

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування інституту, факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ОБЛІКУ
ЗЕМЕЛЬНИХ ДІЛЯНОК
SOFTWARE FOR KEEPING TRACK OF LAND PLOTS

Виконав: студент(ка) 4 курсу, групи КНТз-219_2

Спеціальності 122 Комп'ютерні науки

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерні науки

Пащенко А.С.

(прізвище та ініціали)

Керівник Олійник А.О.

(прізвище та ініціали)

Рецензент Гофман Є.О.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Інститут, факультет ФКНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 122 Комп'ютерні науки
(код і найменування)
Освітня програма (спеціалізація) Комп'ютерні науки
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
“ ” 2022 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Пащенко Артема Сергійовича

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення обліку земельних ділянок.
Software for Keeping Track of Land Plots

керівник проєкту (роботи) Олійник Андрій Олександрович, д.т.н., професор,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “27” квітня 2022 року № 102

2. Строк подання студентом проєкту (роботи) 07 червня 2022 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне
завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали і методи. 3. Опис програми. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	Олійник А.О., професор		
Нормоконтроль	Липовець М.В., асистент		

7. Дата видачі завдання “04” квітня 2022 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій розробки.	2 тиждень	Розділ 2
4	Розробка архітектури програми.	2 тиждень	Розділ 3
5	Розробка програми.	3-4 тижні	Розділ 3, 4
6	Тестування та експериментальне дослідження програмного забезпечення.	5 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї.	6 тиждень	Додатки
8	Нормоконтроль та рецензування.	7 тиждень	
9	Захист роботи.	8 тиждень	

Студент(ка)

(підпис) Пашенко А.С.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис) Олійник А.О.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 91 с., 29 рис., 7 табл., 4 дод., 15 джерел.

ЗЕМЕЛЬНА ДІЛЯНКА, АРАСНЕ, КЛІЄНТ, СЕРВЕР, HTML, CSS, PHP, MySQL.

Об'єкт дослідження – процес розробки програмного забезпечення для обліку земельних ділянок.

Предмет дослідження – програмні засоби обліку земельних ділянок.

Метою роботи є програмна реалізація процесу обліку земельних ділянок.

Матеріали, методи та технічні засоби: мова програмування Php, середовище розробки, система керування базами даних MySQL.

Результати. Розроблено програмні засоби для обліку земельних ділянок.

Практична цінність роботи полягає у розробці програмного забезпечення для автоматизації обліку діяльності кадастрового підприємства з обліку земельних ділянок.

Висновки. Розроблено програмне забезпечення для автоматизації обліку земельних ділянок.

Галузь використання – кадастрова діяльність.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 91 pages, 29 figures, 7 tables, 4 appendixes, 15 sources.

LAND, APACHE, CLIENT, SERVER, HTML, CSS, PHP, MYSQL.

The object of research is the process of software development for land accounting.

The subject of the research is software tools for land accounting.

The purpose of this work is the program implementation of the process of land accounting.

Materials, methods and technical tools: Php programming language, development environment, MySQL database management system.

Results. The software tools for land registration have been developed.

The practical value of the work lies in the development of software to automate the accounting of cadastral enterprises for land registration.

Conclusions. The software for automation of the land accounting is developed.

Scope of use – cadastral activities.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Аналіз предметної області.....	11
1.1 Загальні поняття про облік земельних ділянок	11
1.2 Дослідження програмних засобів для обліку земельних ділянок	16
1.3 Висновки за розділом 1	24
2 Матеріали і методи	25
2.1 Вибір мови програмування та інших технологій розробки	25
2.2 Вибір системи керування базою даних	28
2.3 Вибір вебсерверу для розташування програмного забезпечення.....	31
2.4 Висновки за розділом 2	32
3 Опис програми	33
3.1 База даних програмного забезпечення обліку земельних ділянок.....	33
3.2 Структурна схема програми	36
3.3 Основні модулі програми обліку земельних ділянок	38
3.4 Висновки за розділом 3	40
4 Експлуатація, тестування та експериментальне дослідження програми.....	41
4.1 Призначення й умови застосування програми	41
4.2 Характеристики програми	42
4.3 Інструкція по експлуатації програми	42
4.3.1 Звернення до програми.....	42
4.3.2 Вхідні й вихідні дані	42
4.3.3 Повідомлення	42
4.4 Тестування та результати роботи програмного забезпечення обліку земельних ділянок	43
4.5 Висновки за розділом 4	46
Висновки	47
Перелік джерел посилання	49

Додаток А Технічне завдання	51
Додаток Б Опис програми	55
Додаток В Текст програми	58
Додаток Д Слайди презентації.....	84

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

БД – база даних;

ГІС – геоінформаційна система;

ПЗ – програмне забезпечення;

ПП – програмний проєкт;

ОС – операційна система;

СКБД – система керування базами даних.

ВСТУП

Земля відіграє величезну роль у створенні матеріальних благ. Це джерело людського соціального багатства [1]-[5]. Процес матеріального виробництва безпосередньо пов'язаний з використанням землі як просторової основи і засобу виробництва. Водні та лісові ресурси також органічно пов'язані з ґрунтом [1]-[5]. Ґрунт відіграє активну роль як основний засіб виробництва в сільському та лісовому господарстві. Будь-яка діяльність людини (виробнича, комерційна чи просто побут) нерозривно пов'язана з землею. Перш за все земля є природним ресурсом, що визначається простором, рельєфом, ґрунтом і водою, є розвитком продуктивних сил, об'єктом соціально-економічних відносин і основою всієї сфери життя [1]-[5].

Важливим завданням організації і планування діяльності сільськогосподарських підприємств є раціональне поєднання і використання земельних ресурсів з метою одержання максимального прибутку [1]-[5]. Проте багато земельних ділянок, що знаходяться у розпорядженні сільськогосподарських підприємств на правах оренди, має різних власників, що ускладнює процес обліку таких ділянок та оцінювання ефективності результатів господарювання на них [1]-[5].

Тому актуальною є розробка програмного забезпечення для автоматизації процесу обліку земельних ділянок, у т.ч. для підприємств сільського господарства.

Отже, метою роботи є програмна реалізація програмна реалізація процесу обліку земельних ділянок.

Для досягнення поставленої мети в роботі необхідно розв'язати такі задачі:

- виконати аналіз предметної області відносно процесу обліку земельних ділянок;

- виконати проєктування програмного забезпечення для обліку земельних ділянок;
- створити програмне забезпечення для обліку земельних ділянок;
- виконати тестування розробленого програмного забезпечення обліку земельних ділянок.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Загальні поняття про облік земельних ділянок

Україна є однією з найбільших країн Європи, має багато землі та ресурсів [1]-[5]. Майже 70% із 60,3 млн. га становлять сільськогосподарські угіддя, також на території нашої держави знаходиться біля 10 мільйонів гектарів лісу та приблизно 2,3 млн. га (4%) знаходяться в стадії будівництва [1]-[5].

Важливим завданням організації і планування діяльності підприємств є раціональне поєднання і використання земельних ресурсів з метою одержання максимального прибутку [1]-[5]. Тому створення сучасного земельного кадастру є складним науково-технічним завданням [1]-[5].

Крім того, завдання обліку земельних ділянок є також актуальним для підприємств, що працюють на різних земельних ділянках, як орендованих, так і власних [1]-[5].

Сьогоднішній земельний кадастр включає інтегровані бази даних, інформацію, геоінформаційні технології, цифрові електронні карти, аерофотозйомку та супутникову фотографію [1]-[5]. Вискокваліфіковані спеціалісти створюють, доповнюють та розвивають національну кадастрову систему. В останні роки системи обліку земель лише частково базуються на використанні інформаційних технологій [1]-[5]. Раніш облік відбувався майже виключно на паперовій картографії, що ускладнює автоматизований облік у наш час. Така ситуація спрогнозувала необхідність використання великої кількості робочої сили, стала джерелом помилок, перешкоджала використанню земельно-кадастрової інформації великою аудиторією та затягувала реєстрацію земельних реєстрів [1]-[5].

Система державного земельного кадастру, розроблена з використанням сучасних інформаційних технологій, дозволяє швидко отримувати необхідну інформацію про землю та ділянки по всій Україні за допомогою інформації кадастрових об'єктів, відображеної на основі

цифрової картографії [1]-[5]. У цьому випадку достовірність та достовірність інформації ґрунтується на отриманні інформації з об'єднаної земельно-кадастрової бази. Інформація вноситься до єдиної бази даних, контролюється, систематизується та упорядковується кваліфікованими спеціалістами в єдиному порядку [1]-[5].

Сьогодні з використанням мережі Інтернет інформація автоматичної кадастрової системи землі є легко та швидко доступною для широкого кола користувачів [1]-[5]. Земельний кадастр є основою для ведення кадастру інших природних ресурсів. Метою діяльності земельного кадастру є забезпечення громадян, підприємств, органів державної влади та місцевого самоврядування, відповідних суб'єктів господарювання, організацій необхідною інформацією щодо регулювання земельних відносин, належного використання та охорони земель, плати за землю та вартості землі, економіко-екологічне обґрунтування контролю за природними ресурсами, використанням і збереженням земель, бізнес-плани, проекти землеустрою [1]-[5]. Облік якості земель включає інформацію, яка ідентифікує землю з її природними особливостями та набутими характеристиками, що впливають на родючість земель, а також ступінь забруднення ґрунтів [1]-[5]. Земельно-кадастровий облік ведеться в управлінні земельного кадастру. Державний земельний кадастр відіграє особливу роль як база даних, яка модернізує земельні відносини, підтримує ефективне землеустрій, реєстрацію, землеустрій, державну податкову та інноваційну політику, створює та розвиває ринки землі, обґрунтовує платіж за землю [1]-[5]. На всіх етапах суспільного розвитку державний земельний кадастр є основним інструментом реалізації державної земельної політики [1]-[5].

Використання геоінформаційних систем (ГІС) є важливим інструментом інтеграції інформації про природні та соціально-економічні об'єкти у вигляді електронних карт [1]-[5]. Використання ГІС у різноманітних автоматизованих кадастрових системах відповідає сучасним вимогам щодо реалізації проєктів управління просторовою інформацією для

всіх підприємств та об'єктів управління [1]-[5]. В Україні багато земельних кадастрів та геоінформаційних систем накопичили інформацію на різних рівнях, але цю інформацію можна використовувати швидко й автоматично. Для об'єднання кількох типів кадастрових даних слід використовувати велику кількість різноманітних кадастрових відомостей для створення єдиної картографічної основи [1]-[5].

Використання ГІС-технологій включає збір, зберігання та гармонізацію цих стандартів і форматів інформації [1]-[5]. Основним завданням ГІС є робота над інтеграцією різної кадастрової інформації в процесі вирішення науково-практичних завдань [1]-[5]. Доведено важливість геоінформаційних технологій у кадастровій роботі для виконання обов'язків муніципальних органів влади [1]-[5]. Виявлено, що загальнотеоретичною основою вивчення географічного середовища та різноманітних об'єктів є проблемне картографування [1]-[5]. Для організації ГІС колективної оцінки земель слід використовувати векторну модель [1]-[5]. Для оцінювання та обліку земельних ділянок доцільно розробити програмні засоби, що забезпечують виконання таких функціональних вимог: підготовку вихідних даних для обліку та оцінки ділянки, тобто даних з різних джерел у різних форматах та на різних носіях використання; перевірка вихідних даних, що полягає у підвищенні основних критеріїв адекватності даних, методів їх виявлення, помилок, доповнень і виправлень; використання методів оцінки вартості земельної ділянки; перевірка результатів проміжної оцінки, що полягає у визначенні значень набору критеріїв адекватності результатів оцінювання; відображення результатів контролю, динаміки змін вихідних даних та результатів оцінки; консолідація результатів оцінювання, що полягає у застосуванні можливостей генерації звітів за результатами оцінки [1]-[5].

Для присвоєння земельній ділянці кадастрового номера ця інформація має бути внесена до системи автоматизації обліку земель у земельному кадастрі [1]-[5]. Кадастровий номер – це унікальний номер, який

присвоюється ділянці протягом її існування [1]-[5]. Кадастровий номер не змінюється при зміні форми власності, форми чи призначення [1]-[5]. Єдина умова зміни кадастрового номера – зміна площі земельної ділянки, її меж, тобто її поділу та єдності, а старий номер набуде статусу архівного [1]-[5]. Кадастровий номер необхідний для ідентифікації місцезнаходження в базі даних автоматизованої системи державного земельно-кадастрового обліку. Для успішного проведення обліку земельних ділянок необхідно знати правовий та природний стан землі, хто є власником та користувачем землі, за яких умов використовується земельна ділянка, її площу, структуру землі. Тому для кожної складової земельного кадастру необхідно визначити відповідні параметри [1]-[5]. Реєстрація землі є основним процесом легалізації прав на землю. При цьому важливо, що узаконити земельну ділянку можна лише шляхом визначення її цільового призначення, площі, меж і кадастрового номера [1]-[5].

На даному етапі інформація про державну власність на землю, регіональні партнерства та громадян не є критичною [1]-[5]. Класифікація земель дозволяє класифікувати їх за найбільш специфічними характеристиками [1]-[5]. Відповідно до земельного законодавства основними ознаками є цільове призначення землі – право власності на землю, право користування землею або нормативно-правовий режим користування та норми користування залежно від діяльності суб'єктів, що становлять суспільний інтерес [1]-[5]. За цільовим призначенням земельні ділянки можуть бути класифіковані таким чином [1]-[5]:

- землі сільськогосподарського призначення;
- земельні ділянки для будівництва квартир та громадських будівель;
- землі заповідного фонду;
- землі рекреаційного призначення;
- лісові угіддя;
- землі, на яких розташовано водні ресурси;

– землі промисловості, транспорту, телекомунікацій, енергетики, оборони та іншого призначення.

Землеустрій — це комплекс соціальних, економічних та екологічних заходів, спрямованих на раціональну організацію території адміністративно-територіальних одиниць і комерційних організацій та регулювання земельних відносин під впливом соціально-виробничих відносин [1]-[5]. Землеустрій забезпечує [1]-[5]:

- визначення адміністративно-територіальних одиниць, права землекористування та меж землекористування;
- розроблення загальнодержавних та регіональних програм використання та збереження земель;
- розроблення плану землеустрою та техніко-економічне обґрунтування використання та охорони земель відповідними адміністративно-територіальними одиницями;
- обґрунтування розмежувань природоохоронних, рекреаційних та заповідних територій;
- координація існуючого землеволодіння та землекористування та розробка нових проєктів;
- розробка проєктів відведення земель;
- накладання обмежень на використання земель;
- складання документів, що підтверджують право власності на землю та права користування;
- підготовка проєктів землеустрою та розроблення заходів з охорони земель для забезпечення землеустрою;
- розробка інших землепорядних документів, пов'язаних із використанням та охороною земель;
- авторський контроль за використанням земель та виконанням природоохоронних проєктів;
- картографічні, ґрунтові та інші дослідження та розвідки.

Метою землеустрою є створення оптимальної системи використання земельних ресурсів, охорона навколишнього середовища, створення екологічно сприятливого середовища, розселення, створення виробничої інфраструктури як засобу виробництва та використання [1]-[5]. Власники земельних ділянок, землекористувачі, у тому числі орендарі, зобов'язані при здійсненні землеустрою дозволити розробникам землевпорядних документів використовувати землю, яка буде використовуватися у землеустрою [1]-[5].

1.2 Дослідження програмних засобів для обліку земельних ділянок

У наш час існує різне програмне забезпечення, що використовується для обліку земельних ділянок.

Заснована на технології Microsoft Azur, платформа Peloton LandView [6] працює з користувачами, щоб допомогти організаціям максимізувати ефективний цикл використання даних, життєвий цикл виробничих даних і рішення життєвого циклу керування даними. Peloton охоплює лізингові покупки, історію власності, платежі, строки погашення та пов'язані з ними зобов'язання [6]. Програмне забезпечення LandView є частиною інтегрованого рішення з управління земельними ресурсами Peloton і є комплексною системою, яка контролює всі операції земельного процесу [6].

