

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ**

Інститут (факультет) _____ **автоматизації і комп'ютерних систем** _____
Кафедра _____ **інформаційних систем** _____

«До захисту в ЕК»
Директор інституту(декан факультету)
_____ Форсюк А. В. _____
(підпис) (прізвище та ініціали)

« ____ » _____ 20 ____ р.

«До захисту допущено»
Завідувач кафедри
_____ Чумаченко С.М. _____
(підпис) (прізвище та ініціали)

« ____ » _____ 20 ____ р.

**КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВРА**

зі спеціальності _____ 122 " Комп'ютерні науки " _____
(код та назва спеціальності)

освітньо-професійної програми _____ 122 " Комп'ютерні науки " _____
на тему: Створення Web-орієнтованої інформаційної системи лабораторії
біобезпеки, якості харчової продукції та безпеки харчування НУХТ

Виконав: здобувач _____ 4 _____ курсу, групи _____ 4 _____

_____ Медуха Євгеній Сергійович _____
(прізвище, ім'я, по батькові повністю) (підпис)

Керівник _____ Грибков Сергій Віталійович _____
(прізвище та ініціали) (підпис)

Консультанти _____ Грибков Сергій Віталійович _____
(прізвище та ініціали) (підпис)

_____ Грибков Сергій Віталійович _____
(прізвище та ініціали) (підпис)

_____ Грибков Сергій Віталійович _____
(прізвище та ініціали) (підпис)

Рецензент _____ Власенко Лідія Олександрівна _____
(прізвище та ініціали) (підпис)

Засвідчую, що в цій кваліфікаційній
роботі немає запозичень із праць
інших авторів без відповідних
посилань.

Здобувач _____
(підпис)

Київ – 2020р.

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) _____ автоматизації і комп'ютерних систем _____

Кафедра _____ інформаційних систем _____

Освітній ступінь _____ бакалавр _____

Спеціальність _____ 122 «Комп'ютерні науки» _____

(код і назва)

Освітньо-професійна програма _____ Комп'ютерні науки _____

(назва)

ЗАТВЕРДЖУЮ

Завідувач

кафедри _____

“ _____ ” _____ 20 _____ року

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА

Медуха Євгеній Сергійович

(прізвище, ім'я, по батькові)

1. Тема роботи _____ Створення Web-орієнтованої інформаційної системи лабораторії біобезпеки, якості харчової продукції та безпеки харчування НУХТ _____

керівник роботи _____ Грибков Сергій Віталійович, доц, к.т.н _____,

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “27” квітня 2020 року №269-кв

2. Строк подання здобувачем роботи _____ 29.05.2020 року

3. Вихідні дані до роботи _____ посадові інструкції, інформація про угоди, замовлення та технічний засоби лабораторії, зразки документів _____

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити) _____ системний аналіз діяльності підприємства, функціональний аналіз роботи лабораторії, постановка задачі, створення моделей даних, бази даних, створення ІС, інструкція користувача, заходи з охорони праці, висновки _____

5. Перелік графічного матеріалу _____

_____ функціональна модель «AS-IS» процесу виконання замовлень, функціональна модель «TO-BE» процесу виконання замовлень, моделі бази даних, структура бази даних, інтерфейс користувача _____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Вступ	Грибков С.В., доцент		
1	Грибков С.В., доцент		
2	Грибков С.В., доцент		
3	Грибков С.В., доцент		

7. Дата видачі завдання 27 квітня 2020 року

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів виконання кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Формування вступу	29.04.2020	виконано
2	Системний аналіз діяльності підприємства	01.05.2020	виконано
3	Функціональне моделювання	04.05.2020	виконано
4	Пошук та аналіз ІС	07.05.2020	виконано
5	Розробка логічної та фізичної моделей даних	08.05.2020	виконано
6	Генерація бази даних в СУБД	10.05.2020	виконано
7	Розробка елементів системи	12.05.2020	виконано
8	Створення інструкції користувача	14.05.2020	виконано
9	Розробка розділу з охорони праці	23.05.2020	виконано
10	Завершення роботи над пояснювальною запискою	26.05.2020	виконано
11	Створення презентації	28.05.2020	виконано

Здобувач

(підпис)

Медуха Є.С.

(прізвище та ініціали)

Керівник роботи

(підпис)

Грибков С.В.

(прізвище та ініціали)

АНОТАЦІЯ

Кваліфікаційна робота «Створення Web-орієнтованої інформаційної системи лабораторії біобезпеки, якості харчової продукції та безпеки харчування НУХТ» складається з 87 сторінок, 5 таблиць, 29 рисунків, 4 додатків, 20 сторінок графічного матеріалу, 20 літературних джерел.

У кваліфікаційній роботі наведені результати дослідження діяльності об'єкта автоматизації лабораторії біобезпеки, якості харчової продукції та безпеки харчування НУХТ – відділу кадрів. Обґрунтована необхідність розробки інформаційної системи для відділу.

Дослідження бізнес-процесів підприємства покладені в основу функціональної моделі, яка документує діяльність Центру. Функціональна модель розроблена у CASE-засобі CA ERwin Process Modeler 7.3 на основі використання стандарту IDEF0. На її основі створені логічна та фізична моделі у CASE-засобі CA ERwin Data Modeler 7.3 і згенерована база даних в MySQL 5.7.

Графічний інтерфейс користувача реалізований мовою об'єктно-орієнтованого програмування PHP, середовище розробки PHPStorm.

