data.DataRowBuilder builder) {
return new МісцяЗберіганняRow(builder);
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
protected override global::System.Type GetRowType() {
return typeof(МісцяЗберіганняRow);
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
protected override void
OnRowChanged(global::System.Data.DataRowChangeEventArgs e) {
if ((this. МісцяЗберіганняRowChanged != null)) {
 this. МісцяЗберіганняRowChanged(this, new
МісцяЗберіганняRowChangeEvent(((МісцяЗберіганняRow)(e.Row)), e.Action));
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
```

```
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
protected override void
OnRowChanging(global::System.Data.DataRowChangeEventArgs e) {
base.OnRowChanging(e);
if ((this. МісцяЗберіганняRowChanging != null)) {
 this. МісцяЗберіганняRowChanging(this, new
МісцяЗберіганняRowChangeEvent(((МісцяЗберіганняRow)(e.Row)), e.Action));
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
protected override void
OnRowDeleted(global::System.Data.DataRowChangeEventArgs e) {
base.OnRowDeleted(e);
if ((this. МісцяЗберіганняRowDeleted != null)) {
 this. МісцяЗберіганняRowDeleted(this, new
МісцяЗберіганняRowChangeEvent(((МісцяЗберіганняRow)(e.Row)), e.Action));
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
```

```
protected override void
OnRowDeleting(global::System.Data.DataRowChangeEventArgs e) {
base.OnRowDeleting(e);
if ((this. МісцяЗберіганняRowDeleting != null)) {
 this. МісцяЗберіганняRowDeleting(this, new
МісцяЗберіганняRowChangeEvent(((МісцяЗберіганняRow)(e.Row)), e.Action));
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public void RemoveМісцяЗберіганняRow(МісцяЗберіганняRow row) {
this.Rows.Remove(row);
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Desig
n.TypedDataSetGenerator", "4.0.0.0")]
public static global::System.Xml.Schema.XmlSchemaComplexType
GetTypedTableSchema(global::System.Xml.Schema.XmlSchemaSet xs) {
global::System.Xml.Schema.XmlSchemaComplexType type = new
global::System.Xml.Schema.XmlSchemaComplexType();
global::System.Xml.Schema.XmlSchemaSequence sequence = new
global::System.Xml.Schema.XmlSchemaSequence();
MyDataSet ds = new MyDataSet();
```

```
global::System.Xml.Schema.XmlSchemaAny any1 = new
global::System.Xml.Schema.XmlSchemaAny();
any1.Namespace = "http://www.w3.org/2001/XMLSchema";
any1.MinOccurs = new decimal(0);
any1.MaxOccurs = decimal.MaxValue;
any1.ProcessContents =
global::System.Xml.Schema.XmlSchemaContentProcessing.Lax;
sequence.Items.Add(any1);
global::System.Xml.Schema.XmlSchemaAny any2 = new
global::System.Xml.Schema.XmlSchemaAny();
any2.Namespace = "urn:schemas-microsoft-com:xml-diffgram-v1";
any2.MinOccurs = new decimal(1);
any2.ProcessContents =
global::System.Xml.Schema.XmlSchemaContentProcessing.Lax;
sequence.Items.Add(any2);
global::System.Xml.Schema.XmlSchemaAttribute attribute1 = new
global::System.Xml.Schema.XmlSchemaAttribute();
attribute1.Name = "namespace";
attribute1.FixedValue = ds.Namespace;
type.Attributes.Add(attribute1);
global::System.Xml.Schema.XmlSchemaAttribute attribute2 = new
global::System.Xml.Schema.XmlSchemaAttribute();
attribute2.Name = "tableTypeName";
attribute2.FixedValue = "МісцяЗберіганняDataTable";
```

```
type.Attributes.Add(attribute2);
type.Particle = sequence;
global::System.Xml.Schema.XmlSchema dsSchema = ds.GetSchemaSerializable();
if (xs.Contains(dsSchema.TargetNamespace)) {
 global::System.IO.MemoryStream s1 = new global::System.IO.MemoryStream();
 global::System.IO.MemoryStream s2 = new global::System.IO.MemoryStream();
 try {
 global::System.Xml.Schema.XmlSchema schema = null;
 dsSchema.Write(s1);
 for (global::System.Collections.IEnumerator schemas =
xs.Schemas(dsSchema.TargetNamespace).GetEnumerator();
schemas.MoveNext(); ) {
 schema = ((global::System.Xml.Schema.XmlSchema)(schemas.Current));
 s2.SetLength(0);
 schema.Write(s2);
 if ((s1.Length == s2.Length)) {
 s1.Position = 0;
 s2.Position = 0;
 for (; ((s1.Position != s1.Length)
  && (s1.ReadByte() == s2.ReadByte())); ) {
 ;
 }
 if ((s1.Position == s1.Length)) {
  return type;
```

```
}
}
}
finally {
if ((s1 != null)) {
s1.Close();
}
if ((s2 != null)) {
s2.Close();
}
```