Програмне забезпечення Peloton LandView має наступні особливості та переваги [6]:

- управління договором, орендою землі;
- візуалізатор ланцюжка заголовків інформації щодо земельної ділянки;
- зобов'язання та платежі;
- наявність інформаційного розділу, що вміщує основну інформацію про земельну ділянку;
- управління роялті;

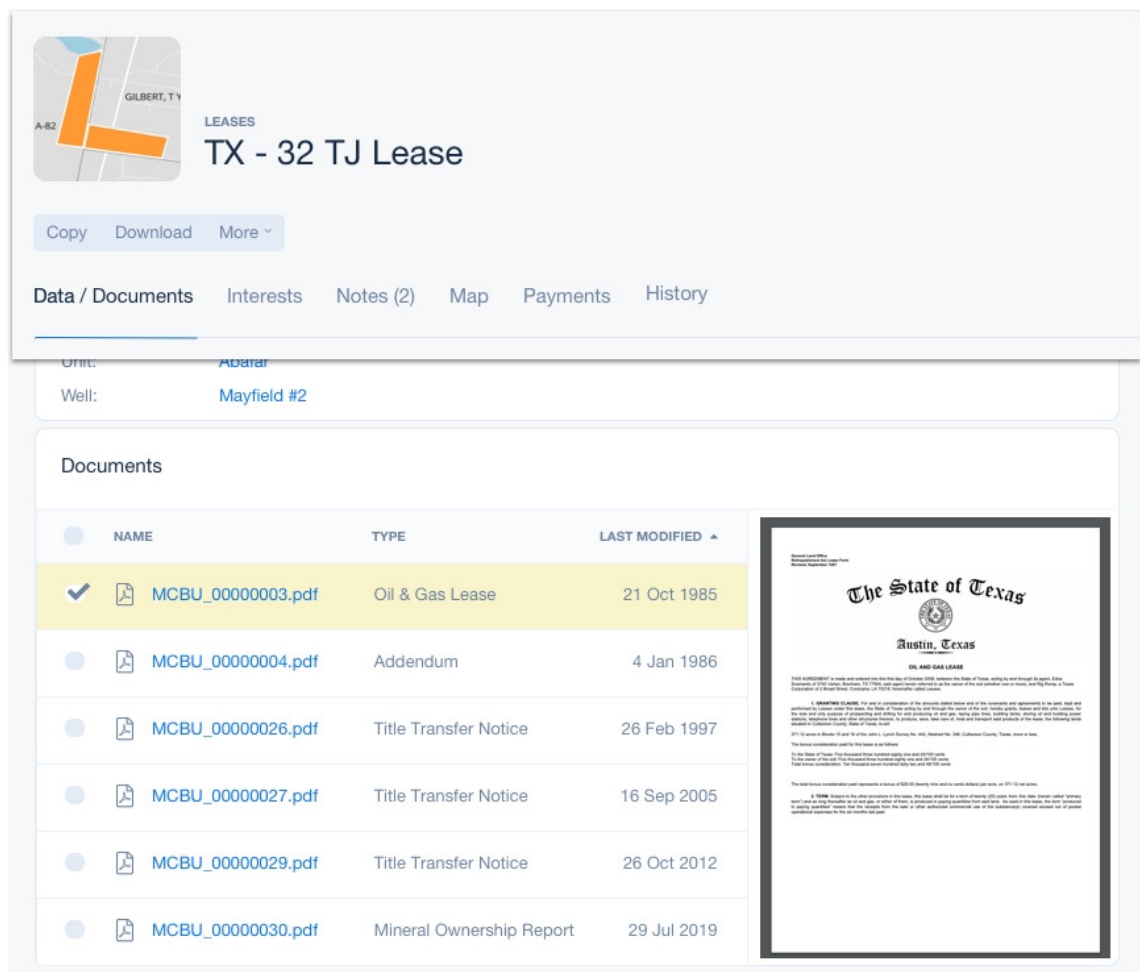
- відстеження інформації про споруди на земельних ділянках, зокрема інформації про колодязі та будівлі;
- наявність карт.

Peloton Map є хмарною програмою, призначеною для перегляду, запиту та аналізу даних Peloton [6]. Він розширює платформу онлайн-картографування ArcGIS від ESRI та інтегрується безпосередньо з LandView через вебпротокол зв'язку та використовує наявні інвестиції користувачів у картографування [6]. LandView створює просторове представлення даних шляхом аналізу та відображення форм рельєфу в реальному часі в режимі реального часу. Існує двостороння, залежна від контексту навігація між LandView та картою Peloton. Користувачі можуть створювати власні карти, виконувати ті ж запитання, що й LandView, щоб знайти інформацію, проаналізувати набір даних і створити повну картину області, яка цікавить [6].

P2 iLandMan є досить відомим програмним забезпеченням у наземній промисловості [7]. Підтримує весь ланцюжок створення вартості роботи землевласника, від контрактів до платежів. Загалом, рішення дуже зручне для користувачів, і за кілька днів навчання ви будете повністю готові до роботи. Має надійний API. Відображає зобов'язання щодо платежів, відображення та перерахування відсотків (рис. 1.1) [7].

Однією з найсильніших особливостей iLandMan є його здатність обробляти контракти (оренди нафти та газу) з кількома угіддями та кількома власниками корисних копалин в рамках договорів оренди унікальних завдяки унікальним ідентифікаторам, що використовуються у системі [7].

Landdiox представляє собою хмарне рішення, яке поєднує в собі все, від керування даними та документами, до картографування та аналітики, у повністю налаштованій платформі, розробленій (на нашу думку) для більш вимогливих користувачів (рис. 1.2). Цей продукт надає глибокий галузевий стандарт та передову архітектуру, які допоможуть енергетичним компаніям переосмислити свої зусилля з управління землею [8].



Безпечний API та набір інструментів для розробників Landdox забезпечують інтеграцію з іншими системами, базами даних (наприклад, Drillinginfo) і картами (наприклад, ArcMap) [8].

Усі форми введення даних повністю налаштовуються [8]. Це щось нове для програмного забезпечення землі. Платформа схожа на LinkedIn або Facebook з можливістю швидкого пошуку та знаходження інформації. Новим користувачам із землею знадобиться лише кілька годин навчання, щоб стати кваліфікованим [8].

Landdox також включає в систему управління документами. Документи, завантажені в систему, мають OCR (оптичне розпізнавання символів) для легкого пошуку тексту (рис. 1.3) [8].

WELLS

ABNEY 09-07 # 3-21H16

Data Notes Documents Revenue Decks Expense Decks **Payments**

REVENUE
\$20,927.73

EXPENSE
\$12,901.78

+ New revenue

Showing 1-10 of 10 « Previous 1 Next »

DATE	ID	PAYER	PAYEE	INTEREST	AMOUNT	DESCRIPTION
18 Oct 2019	R-C81804FE	Chao Energy	Walter Coker	RI	\$931.20	
18 Sep 2019	R-285C1AE9	AARON TERRY	Andrew Don Fry	RI	\$2,908.12	
18 Sep 2019	R-FD03556A	Chao Energy	Walter Coker	RI	\$3,178.35	
18 Aug 2019	R-7ADC4527	Chao Energy	Walter Coker	RI	\$2,698.01	
18 Jul 2019	R-D5B56F21	Chao Energy	Walter Coker	RI	\$2,847.49	
18 Jun 2019	R-9637BFC9	Chao Energy	Walter Coker	RI	\$841.00	
18 May 2019	R-8514AA11	Chao Energy	Walter Coker	RI	\$1,003.76	
18 Apr 2019	R-639FBB34	Chao Energy	Walter Coker	RI	\$509.12	

Рисунок 1.3 – Система управління документами програми Landdox [8]

Програмне забезпечення MyQuorum пропонує інтеграцію з іншими хостинговими рішеннями Quorum [9]. Land on Demand, розроблений для

малих і середніх нафтогазових компаній, які мають обмежену кількість контрактів і ще не готові до корпоративного рішення Quorum (рис. 1.4).

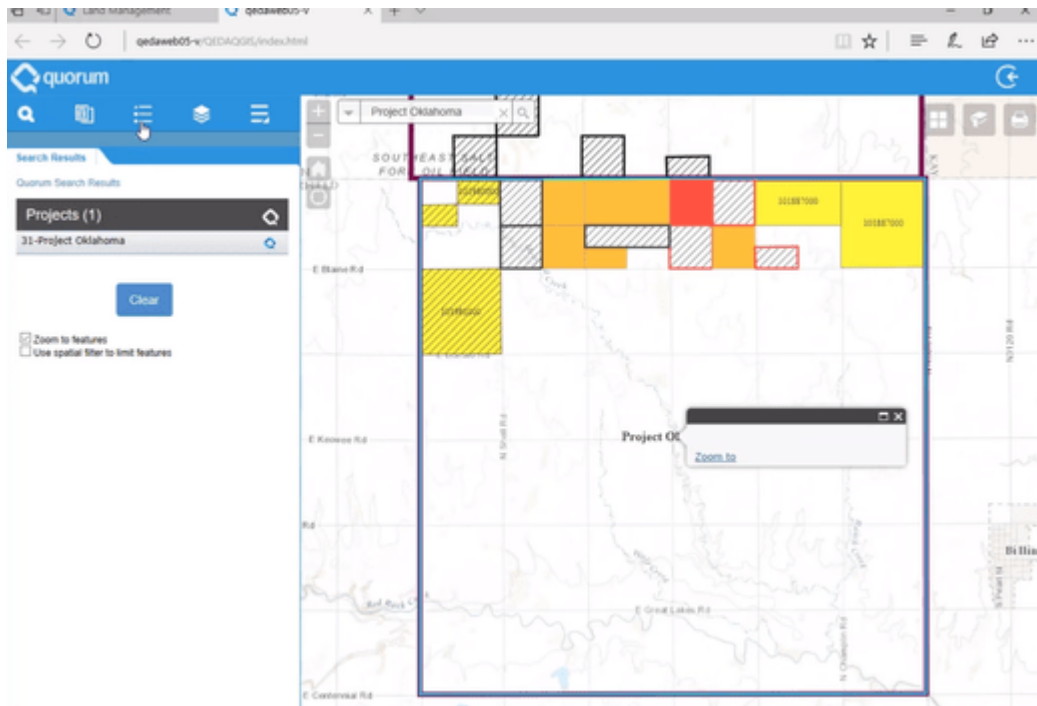


Рисунок 1.4 – Програмне забезпечення Land on Demand [9]

Початкове налаштування земельної ділянки Quorum на вимогу набагато простіше, ніж будь-яке інше рішення щодо землі. Quorum має такі функції і переваги [9]:

- об'єднана модель даних, яка зменшує дублювання даних і подвійне введення, а також спрощує доступ до інформації;
- централізовані основні дані та загальні елементи даних допомагають людям у організації працювати з документами стосовно земельного обліку;
- відкриті API надають оперативні, фінансові та польові дані, які можна змішувати та представляти унікальним чином;
- інтегровані фінансові, бухгалтерські, операційні та польові дані покращують міжфункціональну видимість, що забезпечує більшу цінність даних;

– використовує адаптивні методології дизайну, щоб забезпечити узгоджений користувацький досвід у всіх сучасних веббраузерах і комп'ютерних платформах.

Програмне забезпечення Visual LandPro є дуже ретельною та довготривалою системою обліку земельних ділянок на певній території [10] (рис. 1.5).

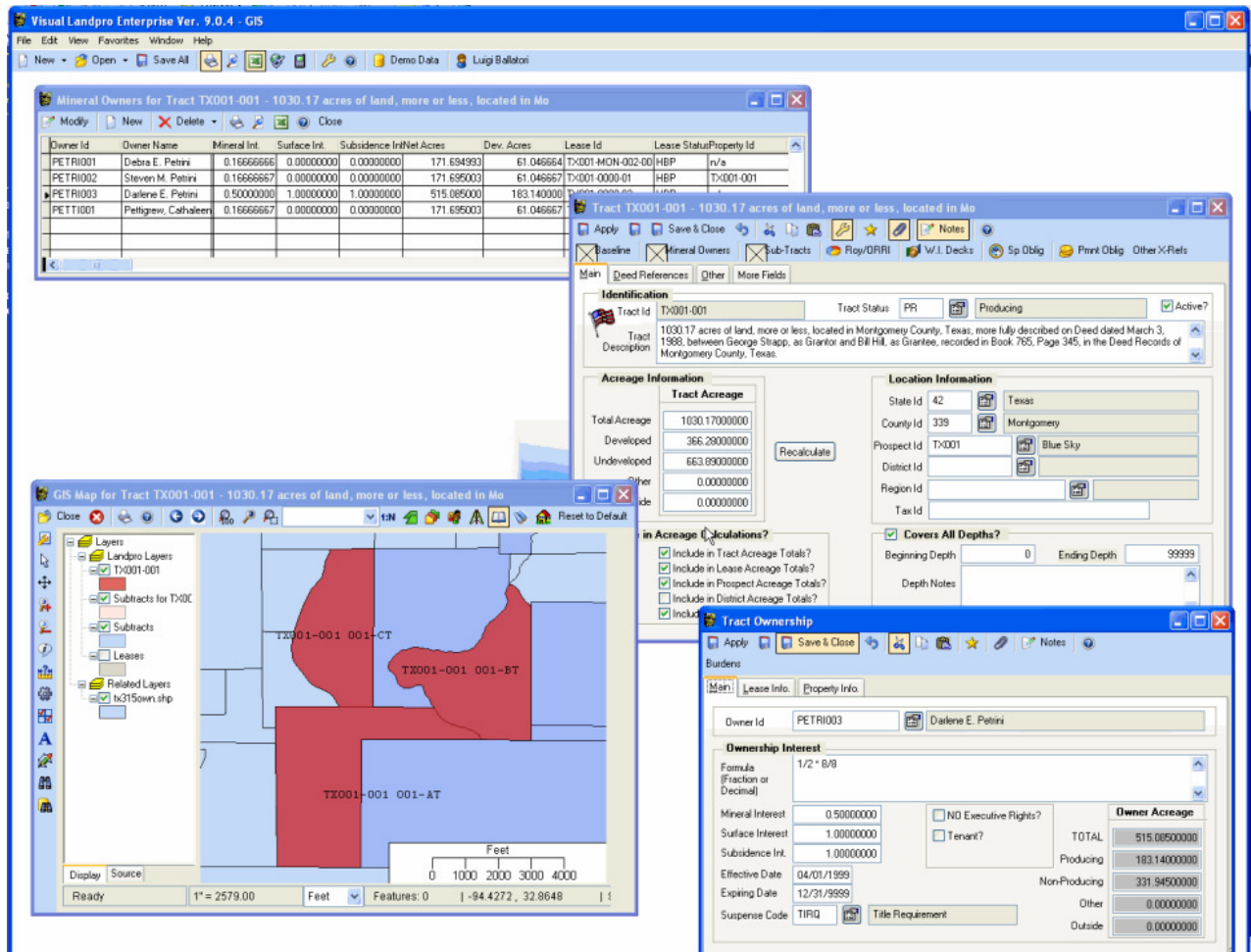


Рисунок 1.5 – Програмне забезпечення Visual LandPro [10]

Це програмне забезпечення містить усі аспекти фіксації земельних записів від перспективи до трубопроводу [10]. Регулярні оновлення підтримували актуальність програмного забезпечення, включаючи нещодавнє оновлення для вкрай необхідного серверного сервера SQL. Microsoft Excel є основним інструментом для введення наявних електронних даних. За серверну частину відповідає SQL, щоб забезпечити

масштабованість і збільшення швидкості в майбутньому [10]. Це програмне забезпечення подається як послуга з управління землею, яке включає в себе управління документами та велику кількість наскрізних робочих процесів для земельних ділянок, записів оренди, наказів підрозділів, відносин власників та ін. (рис. 1.6) [10]. Ця програма надає організаціям можливість створити шаблон завантаження та масове завантаження даних [10].

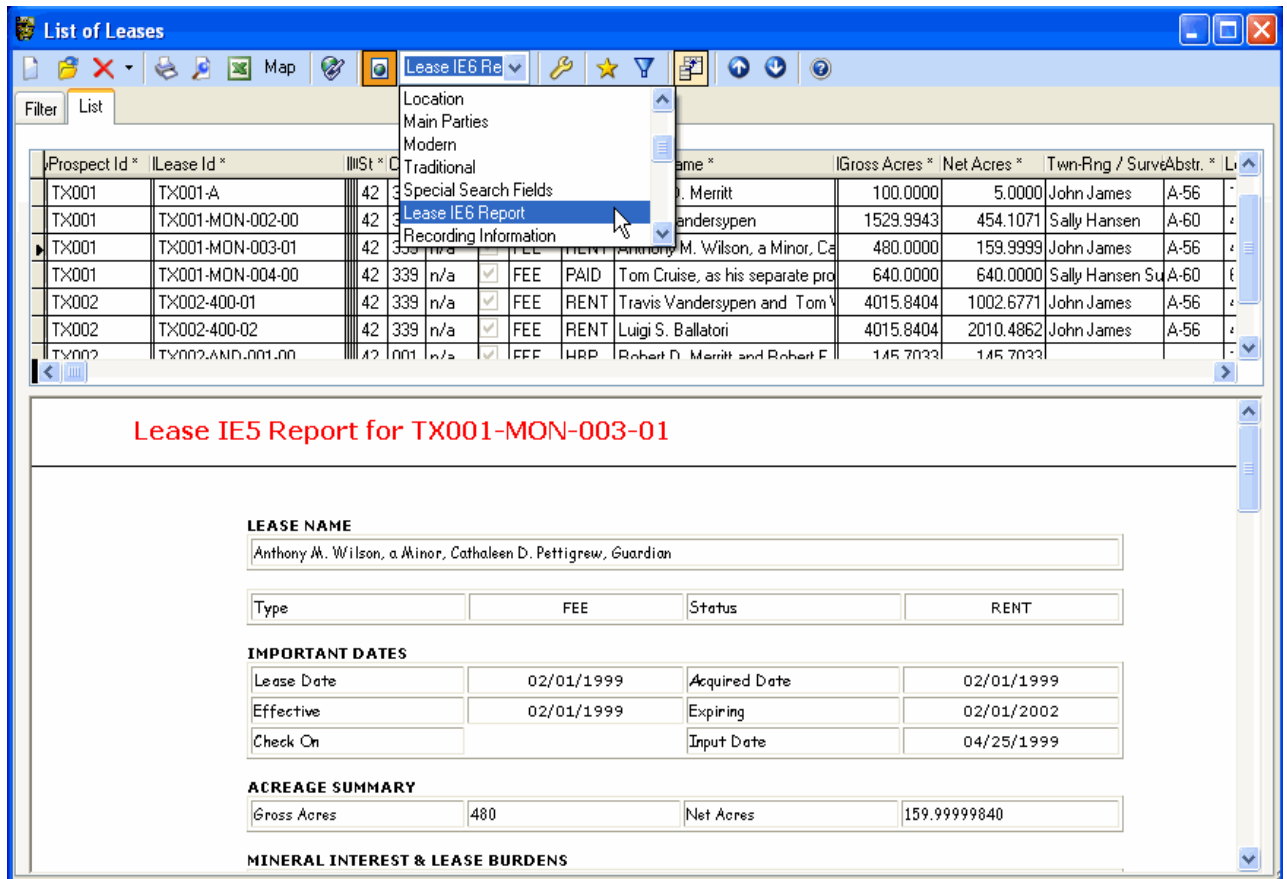


Рисунок 1.6 – Управління документами у програмному забезпеченні Visual LandPro [10]

Таким чином, у наш час на програмному ринку широко представлені програмні засоби для обліку земельних ділянок. Порівняльний аналіз програмного забезпечення для обліку земельних ділянок наведено у табл. 1.1.