Розрахована економічна ефективність впровадження інформаційної системи на підприємстві, описані вимоги з охорони праці.

КЛЮЧОВІ СЛОВА: ІНФОРМАЦІЙНА СИСТЕМА, ЛАБОРАТОРІЯ, ФУНКЦІОНАЛЬНА МОДЕЛЬ, ФІЗИЧНА МОДЕЛЬ, БАЗА ДАНИХ, ГРАФІЧНИЙ ІНТЕРФЕЙС, ER-МОДЕЛЬ.

ANNOTATION

Qualification work "Creation of Web-oriented information system of the laboratory of biosafety, food quality and food safety NUHT" consists of 87 pages, 5 tables, 29 figures, 4 appendices, 20 pages of graphic material, 20 literature sources.

The qualification work presents the results of the study of the activity of the object of automation of the laboratory of biosafety, food quality and food safety of NUHT - personnel department. The necessity of developing an information system for the department is substantiated.

Research of business processes of the enterprise is the basis of the functional model, which documents the activities of the Center. The functional model was developed in the CASE tool CA ERwin Process Modeler 7.3 based on the use of the IDEF0 standard. Based on it, logical and physical models were created in CASE-tool CA ERwin Data Modeler 7.3 and a database was generated in MySQL 5.7.

The graphical user interface is implemented in the object-oriented programming language PHP, PHPStorm development environment.

The economic efficiency of the information system implementation at the enterprise is calculated, the labor protection requirements are described.

KEY WORDS: INFORMATION SYSTEM, LABORATORY, FUNCTIONAL MODEL, PHYSICAL MODEL, DATABASE, GRAPHIC INTERFACE, ER-MODEL.

ЗМІСТ

ВСТУП.....	7
РОЗДІЛ 1 СИСТЕМНИЙ АНАЛІЗ ОБ'ЄКТУ АВТОМАТИЗАЦІЇ ТА ПОСТАНОВКА ЗАДАЧІ НА ПРОЕКТУВАННЯ.....	10
1.1 Загальна характеристика	10
1.2 Дослідження основних функцій роботи об'єкта автоматизації	14
1.3 Аналіз існуючого на підприємстві програмного забезпечення.....	19
1.4 Огляд існуючих рішень	20
1.5. Обґрунтування доцільності проектування і розробки інформаційної системи	21
1.5.1. Мета та цілі розвитку Web-орієнтованої інформаційної системи лабораторії біобезпеки.....	21
1.5.2 Розробка вимог до функціональних характеристик і до якості програмного продукту	22
1.6 Постановка задачі на розробку інформаційної системи	23
Висновки до розділу 1	24
РОЗДІЛ 2 РОЗРОБКА КОМПЛЕКСУ ЗАДАЧ АВТОМАТИЗАЦІЇ.....	26
2.1 Обґрунтування вибору програмного забезпечення та технології.....	26
2.2 Методи вирішення задачі	30
2.2.1 Проектування бази даних	30
2.2.2 Створення інтерфейсу та реалізація функцій системи.....	44
2.3 Інструкція користувача.....	53
2.3.1 Технологічна послідовність роботи з системою.....	53
2.3.2 Виконання функцій адміністрації бази даних.....	58
2.4 Розрахунок ефективності витрат	61
Висновки до розділу 2	71
РОЗДІЛ 3 ОХОРОНА ПРАЦІ ТА ТЕХНІКА БЕЗПЕКИ	75
Висновки до розділу 3	84
ВИСНОВОК	85
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	86
ДОДАТКИ.....	88

ВСТУП

Метою будь-якої організації, що займається наданням товарів або послуг, є максимізація та поглиблення зв'язків між продавцями і покупцями товарів і послуг, забезпечуючи ці посилення в найкоротші терміни. Таким чином, організація забезпечує зв'язки між зацікавленими сторонами ринку.

Цілями організації є досягнення найвищого успіху в постачанні товарів і послуг, забезпечення сприятливих умов для клієнтів, шукати нові можливості для дистрибуції та розширити спектр діяльності.

Надання послуг через Інтернет завжди спрямоване на споживачів і спрямована на те, щоб повідомляти їх по-різному про нові продукти або послуги та їх властивості. Великі активи, інтелектуальний потенціал і нові технології, в тому числі сучасні інформаційні технології, інвестуються в рекламу і просування товарів і послуг.

Об'єктом дослідження є центр колективного користування науковим обладнанням «Лабораторія біобезпеки, якості харчової продукції та безпеки харчування», яке займається послугами у сфері надання послуг з дослідження товарів харчування різного виду.

Метою цієї роботи є запропонувати шляхи підвищення ефективності діяльності «Лабораторія біобезпеки, якості харчової продукції та безпеки харчування» за рахунок нових інформаційних технологій та Інтернету.

На підставі аналізу цього закладу були розроблені висновки і розроблено універсальне програмне забезпечення та модель бази даних. Розглянуто створення та розвиток програмного забезпечення інформаційних систем у «Лабораторія біобезпеки, якості харчової продукції та безпеки харчування», що базується на сучасних програмних платформах, сучасних концепціях в області систем управління базами даних (БД), сучасних концепціях прикладного програмування, інтернет-технологій. Це використання поєднання цих технологій, що дозволяє отримати доступ до бази даних і компонентів програмного забезпечення не тільки для

співробітників закладу, але і для всіх зацікавлених сторін у цій сфері діяльності, кожна зі своєю власною, певною інформацією.