Таблиця 1.1 – Програмне забезпечення для обліку земельних ділянок

Критерії порівняння	Програмні засоби				
	Peleton LandView	P2 iLandman	Landdoh	MyQuorum Land	LandPro 2000
Карти	cloud-based	Arc-based	API	SQL	Map Suite GIS
Звітність	Query файли	Excel & PDF Формат звітів	SQL	SQL	Зручна звітність у додатку
Ліцензування	Немає обмежень	Підписка + безкоштовна версія	Продаж на основі вартості	Ліцензійна плата/кільк співробіт.	Немає обмежень
Платіжні зобов'язання	Фільтри для платіжної системи	Звіти в Excel	Неавтомат. Звіти для бухгалтерії	Quorum звіти	Звіти в Excel
Розрахунки порядку поділу	облік доходів	Геодезичні розрахунки	Розрахунки об'ємів праці та оплати	Розрахунки в порядку поділу об'ємів роботи	Розрахунків порядку поділу
Пакет розрахунків за спільні інтереси	JIB, GWI	Можна відмінити роялті	Фіксує колами витрат і доходів на хорошому рівні	Фіксує	Можна керувати холдингами
Платформа	Хмарне сховище, SaaS, Web-Based	Хмарне сховище, SaaS, Web-Based	Хмарне сховище, SaaS, Web-Based	Хмарне сховище, SaaS, Web-Based	Windows

Таким чином доцільно розробити програмне забезпечення обліку земельних ділянок, що дозволяє виконувати такі функції:

- додавання даних про земельні ділянки в базу даних;
- видалення даних про земельні ділянки;
- редагування даних про земельні ділянки;
- фільтрація та пошук даних про земельні ділянки в базі даних;
- експорт інформації в Microsoft Excel.

1.3 Висновки за розділом 1

Проаналізовано загальні поняття про облік земельних ділянок. Визначено, що важливим завданням організації і планування діяльності підприємств є раціональне поєднання і використання земельних ресурсів з метою одержання максимального прибутку. Тому створення сучасного земельного кадастру та програмного забезпечення для обліку земельних ділянок є складним та актуальним завданням.

Досліджено та проаналізовано програмні засоби для обліку земельних ділянок.

Визначено, що доцільно розробити програмне забезпечення обліку земельних ділянок, що дозволяє виконувати такі функції: додавання даних про земельні ділянки в базу даних, видалення даних про земельні ділянки, редагування даних про земельні ділянки, фільтрація та пошук даних про земельні ділянки в базі даних, експорт інформації в Microsoft Excel.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Вибір мови програмування та інших технологій розробки

Для написання програми було обрано мову програмування PHP. PHP (Hypertext Preprocessor) — це широко використовувана мова сценаріїв із відкритим кодом загального призначення. Синтаксис мови походить від C, Java та Perl. Вивчати PHP дуже просто. Перевага PHP полягає в тому, що він дозволяє веброзробникам швидко створювати динамічні вебсайти [11]-[12].

Ключовою перевагою PHP перед такими мовами, як Perl і C, є можливість створювати документи HTML за допомогою команд PHP. Переваги PHP такі [11]-[12]:

- отримати продукти, які можна легко замінити та обслуговувати в майбутньому, створюючи якісні вебдодатки за дуже короткий час;
- мова є легкою для навчання і в той же час може задовольнити потреби професійних програмістів.

PHP є широкорозповсюдженою мовою сценаріїв для веброзробки. Реалізація довідника PHP в даний час розробляється PHP Group [11]-[12]. Спочатку PHP означав Personal Home, але тепер PHP: Hypertext Preprocessor означає рекурсивну ініціалізацію [11]-[12]. Код PHP обробляється перекладачем PHP, реалізованим у вигляді програми, модуля, демона або програми Common Gateway Interface (CGI) на вебсервері [11]-[12]. Результати, які інтерпретуються та виконуються кодом PHP на вебсервері (який може бути будь-яким типом даних, наприклад згенерованим HTML або двійковими зображеннями), складають відповідь HTTP, повністю або частково [11]-[12].

Існує багато різних систем вебдизайну, систем керування вебконтентом та вебфреймворків, які допомагають створювати або редагувати програмні засоби на PHP. Крім того, PHP можна використовувати для багатьох завдань, не пов'язаних з вебпрограмуванням, таких як автономні графічні програми та роботизоване керування дроном [11]-[12].

PHP-код можна запустити безпосередньо з командного рядка. Стандартний перекладач PHP, що працює на платформі Zend Engine, є безкоштовним програмним забезпеченням, випущеним під ліцензією PHP. PHP широко доступний і може використовуватися на більшості вебсерверів на різних операційних системах і платформах [11]-[12]. PHP розроблявся без написання формального визначення чи стандарту, і його перша реалізація слугувала де-факто стандартом, якого повинні дотримуватися інші програми. У наш час PHP використовується у більшості вебсайтів, які знають мову програмування на стороні сервера. Цільові змінні у цій мові можна призначити за допомогою десяткових чисел (додатних і від’ємних), вісімкових, шістнадцяткових і двійкових чисел. Кількість плаваючих точок також зберігається в діапазоні, що залежить від платформи. Вони можуть бути представлені двома способами: знаками з плаваючою комою або науковими знаками. PHP має свій власний логічний тип, подібний до основних логічних типів у Java та C++. Використовуючи конвенцію про перетворення логічних типів, ненульові значення в Perl і C++ інтерпретуються як істина, а нуль як неправда [11]-[12].

Нульовий тип даних представляє важливу змінну; NULL є єдиним дозволеним значенням для цього типу даних [11]-[12]. Змінна ресурсу – це посилання на ресурс із зовнішнього ресурсу [11]-[12]. Зазвичай вони створюються спеціальною функцією розширення і можуть бути оброблені лише однією функцією розширення [11]-[12]. Масиви можуть містити будь-який елемент, який може обробляти PHP, наприклад ресурси, об’єкти та навіть інші масиви [11]-[12].

Стандартна бібліотека PHP (SPL) намагається вирішити стандартні проблеми та реалізує ефективні інтерфейси та класи доступу до даних. PHP використовується в системах керування вебконтентом [11]-[12].

При розробці програмного забезпечення у вигляді вебзастосунків, зокрема і програмного забезпечення обліку земельних ділянок, доводиться працювати з CSS (Cascading Style Sheets) – каскадними таблицями стилів.

Стиль – набір параметрів для налаштування зовнішнього представлення об'єкта. Кожен елемент сторінки може мати свій власний стиль (абзац, заголовок, рядок, текст). Пакет стилів для всіх елементів називається таблицею стилів. Якщо для одного елемента вказано кілька стилів (як у прикладі заголовка), використовується порядок пріоритету проєктування [7]. Переваги CSS: дозволяє значно зменшити розмір коду і зробити його читабельним, дозволяє встановлювати параметри, які не можна встановити тільки в HTML [11]-[12].

Розробка програмного забезпечення у вигляді вебзастосунків, зокрема і програмного забезпечення обліку земельних ділянок, пов'язана з необхідністю роботи з HTML-сторінками. HTML (Hypertext Markup Language) – спеціальний посібник для браузера, який використовується для створення вебсайтів [11]-[12]. Вебсторінки — це документи HTML, які містять текст і спеціальний тег HTML (інтерпретатор). Загалом, HTML-теги потрібні для форматування тексту, який браузер розуміє (тобто, щоб надати йому потрібний вигляд). Документи HTML зберігаються як розширення [11]-[12].

Деякі скрипти при розробці програмного забезпечення обліку земельних ділянок створюються за допомогою мови JavaScript, яка є об'єктно-орієнтованою мовою програмування сценаріїв. JavaScript має кілька особливостей об'єктно-орієнтованої мови [11]-[12]. Прототипування відрізняється від роботи з об'єктами в порівнянні з традиційними об'єктно-орієнтованими мовами. Крім того, JavaScript має ряд функцій, які застосовуються до функціональних мов — першокласні об'єкти, списки об'єктів, анонімні функції та блокування забезпечують додаткову гнучкість мови [11]-[12]. Хоча синтаксис схожий, JavaScript принципово відрізняється від C [11]-[12].

У таблиці 2.1 наведено результати аналізу різних мов програмування для розробки програмного забезпечення обліку земельних ділянок.

Таблиця 2.1 – Порівняння мов програмування для створення ПЗ обліку земельних ділянок

Критерій порівняння мов програмування	Мова програмування		
	PHP	JavaScript	C#
Розташування скриптів на стороні сервера	+	–	+
Чутливість до регістру змінних	+–	+	+
Можливість роботи з асоціативними масивами	+	–	+
Обробка критичних ситуацій	+	+–	+
Взаємодія з базами даних	+	+–	+
Підтримка багатопоточності	+	–	+
Зручність для створення застосунку обліку земельних ділянок	+	+–	+–

Таким чином для програмної реалізації програми обліку земельних ділянок обрано мову програмування PHP.

2.2 Вибір системи керування базою даних

Для створення програми обліку земельних ділянок як система керування базами даних обрана MySQL [13]-[15].

MySQL — це система керування базами даних реляційного типу. Реляційні бази даних зберігають дані в окремих таблицях [13]-[15]. Ця СКБД складається з двох частин: сервера і клієнта [13]-[15]. Технічні характеристики MySQL — це клієнт-серверна система з багатопоточним сервером SQL, який підтримує широкий спектр комп'ютерних баз даних, а також кілька різних клієнтських програм, бібліотек, інструментів керування

та повного програмного інтерфейсу (API) [13]-[15]. Існує також можливість підключити сервер MySQL до спеціальної програми у вигляді багатопотокової бібліотеки, щоб отримати компактний, швидкий і простий у керуванні продукт [13]-[15].

Перевагами MySQL є такі: MySQL краще підходить для роботи в середовищі Інтернету, широкі можливості повнотекстового пошуку, масштабованість (можна об'єднати кілька комп'ютерів в один кластер), підтримка всіх популярних платформ, вимагає менше ресурсів, не вимагає великої кількості коштів на її застосування [13]-[15].

MySQL — це система управління контактною інформацією з відкритим вихідним кодом [13]-[15]. Назва є комбінацією «Му», дочки засновника Майкла Віденіуса, і «SQL», скорочення від Structured Query Language. Діалогова база даних упорядковує дані в одну або кілька таблиць даних, які можуть зв'язувати типи даних [13]-[15]. Ці відносини допомагають структурувати дані. SQL — це мова, яку використовують програмісти для створення, модифікації, вилучення й керування доступом користувачів із комунікаційних баз даних до баз даних [13]-[15]. На додаток до комунікаційних баз даних і SQL, такі бази даних, як MySQL з операційною системою, призначені для включення баз даних у системи зберігання комп'ютерів, керування користувачами, забезпечення доступу до мережі та забезпечення цілісності бази даних і резервного копіювання [13]-[15].

MySQL є безкоштовним програмним забезпеченням з відкритим кодом і доступне за різними ліцензіями власності. MySQL має автономні клієнти, які дозволяють користувачам безпосередньо взаємодіяти з базою даних MySQL за допомогою SQL, але MySQL часто використовується в поєднанні з іншими програмами для реалізації програм, які потребують навичок спілкування з базою даних [13]-[15]. MySQL є компонентом стеку вебпрограм LAMP (та інших), скорочено Linux, Apache, MySQL, Perl / PHP / Python. Він використовується багатьма вебдодатками баз даних, такими як MySQL, Drupal, Joomla, phpBB і WordPress. MySQL також використовується

багатьма популярними вебсайтами. MySQL написаний на C і C++ [13]-[15]. Парсер SQL написаний на уасс, але використовує саморобний парсер слів. Він працює на MySQL, AIX, BSDi, FreeBSD, HP-UX, ArcaOS, eComStation, IBM i, IRIX, Linux, macOS, Microsoft Windows, NetBSD, Novell NetWare, OpenBSD, OpenSolaris, OS/2 Warp, QNX та багатьох інших системах платформи. OpenVMS також має порт MySQL [13]-[15]. Серверне програмне забезпечення MySQL та клієнтські бібліотеки використовують подвійне розповсюдження ліцензій [13]-[15].

У таблиці 2.2 наведено порівняння систем керування базами даних для розробки програмного забезпечення обліку земельних ділянок.

Таблиця 2.2 – Порівняння систем керування базами даних для розробки програмного забезпечення обліку земельних ділянок

Критерій порівняння систем керування базами даних	Системи управління базами даних		
	MySQL	MS Access	MongoDB
Зручність для створення ПЗ обліку земельних ділянок	+	+-	+-
Типи даних, що підтримуються	структуровані, неструктуровані	структуровані, неструктуровані	неструктуровані
Наявність документації у вільному доступі	+	+-	+-
Ліцензія	GNU	приватна	SSPL
Зручність для обробки великих даних	+	+-	+-

Таким чином для розробки програмного забезпечення обліку земельних ділянок обрано систему управління базами даних MySQL.

2.3 Вибір вебсерверу для розташування програмного забезпечення

Для розташування програмного забезпечення обліку земельних ділянок обрано вебсервер Apache [13]-[15].

Apache HTTP Server — це безкоштовний вебсервер, що представляє собою кросплатформне програмне забезпечення. Apache підтримує такі операційні системи: BSD, Microsoft Windows, Linux, Mac OS, BeOS, Novell NetWare [13]-[15].

Основними перевагами Apache є надійність і гнучкість конфігурації. Він дозволяє підключати зовнішні модулі для надання даних, використовувати базу даних для автентифікації користувачів і маніпулювати повідомленнями про помилки [13]-[15]. HTTP-сервер Apache — це UNIX-подібний відкритий Інтернет-сервер, розроблений для Microsoft Windows, Novell NetWare та інших операційних систем. Сьогодні це найбільш використовуваний вебсервер в Інтернеті [13]-[15]. Apache розробляється та підтримується спільнотою розробників програмного забезпечення з відкритим кодом, яку керує Apache Software Foundation [13]-[15].

Вебсервер Apache є окремим, некомерційним продуктом, який вільно поширюється [13]-[15]. Продукт підтримує багато функцій, реалізованих за допомогою скомпільованих модулів, щоб розширити більшість основних функцій [13]-[15]. Ці мови програмування варіюються від підтримки на стороні сервера до схем автентифікації. Існують інтерфейси, які підтримують Perl, Python і PHP [13]-[15]. Apache в основному використовується для передачі статичних і динамічних вебсторінок через всесвітню мережу. Багато вебдодатків розроблено з урахуванням цього середовища вебсервера та його можливостей [13]-[15].

Цей продукт може виконувати роль проксі-сервера кешу, що може значно підвищити продуктивність користувачів локальної мережі під час

роботи з документами, знайденими в Інтернеті [13]-[15]. Можна використовувати такі параметри роботи сервера [13]-[15]:

- типи файлів, які необхідно кешувати або навпаки;
- максимальний дисковий простір для кешування;
- періодичний перегляд та індексування бази даних кешу, щоб звільнити місце на диску шляхом видалення старих об'єктів.

Apache зіграв важливу роль у ранньому розвитку всесвітньої павутини і залишається найпопулярнішим вебсервером у світі, орієнтованим на інші вебсервери [13]-[15]. Функції віртуального хостингу дозволяють одній установці Apache обслуговувати різні вебсайти. Наприклад машина з однією інсталяцією Apache може одночасно містити різні вебсайти [13]-[15].

Таким чином, для розташування програмного забезпечення обліку земельних ділянок обрано вебсервер Apache.

2.4 Висновки за розділом 2

Для створення програмного забезпечення обліку земельних ділянок було обрано мову програмування Php, яка є широко використовувана мова сценаріїв із відкритим кодом загального призначення, ключовою перевагою PHP є можливість створювати документи HTML за допомогою команд PHP.

Для створення програми обліку земельних ділянок як система керування базами даних обрана MySQL, система керування базами даних реляційного типу.

Для розташування програмного забезпечення обліку земельних ділянок обрано вебсервер Apache.

3 ОПИС ПРОГРАМИ

3.1 База даних програмного забезпечення обліку земельних ділянок

Для розробки програмного продукту обліку земельних ділянок була використана реляційна модель. У цій моделі дані поділяються на кластери, які складають структуру таблиці. Структура цієї таблиці складається з окремих елементів даних, які називаються полями. Набір або група полів називається записом.

Кожен рядок, що містить дані, називається кортежем, а кожен стовпець зв'язку — атрибутом (терміни «запис» і «поле» використовуються на практичному рівні з сучасними базами даних зв'язку). Елементами, які описують модель даних відносин на концептуальному рівні, є об'єкти, властивості, поля та зв'язки. Сутність — це ізольований об'єкт або подія, яка має зберігатися в базі даних із великою кількістю властивостей. Домен — це набір усіх дійсних значень, які може містити властивість [13]-[15]. Зв'язок — на концептуальному рівні існує простий зв'язок між сутностями [13]-[15].

База даних зв'язку на фізичному рівні складається з таблиць, між якими можна знайти ключові зв'язки. Правила зіставлення зовнішніх ключів із за замовчуванням є основними правилами для дотримання цілісності посилання. Для кожного значення зовнішнього ключа має бути значення первинного ключа, що відповідає базовій таблиці [13]-[15].

База даних розроблена у форматі MySQL. Розроблена база даних містить чотири таблиці. База даних показана на рисунку 3.1.

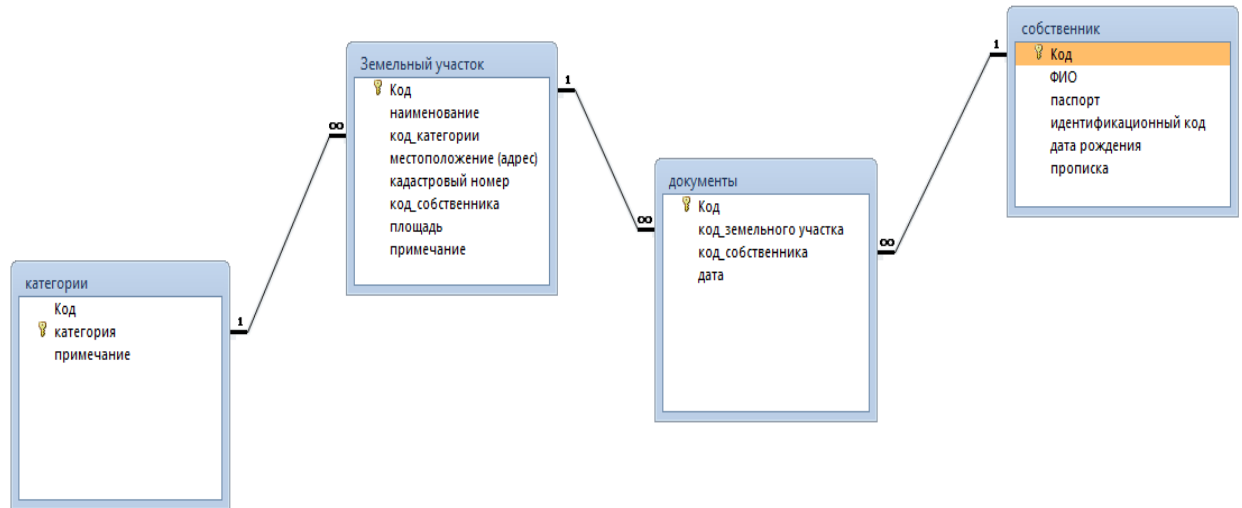


Рисунок 3.1 – Реляційна модель бази даних

Розглянемо таблиці «Category», «Dogovor», «Sobstvenik», «Zem_uch».

Сутність таблиці «Category» (id, name, status). Сутність призначена для збереження інформації про категорії земельних ділянок. В реалізації бази даних сутність представлено таблицею Category (табл. 3.1).

Таблица 3.1 – Структура таблиці Category

Поле	Тип	Нульове значення	За замовченням	Додатково
id	int(11)	Так	NULL	auto_increment
name	varchar(30)	Так		
status	varchar(15)	Так		

Сутність таблиці «Dogovor» (id, id_zem, id_sobstv, date). Сутність призначена для збереження інформації о договорах.

В реалізації бази даних сутність представлено таблицею Dogovor (табл. 3.2).

Таблиця 3.2 – Структура таблиці Dogovor

Поле	Тип	Нульове значення	За замовченням	Додатково
id	int(11)	Так	NULL	auto_increment
id_zem	int(11)	Так		
id_sobstv	int(11)	Так		
date	date	Так		

Сутність таблиці «Sobstvenik» (id, fio, passport, indnomer, date, propiska). Сутність призначена для збереження інформації о власниках. В реалізації бази даних сутність представлено таблицею Sobstvenik (табл. 3.3).

Таблиця 3.3 – Структура таблиці Sobstvenik

Поле	Тип	Нульове значення	За замовченням	Додатково
id	int(11)	Так	NULL	auto_increment
fio	varchar(30)	Так		
passport	varchar(30)	Так		
indnomer	varchar(30)	Так		
date	date	Так	NULL	
propiska	varchar(30)	Так		

Сутність таблиці «Zem_uch» (id, name, id_kat, place, kad_nomer, id_sobstv, square, status). Сутність призначена для зберігання інформації о земельних ділянках. В реалізації бази даних сутність представлено таблицею Zem_uch (табл. 3.4).

Таблиця 3.4 – Структура таблиці Zem_uch

Поле	Тип	Нульове значення	За замовченням	Додатково
id	int(11)	Так	NULL	auto_increment
name	varchar(45)	Так		
id_kat	int(11)	Так		
place	varchar(45)	Так		
kad_nomer	int(11)	Так		
id_sobstv	int(11)	Так		
square	int(11)	Так		
status	int(11)	Так		

Таким чином, виділено основні сутності, які подано у вигляді таблиць бази даних та дозволяють зберігати інформацію, необхідну для обліку земельних ділянок.

3.2 Структурна схема програми

Розроблений програмний продукт для обліку земельних ділянок може бути подано у вигляді фізичної моделі програми (рис. 3.2) та загальної структури програми.

Фізична модель програми обліку земельних ділянок відображає впорядкований набір файлів різного типу (HTML-сторінки, зображення, мультимедійні файли).

Фізична структура програми обліку земельних ділянок представлена на рисунку 3.2.

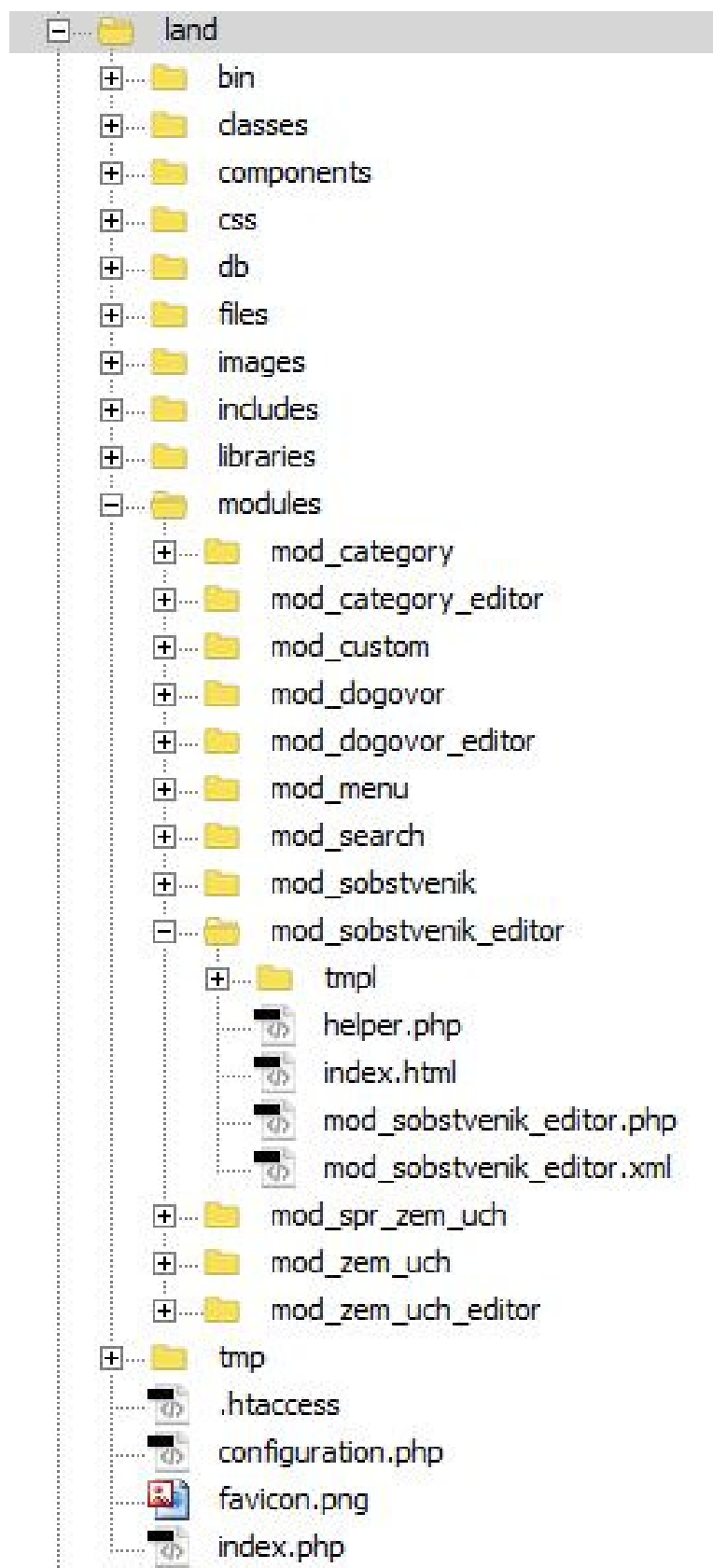


Рисунок 3.2 – Фізична структура програми обліку земельних ділянок

На рис. 3.3 наведено загальну структуру програми обліку земельних ділянок.

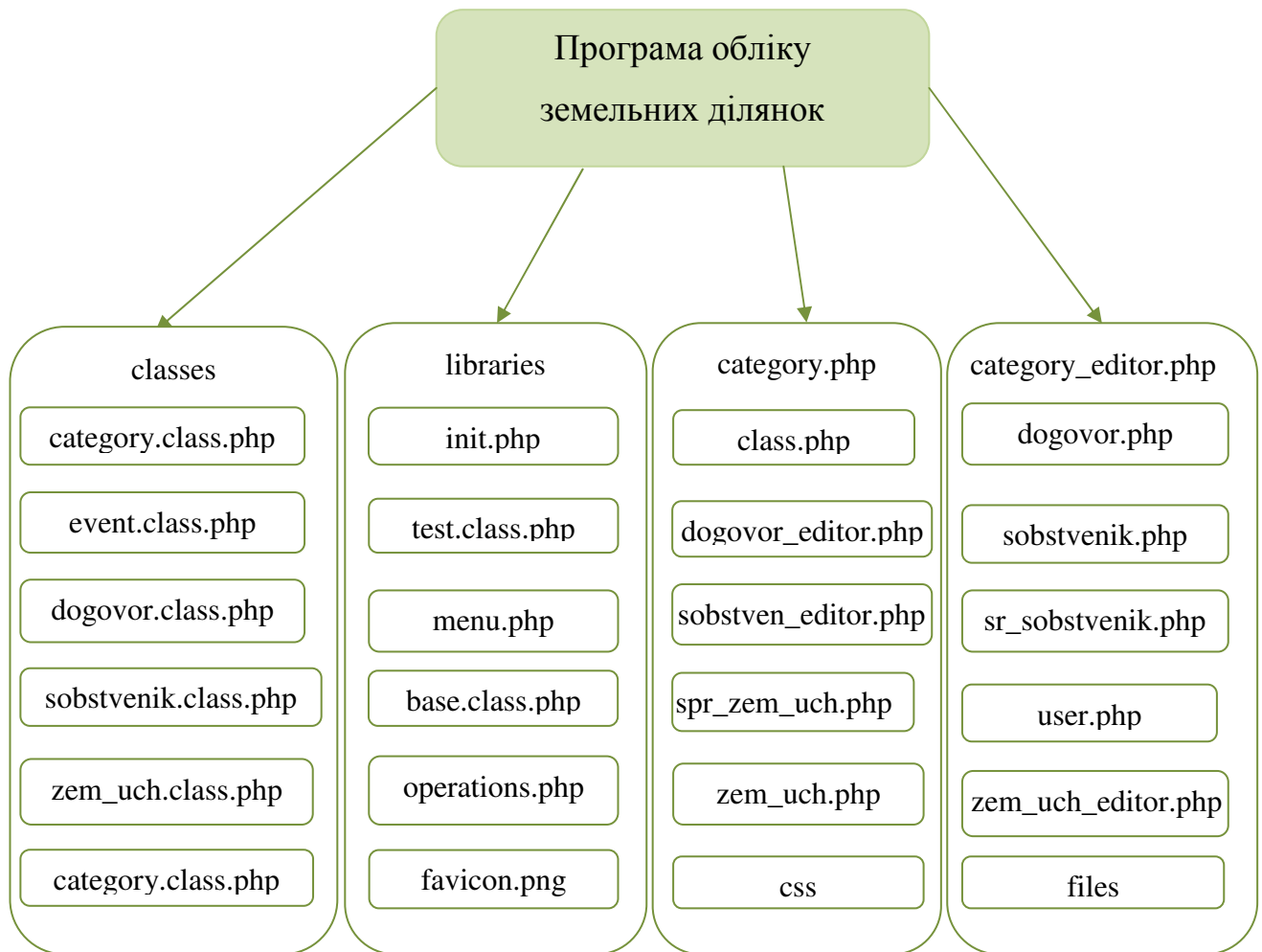


Рисунок 3.3 – Загальна структура програми

Таким чином, запропонована загальна структури програми обліку земельних ділянок відображає сукупність основних сторінок програмного забезпечення, поданих у вигляді php-модулів та об'єднаних єдиним дизайном, стилем і посиланнями (рис. 3.3).

3.3 Основні модулі програми обліку земельних ділянок

Програмне забезпечення обліку земельних ділянок представляє собою клієнт-серверний застосунок, де клієнт — браузер, а сервер — вебсервер. Логіка вебзастосунку розподілена між сервером і клієнтом, а дані зазвичай

зберігаються на сервері та обмінюються по мережі. Однією з переваг цього підходу є те, що вебзастосунки є міжплатформними сервісами, оскільки клієнти не залежать від операційної системи користувача. Клієнт-сервер – обчислювальна або мережева архітектура, яка розподіляє завдання або мережеве навантаження між постачальниками послуг (сервісами), які називаються серверами, і користувачами послуг, які називаються клієнтами. Як правило, клієнти та сервери взаємодіють через комп’ютерну мережу і можуть мати різні фізичні пристрої та програмне забезпечення [11]-[15].

Аналіз і розробка програмного продукту обліку земельних ділянок проводилися в кілька етапів, зокрема: технічний дизайн та художнє проєктування. На етапі технічного дизайну було розроблено основні модулі програми. На цьому етапі в проєкті, який створювався з використанням мови програмування PHP5, було реалізовано каталоги `classe`, `db`, `libraries`, а також файли: `category.php`, `category_editor.php`, `dogovor.php`, `dogovor_editor.php`, `sobstvenik.php`, `sobstvenik_editor.php`, `spr_zem_uch.php`, `spr_zem_uch.php`, `use.php`, `zem_uch.php`, `zem_uch_editor.php`.

Детальний опис файлів наведено нижче:

- `category.php` – модуль, що визначає логіку меню «Категорія»;
- `category_editor.php` – модуль з кодом, який можна використовувати для редагування пункт меню «Категорія»;
- `dogovor.php` – модуль, що визначає логіку меню «Договори»;
- `dogovor_editor.php` – модуль, що містить код, який можна використовувати для редагування меню «Договори»;
- `sobstvenik.php` – модуль, що визначає логіку елемента «Власники»;
- `sobstvenik_editor.php` – модуль з кодом, який редагують «Власники»;
- `spr_zem_uch.php` – модуль, що містить каталог про земельні ділянки;
- `zem_uch.php` – зберігає інформацію про земельні ділянки;
- `zem_uch_editor.php` – модуль, що містить код редагування, який відкриває довідник "Земельні ділянки".

Художнє проєктування має важливе значення для забезпечення необхідних можливостей застосування системи обліку земельних ділянок. Метою художнього дизайну є створення чіткого естетичного підходу до розробленої системи обліку земельних ділянок [13]-[15]. Ергономіка є важливою складовою якості програмного забезпечення обліку земельних ділянок. При розробці користувальницького інтерфейсу системи обліку земельних ділянок звертали увагу на ергономічні вимоги та художню концепцію програмного продукту обліку земельних ділянок. Система пропонує користувачам різноманітні коментарі та повідомлення. Якщо користувач спробує ввести дані в неправильному форматі, програма обліку земельних ділянок їх не прийме [13]-[15]. Після цього користувачеві дозволяється повторно ввести інформацію. Коли виконується багато дій користувача, програма обліку земельних ділянок вимагає їх аутентифікації [13]-[15]. Всі перераховані вище фактори значно полегшують роботу користувача програмного забезпечення обліку земельних ділянок, тим самим підвищуючи його продуктивність.

3.4 Висновки за розділом 3

Розроблено реляційну модель бази даних для створення програмного забезпечення обліку земельних ділянок. Виділено основні сутності, які подано у вигляді таблиць бази даних та дозволяють зберігати інформацію, необхідну для обліку земельних ділянок.

Запропоновано фізичну модель програми обліку земельних ділянок, яка відображає впорядкований набір файлів різного типу (HTML-сторінки, зображення, мультимедійні файли). Розроблено загальну структуру програми обліку земельних ділянок, яка відображає сукупність основних сторінок програмного забезпечення, поданих у вигляді php-модулів та об'єднаних єдиним дизайном, стилем і посиланнями.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Призначення й умови застосування програми

Програма призначена для автоматизації системи введення обліку землевласників. Програма написана на мові PHP. База даних створена в MySQL.

Програмні засоби обліку земельних ділянок забезпечують виконання функцій:

- додавання даних про земельні ділянки в базу даних;
- видалення даних про земельні ділянки;
- редагування даних про земельні ділянки;
- фільтрація та пошук даних про земельні ділянки в базі даних;
- експорт інформації в Microsoft Excel.

Для експлуатації програмних засобів обліку земельних ділянок необхідно таке апаратне забезпечення:

- процесор 1 GHz та вище;
- оперативна пам'ять 256Мб і вище;
- наявність вільного простору на жорсткому диску 150 Мб;
- монітор, клавіатура та маніпулятор типу “миша”.

Також повинна бути у наявності операційна система.

Функціональні характеристики розробленого програмного забезпечення обліку земельних ділянок відповідають технічному завданню, наведеному у додатку А. Опис програми обліку земельних ділянок та текст програми обліку земельних ділянок в наведені у додатках Б та В.

4.2 Характеристики програми

Серверна частина програми забезпечує збереження даних на сервері.

Клієнтська частина отримує данні від сервера та забезпечує перегляд даних у вікні програми, збереження та відправлення його назад до WEB–сервера.

Для роботи програми потрібна база даних, розміщена на Web – сервері, підключення до Інтернету або локальна мережа зі встановленим вебсервером Apache, та браузер для перегляду програми.

4.3 Інструкція по експлуатації програми

4.3.1 Звернення до програми

Для звернення до файлу програми потрібно в адресній строці браузера написати доменне ім'я програми обліку земельних ділянок.