Інформаційні системи є компонентом, який забезпечує швидке збереження та управління інформацією, оперативне пошуку інформації, накопичення статистичних даних та аналітичних досліджень.

Одним з можливих концепцій створення інформаційних систем «Лабораторія біобезпеки, якості харчової продукції та безпеки харчування» вважається клієнт-серверна архітектура, а в якості основної платформи для реалізації програмного забезпечення такої архітектури запропонована платформа сімейства Linux(UNIX) і т. д. з цієї серії. Насправді, мова йде про веб-орієнтовану ІС з окремим доступом до бази даних і системних функцій для різних груп зовнішніх користувачів і співробітників лабораторії.

Це неможливо в одному дипломному проєкті, щоб покрити всі аспекти будівництва і. п., тому давайте зосередимо увагу на невеликий, але важливий фрагмент, такий як база даних, яка є частиною структури установи, програмні засоби, які забезпечують інтерфейс бази даних і веб-сайт лабораторії. База даних має доступ до центру та зовнішніх споживачів послуг, але кожна з них має власну площу бази даних.

На підставі аналізу існуючої облікової документації, була розроблена структура бази даних і створено базу даних, використовуючи систему управління базами даних (СУБД MySQL).

Представлено інтуїтивно зрозумілий веб-інтерфейс для дистанційної роботи з компонентами бази даних та програмного забезпечення, демонструючи, як члени центру отримують віддалений доступ до баз даних установи. Для роботи з системою потрібен лише браузер будь-якого типу.

Використання системи забезпечує принципово нові можливості у взаємодії учасників ринку через Інтернет, архітектуру клієнтського сервера та віддалений доступ до інформації. Замовник, споживач послуг не потребує особистого контакту (або в деяких випадках, до певного, остаточного моменту) з представниками організації. З усією доступною інформацією

закладу, він може ознайомитись онлайн, вибрати необхідні варіанти сервісу (тип необхідного документу).

РОЗДІЛ 1 СИСТЕМНИЙ АНАЛІЗ ОБ'ЄКТУ АВТОМАТИЗАЦІЇ ТА ПОСТАНОВКА ЗАДАЧІ НА ПРОЕКТУВАННЯ

1.1 Загальна характеристика

Об'єктом дослідження є Центру колективного користування науковим обладнанням, яке є структурним підрозділом Національного університету харчових технологій. Метою діяльності центру є організація та проведення наукових досліджень на обладнанні, забезпечення якості та надійності його роботи, необхідне фахове обслуговування, умови найкращого використання робочого часу в інтересах наукової спільноти, а також отримання результатів досліджень та розробок на сучасному науковому рівні.

Користувачі центру (користувачами Центру є навчальні заклади, зазначенні в наказі Міністерства освіти і науки України за № 444 від 02.05.2018 року), які мають потребу в проведенні досліджень на приладах подають заявку в встановленій формі. Потім Ректор НУХТ, базової установи Центру, затверджує загальний розподіл між замовниками часу, відведеного для колективного користування науковими приладами. Далі передбачається автоматизована обробка документа в відділі.

Далі в цій статті ми говоримо про заявника як користувача системи, що розробляється, який потенційно здатний зробити заявку в Інтернеті. Насправді, це не робиться, і заявка оформлюється і відправляється іншими користувачами системи. Тому, кажучи про оформлення заявок онлайн, ми будемо мати на увазі, що це робиться іншими користувачами, персоналом, який «запускає» замовлення-заявку в процес обробки. Це слід врахувати в подальшому викладі матеріалу.

Послуга буде означати виконання замовлення заявника про заявлений вид дослідження.

Кінцевим результатом роботи всього відділу є відправити звіт з дослідів замовнику.

Програмна і технічна архітектура.

На кафедрі під рецензованим встановлено п'ять персональних комп'ютерів.

Чотири комп'ютери мають таку конфігурацію:

- а) процесори: Intel Core 2 Duo 2,6 ГГц;
- б) 3 Гб оперативної пам'яті;
- в) 1 ТБ HDD;
- г) відеокарти: GFX1024 MB;
- д) DVD;
- е) монітори 19" LG.

Комп'ютер керівника:

- а) процесор: IntelPentiumIV 2,6 ГГц;
- б) 2Gb ОЗУ;
- в) 320 Gb HDD;
- г) відеокарта: GF 6200 256 MB;
- д) ACER 17" монітор.

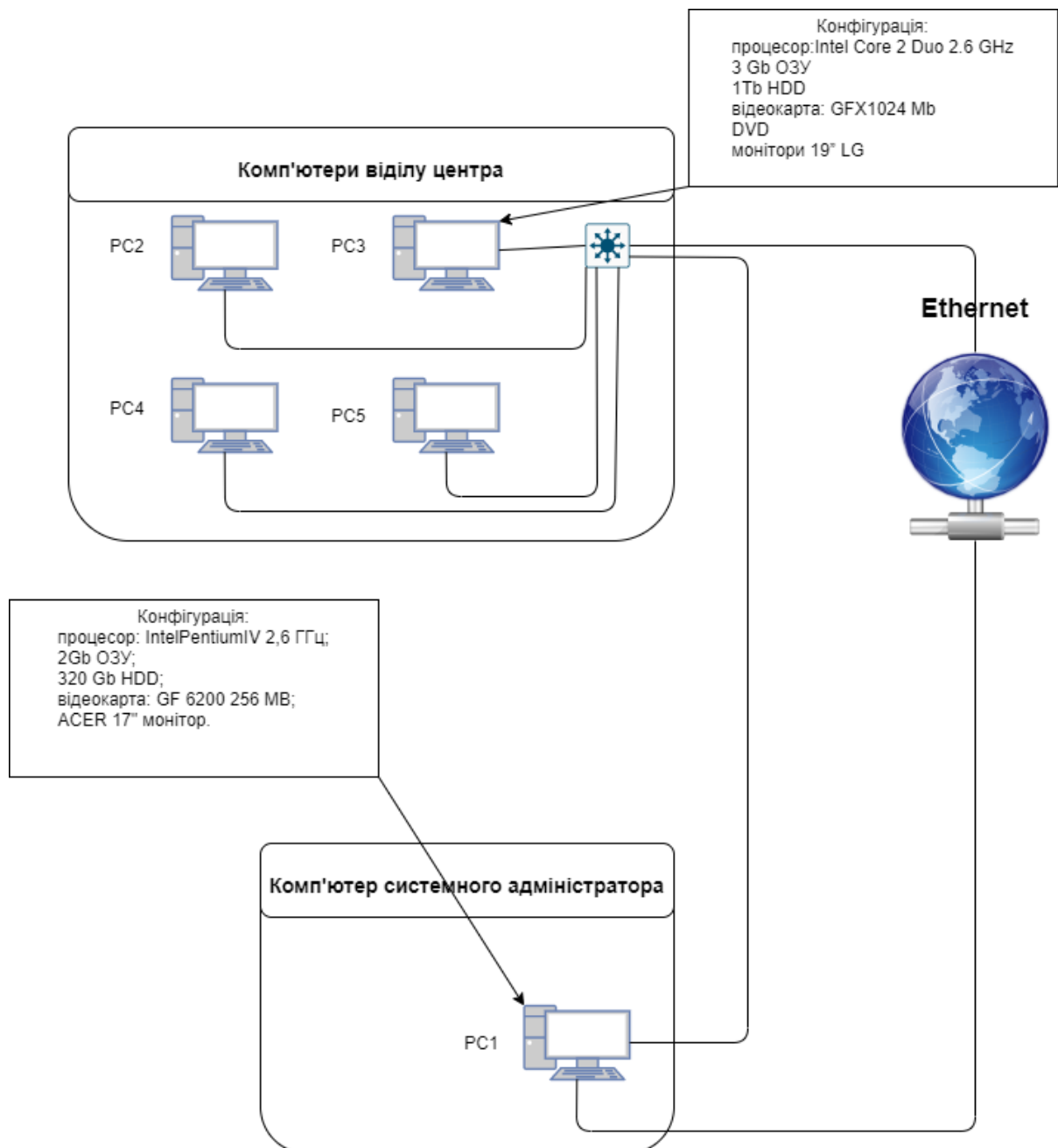


Рис. 1.1. Схема технічної архітектури IC

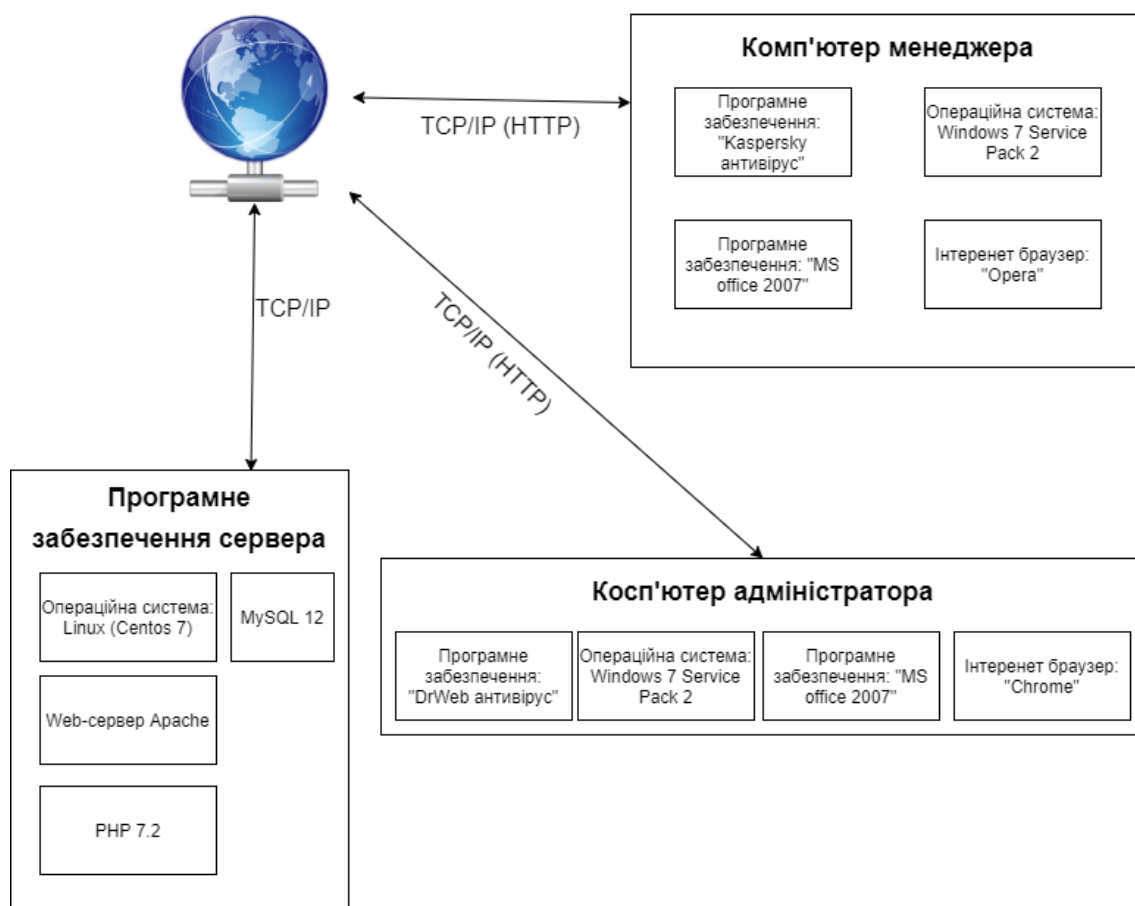


Рис. 1.2. Схема програмної архітектури

На комп'ютерах користувачів також встановлено такі програмні продукти:

- а) ОС Windows 7 Service Pack 2\$
- б) DrWeb AntiVirus;
- в) MS Office 2007;
- г) Браузер Opera.

Конфігурація серверного програмного забезпечення включає:

- а) Linux Centos 7;
- б) Веб-сервер Apache;
- в) MySQL 12;
- г) PHP 7.2.

На даний момент, центр колективного користування науковим обладнанням «Лабораторія біобезпеки, якості харчової продукції та безпеки

харчування» не застосовує системи управління підприємством, але активно використовує програми MS Office: MS Word, MS Excel пакет програмного забезпечення Office. У той же час, як інформаційні канали для відправки та отримання інформації існують:

- а) Електронної пошти;
- б) Телефонна та факсимільної лінії;
- в) Доставка кур'єром;
- г) Це просто звичайна пошта.

Офіс має локальну однорангову мережу для підключення комп'ютерів агентства один з одним і FTP-мережу, яка є позитивним розвитком і подальшим впровадженням автоматизованого робочого місця адміністратора сайту.

1.2 Дослідження основних функцій роботи об'єкта автоматизації

Основними завданнями проблемної науково-дослідної лабораторії є:

- проведення фундаментальних досліджень згідно з пріоритетними напрямками розвитку науки і техніки і виконання прикладних розробок за напрямками науково-технічної діяльності;
- підтримка існуючих та формування нових наукових шкіл; - підтримка наукових досліджень молодих вчених, спеціалістів і обдарованих студентів, залучення їх до діючих наукових шкіл;
- участь у підготовці в НУХТ кваліфікованих фахівців, наукових та науково-педагогічних кадрів вищої кваліфікації на основі новітніх досягнень науково-технічного прогресу;
- ефективне використання наукового і науково-технічного потенціалу університету для вирішення пріоритетних завдань оновлення виробництва та проведення соціально-економічних перетворень.