4.3.2 Вхідні й вихідні дані

Вхідними даними до програми обліку земельних ділянок є дані, що зберігаються у файлі бази даних.

Вихідними даними програми обліку земельних ділянок є звіти з бази даних, які можна продивитися або роздрукувати.

4.3.3 Повідомлення

Розроблена програма обліку земельних ділянок забезпечує коректне функціонування на різних платформах. При некоректному заповненні полей форми видається відповідне повідомлення. Заявка від користувача програми відправляється на сервер бази даних лише при правильному заповненні даних про земельну ділянку.

4.4 Тестування та результати роботи програмного забезпечення обліку земельних ділянок

Для запуску програмного продукту обліку земельних ділянок необхідно відкрити та браузер та набрати в адресній написати доменне ім'я програми обліку земельних ділянок.

На головній сторінці (рис. 4.1) представлено головне меню програми. При розробці програмного продукту за бажанням замовника програма була розроблена на російській мові.

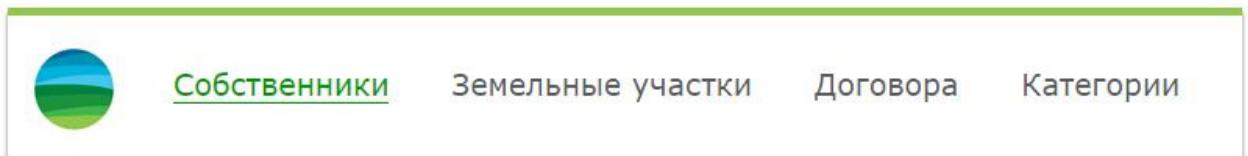


Рисунок 4.1 – Вигляд головного вікна вебзастосунку обліку земельних ділянок

Головне меню включає до себе наступні пункти:

- меню “Собственники”;
- меню “Земельные участки”;
- меню “Договора”;
- меню “Категории”.

Пункт меню «Собственники» призначений для відображення інформації про власників земельних ділянок («Дата рождения», «Идентификационный номер», «Паспорт», «Прописка»), також реалізована можливість редагування та видалення даних (рис. 4.2).

Для пошуку інформації по власникам потрібно написати П.І.Б. в спеціальне поле для вводу і отримати бажаний результат (рис. 4.3).

Для додавання інформації по власникам потрібно заповнити відповідні поля для вводу зберегти та отримати бажаний результат (рис. 4.4).

СОБСТВЕННИКИ

Поиск сведений об указанном владельце:

ФИО: ИСКАТЬ

№	ФИО	Дата рождения	Идентификационный номер	Паспорт	Прописка	Операции
1	Зинченко Анна Владимировна	1990-20-06	3336669965	СЮ02065289	г.Запорожье ул.40 лет Сов. Укр 64, кв.5	удалить добавить
2	Вольнец Ольга Валерьевна	1989-20-06	2316369565	СЮ02165385	г.Запорожье ул.Мира 10, кв.5	удалить добавить
3	Ивахнина Рита Владимировна	1965-15-01	2326669265	СЮ04015288	г.Мелитополь, б-р. 30 лет Победы, 1/72	удалить добавить
4	Сенцов Олег Владимировна	1990-20-06	3336669965	СЮ02065289	г.Бердянск, ул. Шевченко, 12, кв. 4	удалить добавить
5	Зинченко Анна Владимировна	1990-20-06	3336669965	СЮ02065289	г.Мелитополь, пр-т Б Хмельницкого, 24, кв. 432	удалить добавить

 ДОБАВИТЬ

1 2 3 4

 Экспорт в Excel

Рисунок 4.2 – Меню программы «Собственники»

СОБСТВЕННИКИ

Поиск сведений об указанном владельце:

ФИО: ИСКАТЬ

Рисунок 4.3 – Пошук інформації

СОБСТВЕННИКИ

ФИО: Дата рождения: Паспорт: Прописка: Идентификационный номер:

СОХРАНИТЬ

Рисунок 4.4 – Додавання інформації

Пункт меню «Договора» призначений для вибору земельних ділянок та власників (рис. 4.5).

№	Название участка	Местоположение участка	Кадастровый номер участка	Площадь
1	Земли для с/г назначения	Украина, г.Бердянск, ул. Шевченко 55,	2620455100:01:001:0261	0.1001 га
2	Земли для введения товарного с/г производства	Украина, г.Запорожье, ул. Набережная 19	2620455100:01:001:1246	0.15 га
3	Земли поселений	Украина, г.Запорожье, ул. 8 марта, 96	2620455100:01:001:0516	0.1215 га
4	Земли поселений	Украина, г.Мелитополь, ул.Залагерсер, 85	2620455100:01:004:0738	удалить добавить
5	Земли поселений	Украина, г.Мелитополь, ул.Гоголя, 16	2620455100:01:001:1014	удалить добавить

Рисунок 4.5 – Вибір інформації

При некоректному заповненні полей форми видається відповідне повідомлення (рис. 4.6). Заявка відправляється на сервер лише при правильному заповненні даних.

Рисунок 4.6 – Повідомлення оператору

Отже результати тестування програмного забезпечення обліку земельних ділянок підтвердили його працездатність.

4.5 Висновки за розділом 4

Описано програмне забезпечення обліку земельних ділянок. Виконано тестування розробленого програмного забезпечення обліку земельних ділянок. Результати тестування програмного забезпечення обліку земельних ділянок підтвердили його працездатність.

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи бакалавра було проаналізовано та досліджено процес розробки програмного забезпечення для обліку земельних ділянок.

Проаналізовано загальні поняття про облік земельних ділянок. Визначено, що важливим завданням організації і планування діяльності підприємств є раціональне поєднання і використання земельних ресурсів з метою одержання максимального прибутку. Тому створення сучасного земельного кадастру та програмного забезпечення для обліку земельних ділянок є складним та актуальним завданням.

Досліджено та проаналізовано програмні засоби для обліку земельних ділянок.

Визначено, що доцільно розробити програмне забезпечення обліку земельних ділянок, що дозволяє виконувати такі функції: додавання даних про земельні ділянки в базу даних, видалення даних про земельні ділянки, редагування даних про земельні ділянки, фільтрація та пошук даних про земельні ділянки в базі даних, експорт інформації в Microsoft Excel.

Для створення програмного забезпечення обліку земельних ділянок було обрано мову програмування Php, яка є широко використовувана мова сценаріїв із відкритим кодом загального призначення, ключовою перевагою PHP є можливість створювати документи HTML за допомогою команд PHP.

Для створення програми обліку земельних ділянок як система керування базами даних обрана MySQL, система керування базами даних реляційного типу.

Для розташування програмного забезпечення обліку земельних ділянок обрано вебсервер Apache.

Розроблено реляційну модель бази даних для створення програмного забезпечення обліку земельних ділянок. Виділено основні сутності, які

подано у вигляді таблиць бази даних та дозволяють зберігати інформацію, необхідну для обліку земельних ділянок.

Запропоновано фізичну модель програми обліку земельних ділянок, яка відображає впорядкований набір файлів різного типу (HTML-сторінки, зображення, мультимедійні файли). Розроблено загальну структуру програми обліку земельних ділянок, яка відображає сукупність основних сторінок програмного забезпечення, поданих у вигляді php-модулів та об'єднаних єдиним дизайном, стилем і посиланнями.

Описано програмне забезпечення обліку земельних ділянок. Виконано тестування розробленого програмного забезпечення обліку земельних ділянок. Результати тестування програмного забезпечення обліку земельних ділянок підтвердили його працездатність.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Дрововозов В. І. Землеустрій: Навчальний посібник / В. І. Дрововозов. – К.: НАУ, 2006. – 58 с.
2. Карпінський Ю. О. Інфраструктура геопросторових даних: принципи та методика формування базового набору геопросторових даних / Ю. О. Карпінський, А. А. Лященко // Вісник Криворізького технічного університету. – 2014. – Вип. 3. – С. 72–77.
3. Губар Ю. П. Геодезичне забезпечення та удосконалення методів та моделей оцінки нерухомості: монографія / Ю. П. Губар. – Львів: Видавництво «Центр Європи», 2020 – 326 с.
4. Лихогруд М. Г. Класифікація земель в автоматизованій системі державного земельного кадастру / М. Г. Лихогруд // Землевпорядний вісник. – 2002. – №1. –с. 16–22.
5. Свердлюк О. Застосування ГІС-технологій у сфері земельного кадастру та землеустрою / О. Свердлюк // Землевпорядний вісник. – 2006. – № 4. – С. 56 – 59.
6. Peloton [Electronic resource]. – Access mode : <https://www.peloton.com>.
7. P2 iLandMan [Electronic resource]. – Access mode : <https://www.p2energysolutions.com/p2-ilandman>.
8. Landdox [Electronic resource]. – Access mode : <https://www.landdox.com/>.
9. MyQuorum [Electronic resource]. – Access mode : <https://www.quorumsoftware.com/upstream-on-demand/land-management>.
10. Peloton Visual LandPro [Electronic resource]. – Access mode : <https://www.capterra.com/p/93455/Visual-LandPro-2000/>.
11. PHP Documentation [Electronic resource]. – Access mode : <https://www.php.net/docs.php>.

12. PHP Documentation [Electronic resource]. – Access mode : <https://devdocs.io/php/>.

13. MySQL Documentation [Electronic resource]. – Access mode : <https://www.mysql.com>.

14. API MySQL [Electronic resource]. – Access mode : <https://www.php.net/manual/ru/book.mysql.php>.

15. Comparing Database Management Systems: MySQL, PostgreSQL, MSSQL Server, MongoDB, Elasticsearch, and others” [Electronic resource]. – Access mode : <https://www.altexsoft.com/blog/business/comparing-database-management-systems-mysql-postgresql-mssql-server-mongodb-elasticsearch-and-others/>.

ДОДАТОК А
Технічне завдання

Вступ

Програмне забезпечення обліку земельних ділянок може використовуватися для обліку інформації про земельні ділянки та оцінювання ефективності результатів господарювання на них.

A.1 Підстава для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Програмне забезпечення обліку земельних ділянок», затверджене наказом Національного університету «Запорізька політехніка» № 102 від 27 квітня 2022 р.

A.2 Призначення розробки

Програмний продукт обліку земельних ділянок призначений для полегшення роботи працівників підприємств землеустрою, а також підприємств, для необхідно виконувати облік земельних ділянок.

A.3 Основні вимоги до програми, що розробляється

A.3.1 Вимоги до функціональних характеристик

Програмне забезпечення обліку земельних ділянок повинно включати функції:

- додавання даних про земельні ділянки в базу даних;
- видалення даних про земельні ділянки;
- редагування даних про земельні ділянки;
- фільтрація та пошук даних про земельні ділянки в базі даних;
- експорт інформації в Microsoft Excel.

А.3.2 Вимоги до інтерфейсу програми

Інтерфейс програми обліку земельних ділянок повинен бути зручним для користувачів. Взаємодія користувача з програмою обліку земельних ділянок можлива за допомогою вебсторінок.

А.3.3 Вимоги до надійності

Програмне забезпечення обліку земельних ділянок повинно забезпечити обробку проблемних ситуацій та видавати відповідні повідомлення у випадку вводу неправильних даних.

А.3.4 Умови експлуатації

Для використання ПЗ обліку земельних ділянок необхідно мати ПК чи ноутбук. Для роботи програми обліку земельних ділянок потрібна база даних про земельні ділянки, розміщена на вебсервері, необхідно підключення до мережі Інтернет, а також браузер для перегляду програми.

А.3.5 Вимоги до складу та параметрів технічний засобів

Для експлуатації програмного забезпечення обліку земельних ділянок необхідні такі засоби:

- процесор 1 GHz та вище;
- оперативна пам'ять 256Мб і вище;
- наявність вільного простору на жорсткому диску 150 Мб;
- монітор, клавіатура та маніпулятор типу “миша”.

Також у користувача повинна бути у наявності операційна система, а також підключення до мережі Інтернет.

А.3.6 Вимоги до маркування і пакування

Програма обліку земельних ділянок може бути записана як на компакт-диску чи на флешці. На впакуванні повинна бути назва програми – «Програмне забезпечення обліку земельних ділянок».

А.4 Вхідні дані до роботи

Вхідними даними до програми обліку земельних ділянок є дані, що зберігаються у файлі бази даних.

Вихідними даними програми обліку земельних ділянок є звіти з бази даних, які можна продивитися або роздрукувати.

ДОДАТОК Б
Опис програми

Б.1 Загальні відомості

Програмне забезпечення обліку земельних ділянок розроблено за допомогою мови програмування Php, використовується система керування базами даних MySQL, вебсервер Apache.

Б.2 Функціональне призначення

Програма призначена для автоматизації системи введення обліку землевласників.

Програмні засоби обліку земельних ділянок забезпечують виконання функцій:

- додавання даних про земельні ділянки в базу даних;
- видалення даних про земельні ділянки;
- редагування даних про земельні ділянки;
- фільтрація та пошук даних про земельні ділянки в базі даних;
- експорт інформації в Microsoft Excel.

Б.3 Опис логічної структури

Розроблений програмний продукт для обліку земельних ділянок може бути подано у вигляді фізичної моделі програми та загальної структури програми.

Фізична модель програми обліку земельних ділянок відображає впорядкований набір файлів різного типу (HTML-сторінки, зображення, мультимедійні файли). Загальна структури програми обліку земельних ділянок відображає сукупність основних сторінок програмного забезпечення, поданих у вигляді php-модулів та об'єднаних єдиним дизайном, стилем і посиланнями

Б.4 Використані технічні засоби

Для експлуатації програмного забезпечення обліку земельних ділянок необхідні такі засоби:

- процесор 1 GHz та вище;
- оперативна пам'ять 256Мб і вище;
- наявність вільного простору на жорсткому диску 150 Мб;
- монітор, клавіатура та маніпулятор типу “миша”.

Також у користувача повинна бути у наявності операційна система, а також підключення до мережі Інтернет.

Б.5 Виклик і завантаження

Для запуску програмного продукту обліку земельних ділянок необхідно відкрити та браузер та набрати в адресній написати доменне ім'я програми обліку земельних ділянок.

Б.6 Вхідні та вихідні дані

Вхідними даними до програми обліку земельних ділянок є дані, що зберігаються у файлі бази даних.

Вихідними даними програми обліку земельних ділянок є звіти з бази даних, які можна продивитися або роздрукувати.

ДОДАТОК В
Текст програми

```

package Connection;

import java.io.FileInputStream;
import java.sql.*;
import java.text.SimpleDateFormat;
import java.util.*;
import java.util.Date;
import java.sql.*;
import java.io.*;

import com.intersys.jdbc.*;

import ATP.Transport;
import ATP.Voyage;
import ATP.Worker;

public class CacheDB {

    private Connection myConn;
    List<Transport> list;
    List<Worker> listW;
    List<Voyage> listV;
    List<Transport> listT;
    //private String appropriateGovNumber;
    Transport approprTrans;

    public CacheDB()throws Exception{

        Class.forName("com.intersys.jdbc.CacheDriver");
        Properties props = new Properties();
        props.load(new FileInputStream("demo.properties"));
        String user=props.getProperty("user");
        String password=props.getProperty("password");
        String dburl=props.getProperty("dburl");
        CacheDataSource ds = new CacheDataSource();
        ds.setURL(dburl);
        ds.setUser(user);
        ds.setPassword(password);
        myConn = ds.getConnection();
        System.out.println("Db connection succesful to:" + dburl);
    }

    private void close(Statement myStmt, ResultSet myRs) throws SQLException {
        close(null, myStmt, myRs);
    }

    private static void close(Connection myConn, Statement myStmt, ResultSet myRs)
        throws SQLException {

        if (myRs != null) {
            myRs.close();
        }

        if (myStmt != null) {
        }

        if (myConn != null) {
            myConn.close();
        }
    }

    public void addTransport(String govNumber, String _weight, String _volume, String mark,
String model, String _cost, String _consumption)
    {
        PreparedStatement prStmn=null;
        ResultSet rs=null;
        int capacity, volume, cost, consumption;
        capacity = Integer.parseInt(_weight);
        volume = Integer.parseInt(_volume);
        cost = Integer.parseInt(_cost);
        consumption = Integer.parseInt(_consumption);
        try{
            prStmn=myConn.prepareStatement("insert into
Test.Transport (GovNumber,Capacity, Volume, Mark, Model, Cost, Consumption, IsFree)
values(?,?,?,?,?,?,?,?)");
            prStmn.setString(1, govNumber);
            prStmn.setInt(2, capacity);

```

```

        prStmn.setInt(3, volume);
        prStmn.setString(4, mark);
        prStmn.setString(5, model);
        prStmn.setInt(6, cost);
        prStmn.setInt(7, consumption);
        prStmn.setBoolean(8, true);
        prStmn.executeUpdate();
    } catch (SQLException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    } finally{
        try {
            close(prStmn,rs);
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}

public void deleteTransport(int id) throws SQLException
{
    PreparedStatement prStmn=null;
    ResultSet rs=null;
    try{
        prStmn=myConn.prepareStatement("delete from Test.Transport where
ID=?");

        prStmn.setInt(1, id);
        prStmn.executeUpdate();
    } finally{
        close(prStmn,rs);
    }
}

public void addWorker(String firstName, String name, String lastName, String _salary,
String _experience)
{
    PreparedStatement prStmn=null;
    ResultSet rs=null;
    int salary, experience;
    salary = Integer.parseInt(_salary);
    experience = Integer.parseInt(_experience);
    try{
        prStmn=myConn.prepareStatement("insert into Test.Worker( Name,
FirstName, LastName, Salary, Experience, IsFree) values(?, ?, ?, ?, ?, ?)");
        prStmn.setString(1, name);
        prStmn.setString(2, firstName);
        prStmn.setString(3, lastName);
        prStmn.setInt(4, salary);
        prStmn.setInt(5, experience);
        prStmn.setBoolean(6, true);
        prStmn.executeUpdate();
    } catch (SQLException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    } finally{
        try {
            close(prStmn,rs);
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}

public void deleteWorker(int id) throws SQLException
{
    PreparedStatement prStmn=null;
    ResultSet rs=null;
    try{
        prStmn=myConn.prepareStatement("delete from Test.Worker where
ID=?");

        prStmn.setInt(1, id);
        prStmn.executeUpdate();
    } finally{
        close(prStmn,rs);
    }
}

public void searchFreeTransport(int capacity, int volume) throws Exception