Науково-дослідні роботи виконуються в проблемній науково-дослідній лабораторії на підставі тематичних планів, що формуються згідно з

пріоритетними напрямками наукової діяльності НУХТ та відповідають пріоритетним напрямам розвитку науки і техніки. Наукові напрями ПНДЛ розробляються виходячи із задач, поставлених при її організації та з урахуванням профілю підготовки фахівців у НУХТ.

Для організації роботи за тематикою за поданням наукових керівників тем згідно з діючим штатним розписом і фактичним фінансуванням видатків проблемної науково-дослідної лабораторії формується склад наукових (тематичних) груп.

Наукові (тематичні) групи очолюють наукові керівники - доктори і кандидати наук, які несуть відповідальність за своєчасність та результативність виконання досліджень, розв'язання комплексних задач у сфері наукового, технологічного розвитку, впровадження та використання в Україні наукових і науково-практичних результатів.

Проблемна науково-дослідна лабораторія проводить дослідження в тісному контакті з кафедрами, факультетами, науково-дослідними інститутами, галузевими науково-дослідними лабораторіями та іншими структурними підрозділами НУХТ, вищими навчальними закладами (ВНЗ) і науковими установами Академій наук України.

У складі ПНДЛ у необхідних випадках ректором НУХТ можуть бути створені сектори (відділи) за відповідними напрямками наукової діяльності.

Кошти з Державного бюджету України на фінансування досліджень у ПНДЛ виділяє МОН України. Фінансові операції та облік матеріальних активів ПНДЛ виконуються бухгалтерією НУХТ.

Штатний розпис ПНДЛ погоджується з проректором з наукової роботи та затверджується ректором НУХТ.

ПНДЛ знаходиться на балансі НУХТ. За ПНДЛ закріплюються приміщення, наукові прилади та устаткування, придбані за кошторисами на її забезпечення і за рахунок інших джерел.

Матеріально-технічне забезпечення проблемної науково-дослідної лабораторії здійснюється відповідними підрозділами НУХТ.

ПНДЛ надається оперативна самостійність у визначенні напрямків наукової діяльності в межах затверджених планів наукових робіт та кошторисів.