```

```

    {
        SearchTransport(capacity, volume);
        System.out.println("list size: " + list.size());
        appropriateTransport();
    }

    public void SearchTransport(int capacity, int volume) throws Exception {
        list = new ArrayList<Transport>();
        PreparedStatement myStmn = null;
        ResultSet rs = null;
        try {
            myStmn = myConn.prepareStatement("select * from Test.Transport where Capacity >=
? and IsFree = ? and Volume >= ?");
            myStmn.setInt(1, capacity);
            myStmn.setBoolean(2, true);
            myStmn.setInt(3, volume);
            rs = myStmn.executeQuery();
            while(rs.next()) {
                Transport tempTransport = convertRowToTransport(rs);
                list.add(tempTransport);
            }
        } finally {
            close(myStmn, rs);
        }
    }

    public Transport convertRowToTransport(ResultSet rs) throws SQLException {
        {
            int ID = rs.getInt("ID");
            String govNumber = rs.getString("GovNumber");
            int capacity = rs.getInt("Capacity");
            int volume = rs.getInt("Volume");
            String mark = rs.getString("Mark");
            String model = rs.getString("Model");
            int cost = rs.getInt("Cost");
            int consumption = rs.getInt("Consumption");
            boolean isFree = rs.getBoolean("IsFree");

            Transport tempTransport = new Transport(ID, govNumber, capacity, volume, mark,
model, cost, consumption, isFree);

            return tempTransport;
        }

        public int getListSize()
        {
            int size = list.size();
            return size;
        }

        public void appropriateTransport() throws SQLException
        {
            int minCapacity = 400;
            for(int i = 0; i < list.size(); i++)
            {
                if(list.get(i).getCapacity() < minCapacity)
                {
                    minCapacity = list.get(i).getCapacity();
                    //appropriateGovNumber = list.get(i).getGovNumber();
                    apprprTrans = new Transport(list.get(i).getID(),
list.get(i).getGovNumber(), list.get(i).getCapacity(), list.get(i).getVolume(),
list.get(i).getMark(), list.get(i).getModel(), list.get(i).getCost(),
list.get(i).getConsumption(), list.get(i).getisFree());
                }
            }
            if(getListSize() >= 1)
            {
                System.out.println(apprprTrans.getID() + " " +
apprprTrans.getMark() + " " + apprprTrans.getModel());
                //updateTransport();
            }
        }

        public void updateTransport() throws SQLException
        {
            PreparedStatement prStmn = null;
            ResultSet rs = null;
            try {

```

```

ID=?");

        prStmn=myConn.prepareStatement("update Test.Transport set IsFree = ? where
ID=?");

        prStmn.setBoolean(1, false);
        prStmn.setInt(2, apprprTrans.getID());
        prStmn.execute();
    }finally{
        close(prStmn,rs);
    }
}

public String getMark()
{
    return apprprTrans.getMark();
}

public String getModel()
{
    return apprprTrans.getModel();
}

public int getCost()
{
    return apprprTrans.getCost();
}

public double getConsumption()
{
    return apprprTrans.getConsumption();
}

public void setVoyage(String _FIO, String _capacity, String _volume, String _from,
String _to, String _distance, double _money) throws SQLException
{
    searchFreeWorker();
    int workerID = listW.get(0).getID();
    updateWorker(workerID);
    PreparedStatement prStmn=null;
    ResultSet rs=null;
    int capacity = Integer.parseInt(_capacity);
    int volume = Integer.parseInt(_volume);
    int distance = Integer.parseInt(_distance);
    double moneyPlus = distance * 5;

    try{
        prStmn=myConn.prepareStatement("insert into Test.Voyage (FIO,
Capacity, Volume, From, To, Distance, Money, MoneyPlus, Name, FirstName, LastName, Active,
GovNumber) values(?,?,?,?,?,?,?,?,?,?,?,?)");
        prStmn.setString(1, _FIO);
        prStmn.setInt(2, capacity);
        prStmn.setInt(3, volume);
        prStmn.setString(4, _from);
        prStmn.setString(5, _to);
        prStmn.setInt(6, distance);
        prStmn.setDouble(7, _money);
        prStmn.setDouble(8, moneyPlus);
        prStmn.setString(9, listW.get(0).getName());
        prStmn.setString(10, listW.get(0).getFirstName());
        prStmn.setString(11, listW.get(0).getLastName());
        prStmn.setBoolean(12, true);
        prStmn.setString(13, apprprTrans.getGovNumber());
        prStmn.executeUpdate();
    }finally{
        close(prStmn,rs);
    }
    System.out.println( apprprTrans.getGovNumber());
}

public void searchFreeWorker() throws SQLException
{
    listW = new ArrayList<Worker>();
    PreparedStatement myStmn= null;
    ResultSet rs=null;
    try{
        myStmn=myConn.prepareStatement("select * from Test.Worker where IsFree = ?");
        myStmn.setBoolean(1, true);
        rs=myStmn.executeQuery();
        while(rs.next()){
            Worker tempWorker=convertRowToWorker(rs);
            listW.add(tempWorker);
        }
    }
}

```

```

    }
    finally{
        close(myStmn,rs);
    }
}

public Worker convertRowToWorker(ResultSet rs) throws SQLException
{
    int ID = rs.getInt("ID");
    String name = rs.getString("Name");
    String firstName = rs.getString("FirstName");
    String lastName = rs.getString("LastName");
    int experience = rs.getInt("Experience");
    int salary = rs.getInt("Salary");
    boolean isFree = rs.getBoolean("IsFree");

    Worker tempWorker = new Worker(ID, name, firstName, lastName, experience,
salary, isFree);

    return tempWorker;
}

public void updateWorker(int workerID) throws SQLException
{
    PreparedStatement prStmn=null;
    ResultSet rs=null;
    try{
        prStmn=myConn.prepareStatement("update Test.Worker set IsFree = ? where
ID=?");

        prStmn.setBoolean(1, false);
        prStmn.setInt(2, workerID);
        prStmn.execute();
    }finally{
        close(prStmn,rs);
    }
}

public void GetActiveVoyage()throws Exception{
    listV = new ArrayList<>();
    PreparedStatement myStmn=null;
    ResultSet myRs=null;
    try{
        myStmn=myConn.prepareStatement("select * from Test.Voyage where
Active=?");

        myStmn.setBoolean(1, true);
        myRs=myStmn.executeQuery();
        while(myRs.next()){
            Voyage tempVoyage=convertRowToVoyage(myRs);
            listV.add(tempVoyage);
        }
    }finally{
        close(myStmn,myRs);
    }
}

public Voyage convertRowToVoyage(ResultSet rs) throws SQLException
{
    int ID = rs.getInt("ID");
    String FIO = rs.getString("FIO");
    int capacity = rs.getInt("Capacity");
    int volume = rs.getInt("Volume");
    String from = rs.getString("From");
    String to = rs.getString("To");
    int distance = rs.getInt("Distance");
    double money = rs.getDouble("Money");
    double moneyPlus = rs.getDouble("MoneyPlus");
    String name = rs.getString("Name");
    String firstName = rs.getString("FirstName");
    String lastName = rs.getString("LastName");
    boolean active = rs.getBoolean("Active");
    String govNumber = rs.getString("GovNumber");

    Voyage tempVoyage = new Voyage(ID,FIO, capacity, volume, from, to, distance,
money, moneyPlus, name, firstName, lastName, active, govNumber);

    return tempVoyage;
}

public List<Voyage> getList()

```

```

    {
        return listV;
    }

    public void changeVoyage(int id) throws SQLException
    {
        PreparedStatement prStmn=null;
        ResultSet rs=null;

        for(int i=0; i<listV.size(); i++)
        {
            if(listV.get(i).getID() == id)
                id = i;
        }

        try{
            prStmn=myConn.prepareStatement("update Test.Worker set IsFree = ? where
FirstName=? and Name=?");
            prStmn.setBoolean(1, true);
            prStmn.setString(2, listV.get(id).getFirstName());
            prStmn.setString(3, listV.get(id).getName());
            prStmn.execute();
        }finally{
            close(prStmn,rs);
        }
        try
        {
            prStmn=myConn.prepareStatement("update Test.Transport set IsFree = ?
where GovNumber=?");
            prStmn.setBoolean(1, true);
            prStmn.setString(2, listV.get(id).getGovNumber());
            prStmn.execute();
        }finally{
            close(prStmn,rs);
        }
        try
        {
            prStmn=myConn.prepareStatement("update Test.Voyage set Active=?
where ID=?");
            prStmn.setBoolean(1, false);
            prStmn.setInt(2, listV.get(id).getID());
            prStmn.execute();
        }finally{
            close(prStmn,rs);
        }
    }

    public void updateTransId(String id) throws SQLException
    {
        PreparedStatement prStmn=null;
        ResultSet rs=null;
        int id1 = Integer.parseInt(id);
        try{
            prStmn=myConn.prepareStatement("update Test.Transport set IsFree = ? where ID
= ?");
            prStmn.setBoolean(1, true);
            prStmn.setInt(2, id1);
            prStmn.execute();
        }finally{
            close(prStmn,rs);
        }
    }

    public void getAllTransport() throws SQLException
    {
        listT = new ArrayList<Transport>();
        PreparedStatement myStmn= null;
        ResultSet rs=null;
        try{

            myStmn=myConn.prepareStatement("select * from Test.Transport where IsFree = ?
");
            myStmn.setBoolean(1, true);
            rs=myStmn.executeQuery();
            while(rs.next()){
                Transport tempTransport=convertRowToTransport(rs);
                listT.add(tempTransport);
            }
        }
        finally{

```

```

        close(myStmn,rs);
    }
}

public List<Transport> getListTransport()
{
    return listT;
}
}

package ATP;

public class Transport {

    private int ID;
    private String govNumber;
    private int capacity;
    private int volume;
    private String mark;
    private String model;
    private int cost;
    private double consumption;
    private boolean isFree;

    public Transport(int ID, String govNumber, int capacity, int volume, String mark, String
model, int cost, double consumption, boolean isfree)
    {
        this.ID = ID;
        this.govNumber = govNumber;
        this.capacity = capacity;
        this.volume = volume;
        this.mark = mark;
        this.model = model;
        this.cost = cost;
        this.consumption = consumption;
        this.isFree = isfree;
    }

    public int getID()
    {
        return ID;
    }

    public String getGovNumber()
    {
        return govNumber;
    }

    public int getCapacity()
    {
        return capacity;
    }

    public int getVolume()
    {
        return volume;
    }

    public String getMark()
    {
        return mark;
    }

    public String getModel()
    {
        return model;
    }

    public int getCost()
    {
        return cost;
    }

    public double getConsumption()
    {
        return consumption;
    }
}

```

```

    }

    public boolean getisFree()
    {
        return isFree;
    }
}

package ATP;

public class Worker {

    private int ID;
    private String Name;
    private String FirstName;
    private String LastName;
    private int Experience;
    private int Salary;
    private boolean IsFree;

    public Worker(int ID, String name, String firstName, String lastname, int experience, int
salary, boolean isFree)
    {
        this.ID = ID;
        this.Name = name;
        this.FirstName = firstName;
        this.LastName = lastname;
        this.Experience = experience;
        this.Salary = salary;
        this.IsFree = isFree;
    }

    public String getName()
    {
        return Name;
    }

    public String getFirstName()
    {
        return FirstName;
    }

    public String getLastName()
    {
        return LastName;
    }

    public int getID()
    {
        return ID;
    }
}

package ATP;

public class Voyage {

    private int ID;
    private String FIO;
    private int capacity;
    private int volume;
    private String from;
    private String to;
    private int distance;
    private double money;
    private double moneyPlus;
    private String name;
    private String firstName;
    private String lastName;
    private boolean active;
    private String govNumber;

    public Voyage(int ID,String FIO, int capacity, int volume, String from, String to, int
distance, double money, double moneyPlus, String name, String firstName, String lastName, boolean
active, String govNumber)
    {
        this.ID = ID;

```

```

        this.FIO = FIO;
        this.capacity = capacity;
        this.volume = volume;
        this.from = from;
        this.to = to;
        this.distance = distance;
        this.money = money;
        this.moneyPlus = moneyPlus;
        this.name = name;
        this.firstName = firstName;
        this.lastName = lastName;
        this.active = active;
        this.govNumber = govNumber;
    }

    public int getID(){return ID;}

    public String getFIO(){return FIO;}

    public int getCapacity(){return capacity;}

    public int getVolume(){return volume;}

    public String getFrom(){return from;}

    public String getTo(){return to;}

    public int getDistance(){return distance;}

    public double getMoney(){return money;}

    public double getMoneyPlus(){return moneyPlus;}

    public String getName(){return name;}

    public String getFirstName(){return firstName;}

    public String getLastName(){return lastName;}

    public boolean getActive(){return active;}

    public String getGovNumber(){return govNumber;}
}

package ATP_forms;

import java.awt.EventQueue;
import java.awt.Font;
import java.util.List;

import javax.swing.JFrame;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;

import ATP.Transport;
import ATP.Voyage;
import Connection.CacheDB;

import java.awt.BorderLayout;
import javax.swing.GroupLayout;
import javax.swing.GroupLayout.Alignment;
import java.awt.Window.Type;

public class About {

    private JFrame About;
    //private JTable Jtable;

```

```

private List<Transport> list;
CachedDB db;
private DefaultTableModel tableModel;
private JTable table;

/**
 * Launch the application.
 */
public void newScreen() {
    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                About window = new About();
                window.About.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

public About() {
    initialize();
}

private void initialize() {
    About = new JFrame();
    About.setTitle("\u0414\u043E\u0441\u0442\u0443\u043F\u043D\u044B\u0435\u0432\u0442\u043E\u043C\u043E\u0431\u0438\u0438");
    About.setBounds(100, 100, 830, 190);

    try {
        db = new CachedDB();
    } catch (Exception e1) {
        // TODO Auto-generated catch block
        e1.printStackTrace();
    }

    table = new JTable();
    GroupLayout groupLayout = new GroupLayout(About.getContentPane());
    groupLayout.setHorizontalGroup(
        groupLayout.createParallelGroup(Alignment.LEADING)
            .addComponent(table, GroupLayout.DEFAULT_SIZE, 432,
Short.MAX_VALUE)
    );
    groupLayout.setVerticalGroup(
        groupLayout.createParallelGroup(Alignment.LEADING)
            .addComponent(table, GroupLayout.DEFAULT_SIZE, 255,
Short.MAX_VALUE)
    );
    About.getContentPane().setLayout(groupLayout);

```

```

        String col[] = {"ID", "Гос.номер", "Объем", "Грузоподъемность", "Марка",
"Модель", "Свободен(да/нет)"};

        tableModel = new DefaultTableModel(col, 0);
                                                                    // The 0 argument is
number rows.

        table.setEnabled(true);
        table.setFont(new Font("Tahoma", Font.PLAIN, 15));
        table.setFillsViewportHeight(true);
        table.setModel(tableModel);

        loadDataTotable();
    }

    public void loadDataTotable()
    {
        try {
            db.getAllTransport();
            list = db.getListTransport();
            //System.out.println(list.size());
            Object []title = {"ID", "Гос.номер", "Объем", "Грузоподъемность",
"Марка", "Модель", "Свободен(да/нет)"};
            tableModel.addRow(title);

            for (int i = 0; i < list.size(); i++){
                Object[] data = {
                    list.get(i).getID(), list.get(i).getGovNumber(),
list.get(i).getVolume(), list.get(i).getCapacity(), list.get(i).getMark(),
list.get(i).getModel(), list.get(i).getisFree()

                };

                tableModel.addRow(data);
            }
            //System.out.println(list.get(0).getFIO());

        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}

package ATP_forms;

import java.awt.EventQueue;

import javax.swing.JFrame;
import javax.swing.JTabbedPane;
import java.awt.BorderLayout;
import java.awt.Component;
import java.awt.Container;

import javax.swing.JLayeredPane;
```

```

import javax.swing.JPanel;
import javax.swing.JComponent;
import javax.swing.JDesktopPane;
import javax.swing.JFormattedTextField;
import javax.swing.JLabel;
import java.awt.Font;
import java.text.DecimalFormat;
import java.text.NumberFormat;
import java.util.Locale;

import javax.swing.JTextField;

import com.intersys.classes.ResultSet;

import javax.swing.JButton;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;

import Connection.CacheDB;

public class AdminForm {

    private JFrame frame;
    private JTextField govNumber;
    private JTextField weight;
    private JTextField volume;
    private JTextField firstName;
    private JTextField name;
    private JTextField lastName;
    private JTextField experience;
    private JTextField salary;
    private CacheDB db;
    private JTextField mark;
    private JTextField model;
    private JTextField carId;
    private JTextField workerId;
    private JTextField cost;
    private JTextField consumption;
    private JTextField transId;

    /**
     * Launch the application.
     */
    public void newScreen() {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    AdminForm window = new AdminForm();
                    window.frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    /**
     * Create the application.
     */
    public AdminForm() {
        initialize();
    }

    /**
     * Initialize the contents of the frame.
     */
    private void initialize(){
        try {
            db = new CacheDB();
        } catch (Exception e1) {
            // TODO Auto-generated catch block
            e1.printStackTrace();
        }

        frame = new JFrame();
        frame.setTitle("\u0410\u0434\u043C\u0438\u043D\u0438\u0446\u0438\u044F");
    }