Перелік тематик, які виконуються в ПНДЛ охоплює різні галузі знань від економіки і до власне харчової галузі. Нижче у вигляді переліку подані переліки науково-дослідних робіт, що фінансуються із коштів Державного бюджету Міністерством освіти і науки:

- «Комплексна оцінка та шляхи підвищення конкурентоспроможності страхового ринку України в контексті європейської інтеграції»;
- «Створення наукових основ технологічних процесів відновлення води до її природного структурно енергетичного стану»;
- «Науково - технічні засади створення енергоефективних, багатофункціональних пакувальних машин харчових продуктів на основі неоднорідної системи мехатронних модулів»;
- «Баро- та електромембранні процеси в технологіях очищення рідких середовищ харчової промисловості»;
- «Наукові засади розроблення ресурсощадних технологій білоквмісних поліфункціональних концентратів для харчових продуктів цільового призначення».

У Проблемній науково-дослідній лабораторії використовуються різноманітні методи і процедури, що відповідають її галузі діяльності. Вони охоплюють відбір зразків, підготовку зразків та зберігання зразків, що підлягають випробуванню, проведення випробування, статистичні методи аналізування даних випробування.

У ПНДЛ використовують методи, викладені в національних стандартах, викладені в методичних вказівках або рекомендовані виробником обладнання.

Також у ПНДЛ розробляють методи для власного користування, які розробляють високо компетентні фахівці з різних галузей знань, що працюють в ПНДЛ.

Обладнання Проблемної науково-дослідної лабораторії включає:

- засоби вимірювальної техніки;
- випробувальне обладнання;
- допоміжне обладнання.

У ПНДЛ наявне обладнання для проведення відбору зразків, вимірювань та випробувань.

Обладнання та його програмне забезпечення, що використовують для проведення випробувань, здатне досягти необхідної точності та відповідає технічним вимогам, що висувають до випробування. Перед введенням в експлуатацію обладнання перевіряють на відповідність технічним вимогам та відповідним стандартам:

- засоби вимірювальної техніки – шляхом повірки або державної метрологічної атестації;
- випробувальне обладнання – шляхом атестації;
- допоміжне обладнання – шляхом перевірки за експлуатаційною документацією.

З обладнанням працює уповноважений персонал. Актуалізовані інструкції з використання та обслуговування знаходяться у ЗЛ.

Кожна одиниця обладнання ідентифікована на відповідність паспортним даним та зареєстрована в реєстраційних картках.

Для розгляду замовлення на дослідження спочатку користувач оформлює замовлення на дослідження в яке вносить данні про організацію, ПІБ замовника в рамках якої науково дослідницької роботи буде проводитися дослідження та інші необхідні данні. Після чого з директором Центру узгоджують можливість проведення досліджень. Дату проведення дослідів також необхідно вибирати так як деякі досліді проводяться по 24 години, а обладнання є унікальним в своєму роді і не завжди є доступним. Після чого

зразки матеріалу незадовго до дослідів надсилаються в Центр та ставляться для дослідження. І одною з головних частин роботи науково дослідницької лабораторії є надання звітів з проведених досліджень та аналітичних відомостей.

Типова послідовність застосування-обслуговування клієнтів відбувається в такій послідовності:

- а) вибрано потрібну категорію досліджень;
- б) заявник подає заявку на замовлення;
- в) потім-перевірити правильність конструкції;
- г) центр відправляє рахунок і контролює оплату (у випадку платних послуг);
- д) Здійснюється обслуговування та надіслано документи, замовлені заявником.

Концептуальна модель роботи лабораторії представлена на рисунку 1.3.

Функціональна модель роботи діяльності лабораторії біобезпеки, якості харчової продукції та безпеки харчування НУХТ представлена у додатку А. Модель містить контекстну діаграму та діаграми декомпозиції. Розроблена функціональна модель описує процеси, що відбуваються у лабораторії біобезпеки, якості харчової продукції та безпеки харчування НУХТ.

З контекстної діаграми декомпозується та представляє набором основних бізнес-процесів, що проходять у лабораторії біобезпеки, якості харчової продукції та безпеки харчування НУХТ.