```



```

        db.addTransport (govNumber.getText (), weight.getText (),
volume.getText (), mark.getText (), model.getText (), cost.getText (), consumption.getText ());
    }
});
addTransport.setFont (new Font ("Tahoma", Font.PLAIN, 15));
frame.getContentPane ().add (addTransport);

JLabel label_5 = new JLabel ("\u0414\u043E\u0431\u0430\u0432\u0438\u0442\u044C
\u0440\u0430\u0431\u043E\u043E\u043D\u0438\u043A\u0430");
label_5.setBounds (12, 287, 161, 16);
label_5.setFont (new Font ("Tahoma", Font.PLAIN, 15));
frame.getContentPane ().add (label_5);

JLabel label_6 = new JLabel ("\u0424\u0430\u043C\u0438\u043B\u0438\u0440");
label_6.setBounds (12, 316, 103, 16);
label_6.setFont (new Font ("Tahoma", Font.PLAIN, 15));
frame.getContentPane ().add (label_6);

JLabel label_7 = new JLabel ("\u0418\u043C\u0440");
label_7.setBounds (12, 345, 103, 16);
label_7.setFont (new Font ("Tahoma", Font.PLAIN, 15));
frame.getContentPane ().add (label_7);

JLabel label_8 = new
JLabel ("\u041E\u0442\u0447\u0435\u0441\u0442\u0432\u043E");
label_8.setBounds (12, 374, 103, 16);
label_8.setFont (new Font ("Tahoma", Font.PLAIN, 15));
frame.getContentPane ().add (label_8);

JLabel label_9 = new JLabel ("\u0421\u0442\u0430\u0436");
label_9.setBounds (12, 403, 103, 16);
label_9.setFont (new Font ("Tahoma", Font.PLAIN, 15));
frame.getContentPane ().add (label_9);

JLabel label_10 = new JLabel ("\u041E\u0430\u043B\u0430\u0434");
label_10.setBounds (12, 432, 103, 16);
label_10.setFont (new Font ("Tahoma", Font.PLAIN, 15));
frame.getContentPane ().add (label_10);

firstName = new JTextField ();
firstName.setBounds (176, 310, 116, 22);
frame.getContentPane ().add (firstName);
firstName.setColumns (10);

name = new JTextField ();
name.setBounds (176, 339, 116, 22);
frame.getContentPane ().add (name);
name.setColumns (10);

lastName = new JTextField ();
lastName.setBounds (176, 368, 116, 22);
frame.getContentPane ().add (lastName);
lastName.setColumns (10);

experience = new JTextField ();
experience.setBounds (176, 397, 116, 22);
frame.getContentPane ().add (experience);
experience.setColumns (10);

salary = new JTextField ();
salary.addKeyListener (new KeyAdapter () {
    @Override
    public void keyTyped (KeyEvent event) {
        char vchar = event.getKeyChar ();
        if (!Character.isDigit (vchar))
        {
            event.consume ();
        }
    }
});
salary.setBounds (176, 426, 116, 22);
frame.getContentPane ().add (salary);
salary.setColumns (10);

JButton addWorker = new
JButton ("\u0414\u043E\u0431\u0430\u0432\u0438\u0442\u044C
\u0440\u0430\u043E\u0431\u043E\u0442\u0438\u043A\u0430");
addWorker.addActionListener (new ActionListener () {
    public void actionPerformed (ActionEvent e) {

```

```

        db.addWorker(firstName.getText(), name.getText(),
lastName.getText(), salary.getText(), experience.getText());
    }

});
addWorker.setBounds(75, 457, 192, 25);
addWorker.setFont(new Font("Tahoma", Font.PLAIN, 15));
frame.getContentPane().add(addWorker);

JButton button = new JButton("\u0412\u044B\u0445\u043E\u0434");
button.setBounds(221, 702, 97, 25);
button.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        PreEnter preEnter = new PreEnter();
        preEnter.newScreen();
        frame.hide();
    }
});
button.setFont(new Font("Tahoma", Font.PLAIN, 15));
frame.getContentPane().add(button);

JLabel label_11 = new JLabel("\u041C\u0430\u0440\u043A\u0430");
label_11.setBounds(12, 131, 76, 16);
label_11.setFont(new Font("Tahoma", Font.PLAIN, 15));
frame.getContentPane().add(label_11);

JLabel modellabel = new JLabel("\u041C\u043E\u0434\u0435\u043B\u044C");
modellabel.setBounds(12, 160, 76, 16);
modellabel.setFont(new Font("Tahoma", Font.PLAIN, 15));
frame.getContentPane().add(modellabel);

mark = new JTextField();
mark.setBounds(176, 129, 116, 22);
frame.getContentPane().add(mark);
mark.setColumns(10);

model = new JTextField();
model.setBounds(176, 158, 116, 22);
frame.getContentPane().add(model);
model.setColumns(10);

JLabel label_12 = new JLabel("\u0423\u0434\u0430\u043B\u0438\u0442\u044C
\u043C\u0430\u0448\u0438\u043D\u0443 \u0438\u0437 \u0430\u0437\u0437\u0444
\u0434\u0430\u043D\u043D\u044B\u0445");
label_12.setFont(new Font("Tahoma", Font.PLAIN, 15));
label_12.setBounds(12, 502, 238, 16);
frame.getContentPane().add(label_12);

carId = new JTextField();
carId.addKeyListener(new KeyAdapter() {
    @Override
    public void keyTyped(KeyEvent event) {
        char vchar = event.getKeyChar();
        if(!Character.isDigit(vchar))
        {
            event.consume();
        }
    }
});
carId.setBounds(12, 531, 116, 22);
frame.getContentPane().add(carId);
carId.setColumns(10);

JButton button_1 = new JButton("\u0423\u0434\u0430\u043B\u0438\u0442\u044C");
button_1.addActionListener(new ActionListener() { //delete car
    public void actionPerformed(ActionEvent e) {
        String id = carId.getText();
        System.out.println("carId = " + id);
        try {
            db.updateTransId(id);
        } catch (SQLException e1) {
            // TODO Auto-generated catch block
            e1.printStackTrace();
        }
    }
});
button_1.setBounds(140, 531, 97, 25);
frame.getContentPane().add(button_1);

```

```

JLabel label_13 = new JLabel("\u0423\u0434\u0430\u043B\u0438\u0442\u044C
\u0440\u0430\u0431\u043E\u0442\u0434\u0438\u043A\u0430 \u0438\u0437 \u0431\u0440\u0430\u0442\u043E\u0445
\u0434\u0430\u0434\u0434\u0434\u044B\u0445");
label_13.setFont(new Font("Tahoma", Font.PLAIN, 15));
label_13.setBounds(12, 566, 261, 16);
frame.getContentPane().add(label_13);

JButton button_2 = new JButton("\u0423\u0434\u0430\u043B\u0438\u0442\u044C");
button_2.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        int id = Integer.parseInt(workerId.getText());
        try {
            db.deleteWorker(id);
        } catch (SQLException e1) {
            // TODO Auto-generated catch block
            e1.printStackTrace();
        }
    }
});
button_2.setBounds(140, 595, 97, 25);
frame.getContentPane().add(button_2);

workerId = new JTextField();
workerId.addKeyListener(new KeyAdapter() {
    @Override
    public void keyTyped(KeyEvent event) {
        char vchar = event.getKeyChar();
        if (!Character.isDigit(vchar))
        {
            event.consume();
        }
    }
});
workerId.setBounds(12, 595, 116, 22);
frame.getContentPane().add(workerId);
workerId.setColumns(10);

JLabel label_14 = new
JLabel("\u0421\u0442\u0430\u0438\u043C\u0441\u0442\u0442\u044C");
label_14.setFont(new Font("Tahoma", Font.PLAIN, 15));
label_14.setBounds(12, 189, 76, 16);
frame.getContentPane().add(label_14);

JLabel label_15 = new JLabel("\u0420\u0430\u0441\u0445\u043E\u0434");
label_15.setFont(new Font("Tahoma", Font.PLAIN, 15));
label_15.setBounds(12, 218, 76, 16);
frame.getContentPane().add(label_15);

cost = new JTextField();
cost.setBounds(176, 187, 116, 22);
frame.getContentPane().add(cost);
cost.setColumns(10);

consumptioon = new JTextField();
consumptioon.setBounds(176, 216, 116, 22);
frame.getContentPane().add(consumptioon);
consumptioon.setColumns(10);

JLabel label_16 = new JLabel("$");
label_16.setFont(new Font("Tahoma", Font.PLAIN, 15));
label_16.setBounds(303, 189, 15, 16);
frame.getContentPane().add(label_16);

JLabel label_17 = new JLabel("\u0431/100\u043A\u043C.");
label_17.setFont(new Font("Tahoma", Font.PLAIN, 15));
label_17.setBounds(297, 222, 76, 16);
frame.getContentPane().add(label_17);

JLabel label_18 = new JLabel("\u0434\u0440\u0434/\u043A\u043C");
label_18.setFont(new Font("Tahoma", Font.PLAIN, 15));
label_18.setBounds(297, 432, 103, 16);
frame.getContentPane().add(label_18);

transId = new JTextField();
transId.setBounds(12, 659, 116, 22);
frame.getContentPane().add(transId);
transId.setColumns(10);

```

```

        JLabel lblNewLabel_1 = new
JLabel("\u041E\u0441\u0432\u043E\u0431\u043E\u0438\u0442\u044C
\u0430\u0432\u0442\u043E\u043C\u043E\u0431\u0438\u043B\u044C");
        lblNewLabel_1.setFont(new Font("Tahoma", Font.PLAIN, 15));
        lblNewLabel_1.setBounds(12, 630, 181, 16);
        frame.getContentPane().add(lblNewLabel_1);

        JButton setIdTrans = new
JButton("\u0418\u0441\u043F\u043E\u043B\u043D\u0438\u0442\u044C");
        setIdTrans.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent arg0) {
                //int id = Integer.parseInt(setIdTrans.getText());
                try {
                    db.updateTransId(transId.getText());
                } catch (SQLException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }
        });
        setIdTrans.setBounds(140, 658, 97, 25);
        frame.getContentPane().add(setIdTrans);

    }

}

package ATP_forms;

import java.awt.EventQueue;

import javax.swing.JFrame;
import javax.swing.JButton;
import java.awt.BorderLayout;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.awt.Font;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JTextField;

import Connection.CacheDB;

import java.awt.event.InputMethodListener;
import java.awt.event.InputMethodEvent;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
import java.sql.SQLException;

public class ClientForm {

    private JFrame frame;
    private JTextField fio;
    private JTextField weight;
    private JTextField volume;
    private JTextField from;
    private JTextField to;
    private JTextField distance;
    private JButton calc;
    private boolean bool1, bool2, bool3;
    private CacheDB db;

    /**
     * Launch the application.
     */
    public void newScreen() {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    ClientForm window = new ClientForm();
                    window.frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

}

/**

```



```

        public void keyTyped(KeyEvent event) {
            char vchar = event.getKeyChar();
            if(!Character.isDigit(vchar))
            {
                event.consume();
            }
        }
    });
    weight.addKeyListener(new KeyAdapter() {
        @Override
        public void keyTyped(KeyEvent e) {
            bool1 = true;
            if(bool1 && bool2 && bool3){
                calc.setEnabled(true);
            }
        }
    });
    weight.setBounds(211, 36, 116, 22);
    frame.getContentPane().add(weight);
    weight.setColumns(10);

    volume = new JTextField();
    volume.addKeyListener(new KeyAdapter() {
        @Override
        public void keyTyped(KeyEvent event) {
            char vchar = event.getKeyChar();
            if(!Character.isDigit(vchar))
            {
                event.consume();
            }
        }
    });
    volume.addKeyListener(new KeyAdapter() {
        @Override
        public void keyTyped(KeyEvent e) {
            bool2 = true;
            if(bool1 && bool2 && bool3){
                calc.setEnabled(true);
            }
        }
    });
    volume.setBounds(211, 65, 116, 22);
    frame.getContentPane().add(volume);
    volume.setColumns(10);

    from = new JTextField();
    from.setBounds(211, 94, 116, 22);
    frame.getContentPane().add(from);
    from.setColumns(10);

    to = new JTextField();
    to.setBounds(211, 123, 116, 22);
    frame.getContentPane().add(to);
    to.setColumns(10);

    JButton next = new JButton("\u0414\u0430\u0432\u044c\u0448\u0435");
    next.setEnabled(false);
    next.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent arg0) {
            try {
                db.updateTransport();

                double calc = calc();
                db.setVoyage(fio.getText(), weight.getText(),
volume.getText(), from.getText(), to.getText(), distance.getText(), calc );

                ClientTransactionForm transaction = new
ClientTransactionForm();

                transaction.newScreen();
                frame.hide();
            } catch (SQLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    });
    next.setFont(new Font("Tahoma", Font.PLAIN, 15));

```

```

        next.setBounds(230, 187, 97, 25);
        frame.getContentPane().add(next);

        calc = new
JButton("\u0420\u0430\u0441\u0441\u0447\u0438\u0442\u0430\u0442\u044c\u0440\u0430\u0442\u044c");
        calc.setEnabled(false);
        calc.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) { //calc
                int capacity = Integer.parseInt(weight.getText());
                int volumeInt = Integer.parseInt(volume.getText());
                try {
                    db.searchFreeTransport(capacity, volumeInt);
                    if(db.getListSize() >= 1)
                    {
                        JOptionPane.showMessageDialog(null, "Для
вашего заказа выбран автомобиль " + db.getMark() + " " + db.getModel() + "\nСтоимость составит: "
+ calc() + "грн.", null, JOptionPane.INFORMATION_MESSAGE);
                        next.setEnabled(true);
                    }
                    if(db.getListSize() < 1)
                        JOptionPane.showMessageDialog(null, "На
данный момент на базе нету подходящего вида транспорта чтобы принять ваш заказ.", null,
JOptionPane.INFORMATION_MESSAGE);
                    calc.setEnabled(false);
                } catch (Exception e1) {
                    // TODO Auto-generated catch block
                    e1.printStackTrace();
                }
            }
        });
        calc.setFont(new Font("Tahoma", Font.PLAIN, 15));
        calc.setBounds(116, 187, 112, 25);
        frame.getContentPane().add(calc);

        distance = new JTextField();
        distance.addKeyListener(new KeyAdapter() {
            @Override
            public void keyTyped(KeyEvent event) {
                char vchar = event.getKeyChar();
                if(!Character.isDigit(vchar))
                {
                    event.consume();
                }
            }
        });
        distance.addKeyListener(new KeyAdapter() {
            @Override
            public void keyTyped(KeyEvent arg0) {
                bool3 = true;
                if(bool1 && bool2 && bool3){
                    calc.setEnabled(true);
                }
            }
        });

        distance.setBounds(211, 152, 116, 22);
        frame.getContentPane().add(distance);
        distance.setColumns(10);

    }
    public double calc()
    {
        int distanceInt = Integer.parseInt(distance.getText());
        int weightInt = Integer.parseInt(weight.getText());
        double money = distanceInt*5 + 0.01*db.getCost()*weightInt/10 +
distanceInt*(db.getConsumption()/100)*20;
        return money;
    }
}

package ATP_forms;

import java.awt.EventQueue;

import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;

import java.awt.Font;

```

```

import javax.swing.JTextField;

import Connection.CachedDB;

import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
import java.sql.SQLException;
import java.awt.event.ActionEvent;

public class ClientTransactionForm {

    private JFrame frame;
    private JTextField textField;
    private JTextField textField_1;

    /**
     * Launch the application.
     */
    public void newScreen() {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    ClientTransactionForm window = new
ClientTransactionForm();
                    window.frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    public ClientTransactionForm() {
        initialize();
    }

    private void initialize() {
        frame = new JFrame();
        frame.setTitle("\u041E\u043F\u043B\u0430\u0442\u0430");
        frame.setBounds(100, 100, 296, 152);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.getContentPane().setLayout(null);

        JLabel lblNewLabel = new JLabel("\u0414\u0435\u043C\u0435\u0440\u0430\u0442\u0430\u0440\u0442\u044B");
        lblNewLabel.setFont(new Font("Tahoma", Font.PLAIN, 15));
        lblNewLabel.setBounds(12, 13, 105, 16);
        frame.getContentPane().add(lblNewLabel);

        JLabel lblCvv = new JLabel("CVV2");
        lblCvv.setFont(new Font("Tahoma", Font.PLAIN, 15));
        lblCvv.setBounds(12, 40, 105, 16);
        frame.getContentPane().add(lblCvv);

        textField = new JTextField();
        textField.addKeyListener(new KeyAdapter() {
            @Override
            public void keyTyped(KeyEvent event) {
                char vchar = event.getKeyChar();
                if(!Character.isDigit(vchar))
                {
                    event.consume();
                }
            }
        });
        textField.setFont(new Font("Tahoma", Font.PLAIN, 15));
        textField.setBounds(129, 11, 116, 22);
        frame.getContentPane().add(textField);
        textField.setColumns(10);

        textField_1 = new JTextField();
        textField_1.addKeyListener(new KeyAdapter() {
            @Override
            public void keyTyped(KeyEvent event) {
                char vchar = event.getKeyChar();
                if(!Character.isDigit(vchar))
                {
                    event.consume();
                }
            }
        });
    }
}