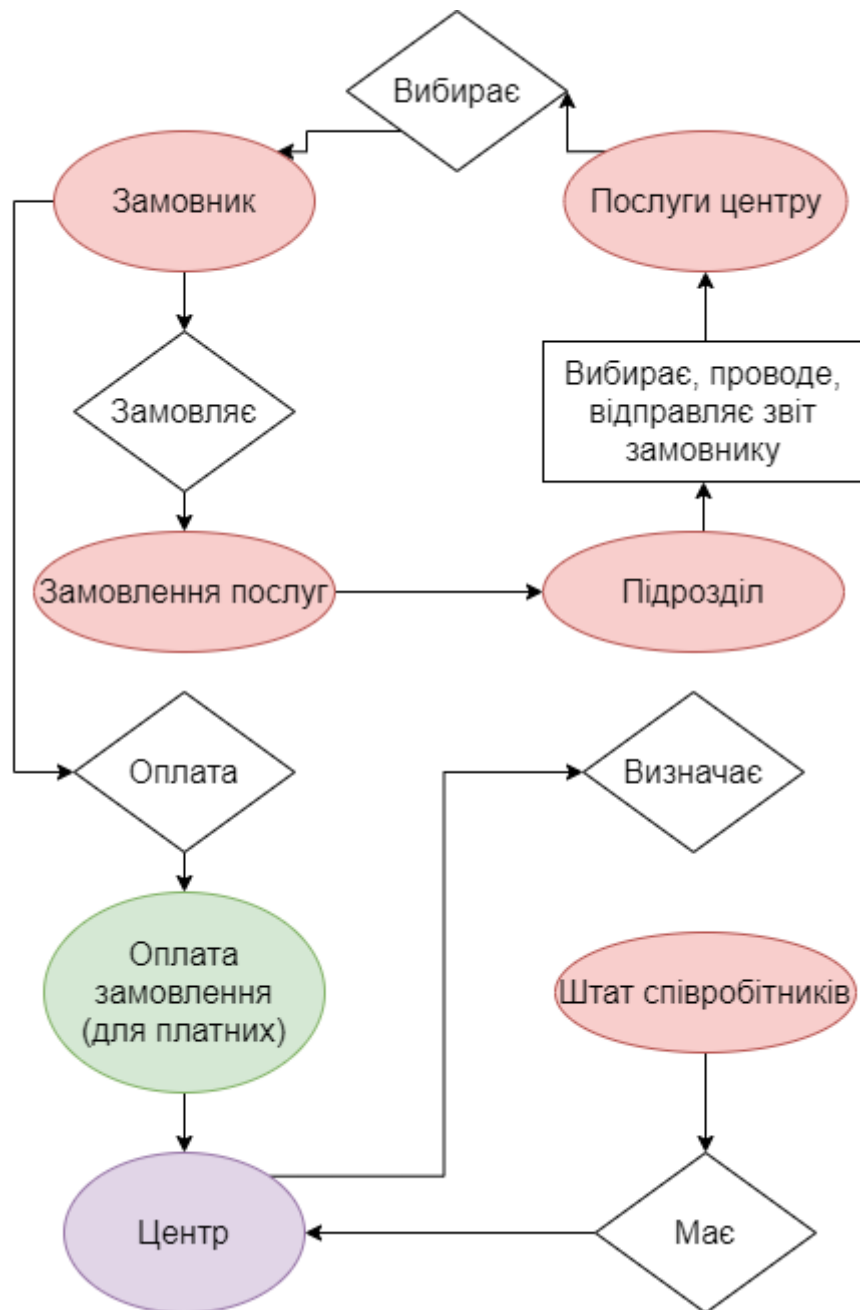


Рис. 1.3. Концептуальна модель впровадження запитів користувачів

1.3 Аналіз існуючого на підприємстві програмного забезпечення

На даний момент Центр не володіє спеціальним програмним забезпеченням для підтримки своєї діяльності, оскільки, це відносно молоде підприємство. В основному використовуються офісні пакети MS Word, Excel, але підприємство зростає, збільшується кількість замовлень і досить гостро

відчувається потреба у сучасній інформаційній системі для задовольняти нагальних потреби, а також майбутніх перспектив компанії.

1.4 Огляд існуючих рішень

Під час дослідження ринку на вже рішення були знайдені наступні варіанти:

1. Gincore;
2. Qsystem;
3. Бітрікс24;
4. amoCRM;
5. FreshOffice;
6. Smarty CRM.

Головним недоліком в більшості систем є те що вони розроблені для ведення бізнесу та заробітку і не є досить гнучкими. Одною із поставлених задач від керівництва центру при розробці було створення системи до котрої може зайти як користувач із фіксованого списку так і управління цим адміністратором.

Але майже всі системи заточені під роботу персоналу центру і не дозволяють(без переробки функціоналу системи) користувачу зі сторони залишати замовлення та перевіряти їх стан і завантажувати результати.

Всі вище зазначені системи розраховані на роботу з продажом товарі і хоча послуги також є товарами але надання електронної інформації як товару в самих системах непередбачене.

Також деякі з вище зазначених систем можуть працювати лише на віддалених серверах компаній що виготовила їх та не мають можливості пакетної установки на власне обладнання що є не припустимо.

1.5. Обґрунтування доцільності проектування і розробки інформаційної системи

1.5.1. Мета та цілі розвитку Web-орієнтованої інформаційної системи лабораторії біобезпеки

Метою проекту є розробка веб-орієнтованої інформаційної системи, веб-сайту (більше веб-додатків, системи) для підвищення ефективності роботи через нові інформаційні технології та Інтернет.

Система призначена, як для виходу центру колективного користування науковим обладнанням «Лабораторія біобезпеки, якості харчової продукції та безпеки харчування» в світовий інформаційний простір мережі Інтернет, так і для внутрішніх потреб в управлінні державної установи.

Таким чином, користувачі системи діляться на кілька основних груп:

- а) адміністративний персонал;
- б) працівників державних органів;
- в) користувач центру від навчальні заклади;
- г) відвідувачі сайту (гість).

У той же час, Web-орієнтована інформаційна система обробки та аналізу даних дозволить досягти наступних позитивних моментів:

а) мінімізації термінів і надання якісної інформації, надання інформації про послуги, процедури, а не періодично публікувати розклади та іншу інформацію про діяльність організації в пресі;

б) скоротити часові рамки для підготовки рішень керівництвом адміністративної групи установи;

в) збільшити кількість заявок і швидкість обробки;

г) мінімізувати трудомісткі та тривалість рутинних технічних операцій, пов'язаних з підготовкою вихідних документів;

д) для створення банку даних, для зберігання інформації про заявників, що дозволить звести до мінімуму час введення вихідної інформації при багаторазове звернення заявників.