```

```

    }
    });
    textField_1.setFont(new Font("Tahoma", Font.PLAIN, 15));
    textField_1.setColumns(10);
    textField_1.setBounds(129, 38, 116, 22);
    frame.getContentPane().add(textField_1);

    JButton btnNewButton = new
JButton("\u041E\u043F\u043B\u0430\u0442\u0438\u0442\u044C");
    btnNewButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            if(textField.getText().equals("")) ||
textField_1.getText().equals(""))
                JOptionPane.showMessageDialog(null, "Заполните все поля",
null, JOptionPane.INFORMATION_MESSAGE);
            else{
                PreEnter enter = new PreEnter();
                enter.newScreen();
                frame.hide();
            }
        }
    });
    btnNewButton.setFont(new Font("Tahoma", Font.PLAIN, 15));
    btnNewButton.setBounds(83, 69, 116, 25);
    frame.getContentPane().add(btnNewButton);
}

}

package ATP_forms;

import java.awt.EventQueue;
import java.util.List;

import javax.swing.JFrame;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;
import javax.swing.table.TableColumn;

import ATP.*;
import Connection.CacheDB;
import javax.swing.JButton;
import javax.swing.GroupLayout;
import javax.swing.GroupLayout.Alignment;
import java.awt.Font;
import javax.swing.JLabel;
import javax.swing.JTextField;
import javax.swing.LayoutStyle.ComponentPlacement;
import java.awt.event.ActionListener;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
import java.sql.SQLException;
import java.awt.event.ActionEvent;

public class DispatcherForm {

    private JFrame frame;
    private JTable table;
    private List<Voyage> list;
    CacheDB db;
    private DefaultTableModel tableModel;
    private JLabel lblNewLabel;
    private JTextField endVoyage;
    /**
     * Launch the application.
     */
    public void newScreen() {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    DispatcherForm window = new
DispatcherForm();

```

```

        window.frame.setVisible(true);
    } catch (Exception e) {
        e.printStackTrace();
    }
}

});

}

/**
 * Create the application.
 */
public DispatcherForm() {
    initialize();
}

/**
 * Initialize the contents of the frame.
 */
private void initialize() {
    try {
        db = new CachedDB();
    } catch (Exception e1) {
        // TODO Auto-generated catch block
        e1.printStackTrace();
    }
    frame = new JFrame();

    frame.setTitle("\u0414\u0438\u0441\u043F\u0435\u0442\u0447\u0435\u0440");
    frame.setBounds(100, 100, 1509, 389);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    String col[] = {"FIO", "Capacity", "Volume", "From", "To", "Distance",
"Money", "Заработок", "Имя водителя", "Фамилия водителя", "Отчество водителя", "В
рейсе (да/нет)", "Гос. номер"};

    tableModel = new DefaultTableModel(col, 0);
    // The 0 argument is
number rows.

    table = new JTable();
    table.setEnabled(false);
    table.setFont(new Font("Tahoma", Font.PLAIN, 15));
    table.setFillViewportHeight(true);
    table.setModel(tableModel);
    //table.getColumnModel().getColumn(0).setPreferredWidth(70);

    JButton end = new
JButton("\u0437\u0430\u0432\u0440\u0440\u0448\u0438\u0442\u044c");
    end.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent arg0) {
            int id = Integer.parseInt(endVoyage.getText());
            try {
                db.changeVoyage(id);
            } catch (SQLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    });
    end.setFont(new Font("Tahoma", Font.PLAIN, 15));

    lblNewLabel = new JLabel("\u0420\u0435\u0439\u0441\u0441\u0443\u0447\u0435\u043d\u0440\u0435\u0435\u0440");
    lblNewLabel.setFont(new Font("Tahoma", Font.PLAIN, 15));

    endVoyage = new JTextField();
    endVoyage.addKeyListener(new KeyAdapter() {
        @Override
        public void keyTyped(KeyEvent event) {
            char vchar = event.getKeyChar();
            if(!Character.isDigit(vchar))
            {

```

```

        event.consume();
    }
}

});
endVoyage.setFont(new Font("Tahoma", Font.PLAIN, 15));
endVoyage.setColumns(10);

JButton exit = new JButton("\u0412\u044B\u0445\u043E\u0434\u0438");
exit.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        PreEnter preEnter = new PreEnter();
        preEnter.newScreen();
        frame.hide();
    }
});
exit.setFont(new Font("Tahoma", Font.PLAIN, 15));
GroupLayout groupLayout = new GroupLayout(frame.getContentPane());
groupLayout.setHorizontalGroup(
    groupLayout.createParallelGroup(Alignment.LEADING)
        .addGroup(groupLayout.createSequentialGroup()

.addGroup(groupLayout.createParallelGroup(Alignment.LEADING)

.addGroup(groupLayout.createSequentialGroup()

.addContainerGap()
.addComponent(lblNewLabel)
.addGap(18)
.addComponent(endVoyage,
GroupLayout.PREFERRED_SIZE, 78, GroupLayout.PREFERRED_SIZE)

.addPreferredGap(ComponentPlacement.RELATED)

.addComponent(end)
.addGap(18)
.addComponent(exit))

.addComponent(table,
GroupLayout.DEFAULT_SIZE, 1479, Short.MAX_VALUE)

.addContainerGap()
);
groupLayout.setVerticalGroup(
    groupLayout.createParallelGroup(Alignment.LEADING)
        .addGroup(groupLayout.createSequentialGroup()
            .addComponent(table,
GroupLayout.PREFERRED_SIZE, 262, GroupLayout.PREFERRED_SIZE)
            .addGap(4)

.addGroup(groupLayout.createParallelGroup(Alignment.BASELINE)
            .addComponent(lblNewLabel)
            .addComponent(endVoyage,
GroupLayout.PREFERRED_SIZE, GroupLayout.DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)
            .addComponent(end)
            .addComponent(exit))
        );
frame.getContentPane().setLayout(groupLayout);

loadDataTotable();
table.setAutoResizeMode(JTable.AUTO_RESIZE_ALL_COLUMNS);
}

public void loadDataTotable()
{
    try {
        db.GetActiveVoyage();
        list = db.getList();
        //System.out.println(list.size());
        Object []title = {"Номер рейса", "ФИО", "Вес
груза (т)", "Объем (м3)", "Откуда", "Куда", "Расстояние", "Плата", "Заработок", "Имя
водителя", "Фамилия водителя", "Отчество водителя", "В рейсе (да/нет)", "Гос. номер"};
        tableModel.addRow(title);

        for (int i = 0; i < list.size(); i++){

```

```

        Object[] data = {
            list.get(i).getID(), list.get(i).getFIO(),
list.get(i).getCapacity(), list.get(i).getVolume(), list.get(i).getFrom(),
list.get(i).getTo(), list.get(i).getDistance(), list.get(i).getMoney(),
list.get(i).getMoneyPlus(), list.get(i).getName(),list.get(i).getFirstName(),
list.get(i).getLastName(), list.get(i).getActive(), list.get(i).getGovNumber()

        };

        tableModel.addRow(data);
    }
    //System.out.println(list.get(0).getFIO());
} catch (Exception e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}
}
}

```

ДОДАТОК Д
Слайди презентації

Національний університет «Запорізька політехніка»

Кафедра програмних засобів

Дипломна кваліфікаційна робота

Програмне забезпечення обліку земельних ділянок

Виконав

ст. гр. КНТз-219_2

А. С. Пащенко

Керівник

д.т.н., професор

А. О. Олійник

Рисунок Д.1 – Слайд 1

Об'єктом дослідження є процес розробки програмного забезпечення для обліку земельних ділянок.

Предмет дослідження – програмні засоби обліку земельних ділянок.

Мета роботи – програмна реалізація процесу обліку земельних ділянок.

Рисунок Д.2 – Слайд 2

Задачі роботи:

- виконати аналіз предметної області відносно процесу обліку земельних ділянок;
- виконати проєктування програмного забезпечення для обліку земельних ділянок;
- створити програмне забезпечення для обліку земельних ділянок;
- виконати тестування розробленого програмного забезпечення обліку земельних ділянок.

3

Рисунок Д.3 – Слайд 3

Програмні засоби для обліку земельних ділянок

Критерії порівняння	Програмні засоби				
	Peleton LandView	P2 iLandman	Landdcox	MyQuorum Land	LandPro 2000
Карти	cloud-based	Arc-based	API	SQL	Map Suite GIS
Звітність	Query файли	Excel & PDF Формат звітів	SQL	SQL	Зручна звітність у додатку
Ліцензування	Немає обмежень	Підписка + безкоштовна версія	Продаж на основі вартості	Ліцензійна плата/кільк співробіт.	Немає обмежень
Платіжні зобов'язання	Фільтри для платіжної системи	Звіти в Excel	Неавтомат. Звіти для бухгалтерії	Quorum звіти	Звіти в Excel
Розрахунки порядку поділу	облік доходів	Геодезичні розрахунки	Розрахунки об'ємів праці та оплати	Розрахунки в порядку поділу об'ємів роботи	Розрахунки в порядку поділу
Пакет розрахунків за спільні інтереси	LJB, GWI	Можна відмінити роялті	фіксує колоди витрат і доходів на хорошому рівні	Фіксує	Можна керувати холдингами
Платформа	Хмарне сховище, SaaS, Web-Based	Хмарне сховище, SaaS, Web-Based	Хмарне сховище, SaaS, Web-Based	Хмарне сховище, SaaS, Web-Based	Windows

4

Рисунок Д.4 – Слайд 4

Вибір мови програмування

Критерій порівняння мов програмування	Мова програмування		
	PHP	JavaScript	C#
Розташування скриптів на стороні сервера	+	–	+
Чутливість до регістру змінних	+–	+	+
Можливість роботи з асоціативними масивами	+	–	+
Обробка критичних ситуацій	+	+–	+
Взаємодія з базами даних	+	+–	+
Підтримка багатопоточності	+	–	+
Зручність для створення застосунку обліку земельних ділянок	+	+–	+–

5

Рисунок Д.5 – Слайд 5

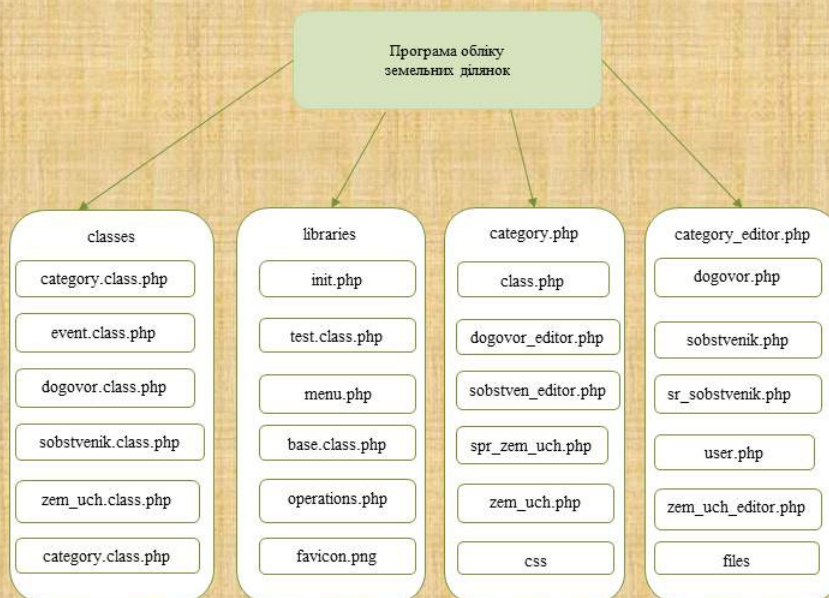
Порівняльний аналіз систем керування базами даних

Критерії порівняння	Системи управління базами даних		
	MySQL	MS Access	MongoDB
Зручність для створення ПЗ обліку земельних ділянок	+	+–	+–
Типи даних, що підтримуються	структуровані, неструктуровані	структуровані, неструктуровані	неструктуровані
Наявність документації у вільному доступі	+	+–	+–
Ліцензія	GNU	приватна	SSPL
Зручність для обробки великих даних	+	+–	+–

6

Рисунок Д.6 – Слайд 6

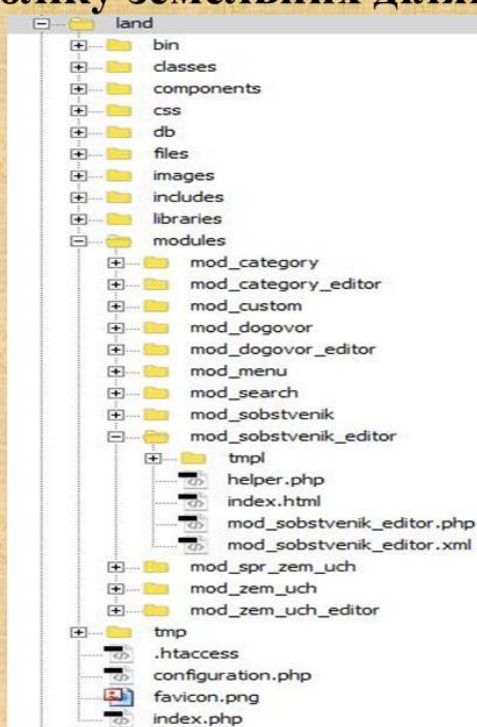
Структура модулів програмного забезпечення обліку земельних ділянок



7

Рисунок Д.7 – Слайд 7

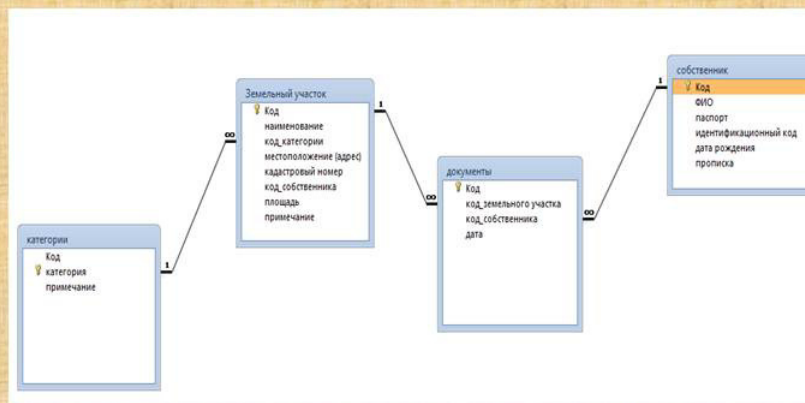
Фізична структура програмного забезпечення обліку земельних ділянок



8

Рисунок Д.8 – Слайд 8

Схема бази даних програмного забезпечення обліку земельних ділянок



9

Рисунок Д.9 – Слайд 9

Приклад роботи з програмою

Собственники Земельные участки Договора Категории

СОБСТВЕННИКИ

Поиск сведений об указанном владельце:

ФИО: ИСКАТЬ

№	ФИО	Дата рождения	Идентификационный номер	Паспорт	Прописка	Операция
1	Зинченко Анна Владимировна	1990-20-06	33366669965	О002065289	г.Запорожье ул.40 лет Сов. Укр 64, кв.5	удалить добавить
2	Вольниц Ольга Валерьевна	1989-20-06	2316369565	С002165385	г.Запорожье ул.Мира 10, кв.5	удалить добавить
3	Иващенко Рита Владимировна	1965-15-01	2326666285	СЮ4015288	г.Мелитополь, б-р. 30 лет Победы, 1/72	удалить добавить
4	Сенцов Олег Владимировна	1990-20-06	33366669965	О002065289	г.Бердянск, ул. Шевченко, 12, кв. 4	удалить добавить
5	Зинченко Анна Владимировна	1990-20-06	33366669965	О002065289	г.Мелитополь, пр-т Б.Хмельницкого, 24, кв. 432	удалить добавить

[+ ДОБАВИТЬ](#)

1 2 3 4

[Экспорт в Excel](#)

10

Рисунок Д.10 – Слайд 10

Приклад роботи з програмою

СОБСТВЕННИКИ

ФІО:

Дата народження:

Паспорт:

Прописка:

Ідентифікаційний номер:

11

Рисунок Д.11 – Слайд 11

Приклад роботи з програмою

ДОГОВОРА

Земельний ділянка:

Власник:

Дата народження:

№	Номер договору	Земельний ділянка	Місцезнаходження ділянки	Кадастровий номер ділянки	Площа
1	№ 2-93	Землі для с/г призначення	Україна, г.Бердянськ, ул. Шевченко 15	2620455100:01:001:0261	0.103 га
2	№ 3-11	Землі поселення	Україна, г.Бердянськ, ул. Шевченко 12	2620455100:01:001:0246	0.15 га
3	№ 2-15	Землі для введення товарного с/г виробництва	Україна, г.Бердянськ, ул. Шевченко 16	2620455100:01:001:0258	0.123 га
4	№ 6-21	Землі поселення	Україна, г.Мелітополь, ул.Залізнична, 85	2620455100:01:004:0738	удалить добавить
5	№ 9-34	Землі поселення	Україна, г.Мелітополь, ул.Гоголя, 16	2620455100:01:001:1014	удалить добавить

12

Рисунок Д.12 – Слайд 12

Висновки

В ході виконання даної дипломної кваліфікаційної роботи бакалавра було проаналізовано та досліджено процес розробки програмного забезпечення для обліку земельних ділянок.

В процесі розробки вирішено такі завдання:

- аналіз предметної області відносно процесу обліку земельних ділянок;
- проектування програмного забезпечення для обліку земельних ділянок;
- розробка програмного забезпечення для обліку земельних ділянок;
- тестування розробленого програмного забезпечення обліку земельних ділянок.

13

Рисунок Д.13 – Слайд 13

Висновки

Для створення програмного забезпечення обліку земельних ділянок було обрано мову програмування Php, яка є широко використовувана мова сценаріїв із відкритим кодом загального призначення, ключовою перевагою PHP є можливість створювати документи HTML за допомогою команд PHP. Для створення програми обліку земельних ділянок як система керування базами даних обрана MySQL.

Розроблено загальну структуру програми обліку земельних ділянок, яка відображає сукупність основних сторінок програмного забезпечення, поданих у вигляді php-модулів та об'єднаних єдиним дизайном, стилем і посиланнями.

Проведені експерименти засвідчили, що програма має достатню працездатність і може використовуватися за призначенням для обліку земельних ділянок.

14

Рисунок Д.14 – Слайд 14