Цілями проекту є:

- а) розробка веб-орієнтованої інформаційної системи, яка стане частиною веб-сайту НУХТ;
- б) розробити базу даних;
- в) розробити програмне забезпечення для управління сайтом, бази даних, інтерфейс користувача;
- г) підвищення ефективності роботи центру;
- д) підтримка контролю основних витрат праці на проведення досліджень у лабораторії.

Загальна характеристика системи з точки зору теорії інформаційних систем.

Інформаційна система реалізує наступні інформаційні процеси:

- а) збір інформації;
- б) зберігання інформації;
- в) обробка інформації;
- г) видачі інформації.

За виконання цих процесів відповідають складові інформаційної системи. Збір і надання інформації, як правило, через користувацький інтерфейс.

1.5.2 Розробка вимог до функціональних характеристик і до якості програмного продукту

Основними вимогами до системи є:

- а) можливість виконання всіх необхідних операцій для роботи з вихідними даними та інформацією про результати (розмежування доступу до даних, введення, зміна і видалення, друк, пошук);
- б) мати чіткий і зручний інтерфейс.
- в) система повинна надавати всі функції даних:
- г) додавання/зміна/видалення різних груп;
- д) додавання/зміна/видалення підрозділів;
- е) додавання/зміна/видалення клієнтів;

- ж) додавання/зміна/видалення співробітників;
- з) реєстрація груп користувачів;
- и) додавання/зміна/видалення підрозділів;
- к) додавання/зміна/видалення замовлень;
- л) формування звітів про виконані замовлення та виконану роботу.

Вимоги до складу та специфікацій апаратного забезпечення представлені в таблиці 1.1 з урахуванням функціональності системи та обсягу даних, що обробляються.

Таблиці 1.1. Апаратні вимоги та технічні характеристики

Характерні	Рекомендовані вимоги
Процесор	Intel Pentium 3/AMD Боратлону, 700MHz
Оперативної пам'яті	1024 MB
Обсяг дискового простору	350 MB
Роздільна здатність екрана відео	1024 × 768 пікс
Принтер	Hp LaserJet p1102

1.6 Постановка задачі на розробку інформаційної системи

Постановка цілей і завдань створення, Web-орієнтованої інформаційної системи лабораторії біобезпеки, якості харчової продукції та безпеки харчування НУХТ.

Від керівництва Центру було отримане усне замовлення на розробку інформаційної системи. В котрому

Процесу розробки будь-якого сайту, веб-орієнтованої ІС повинна передувати вироблення мотивації створення інтернет - сайту, веб-орієнтованої ІС. Важливо на цей момент усвідомлення сформулювати конкретні цілі і завдання проекту - для чого і як це зробити. Неможливо і

дорого досягти миттєво всіх цілей і вирішити всі поставлені завдання, але, безумовно, їх необхідно ставити і послідовно вирішувати, отримуючи безцінний досвід створення і просування інтернет ресурсу в інформаційний простір.

Цілі для веб-орієнтованих ІС. Основними цілями є:

- а) Підвищення швидкості обробки клієнтських додатків
- б) Пряма комунікація з споживачами сервісу
- в) Отримання інформації про актуальні питання, Відгуки про послуги, Оперативне реагування на пропозиції;
- г) Орієнтація на онлайн-сервіси, онлайн-додатки та платежі;
- д) Накопичення статистичних матеріалів і досліджень у статистиці;
- е) Перехід Центру на нові технології роботи на базі архітектури клієнт-сервер, веб-технології, Інтернет-програмне забезпечення.

Розробка веб-орієнтованої інформаційної системи, веб-сайту (більше веб-додатків, систем) для підвищення ефективності роботи лабораторії біобезпеки, якості харчової продукції та безпеки харчування через нові інформаційні технології та Інтернет.

Висновки до розділу 1

Провівши огляд діяльності державного підприємства - центр колективного користування науковим обладнанням «Лабораторія біобезпеки, якості харчової продукції та безпеки харчування»(далі Центру), розумієш необхідність модернізації збору та обробки заявок на витребування документів від громадян, які перебувають в різних точках світу. Модернізація цього процесу дозволить значно скоротити час надходження заявок до відділу, скоротить втрату даних про заявників і потрібних документів документах, дозволить своєчасно і оперативно отримувати інформацію про стадію виконання заявки, створити банк даних про заявників і наданих документах, і, як наслідок, за рахунок мінімізації трудомісткості і тривалості виконання рутинних робіт, підвищить рівень роботи Центру.

Метою проекту є розробка веб-орієнтованої інформаційної системи, веб-сайту для підвищення ефективності діяльності Центру за рахунок нових інформаційних технологій і мережі інтернет, розробка деяких компонент бази даних ІС, розробка компонент програмного забезпечення управління сайтом, базою даних, інтерфейсом користувачів.

Розробляється інформаційна система дозволить здійснювати пошук інформації по заданим критеріям, отримувати звіти про виконання робіт, надавати відомості по співробітниках, виконаних роботах.

РОЗДІЛ 2 РОЗРОБКА КОМПЛЕКСУ ЗАДАЧ АВТОМАТИЗАЦІЇ

2.1 Обґрунтування вибору програмного забезпечення та технології

Клієнт-серверна архітектура розглядається як одне з можливих концепцій створення інформаційних систем, а сімейна платформа Unix-Linux пропонується як фундаментальна платформа для реалізації програмного забезпечення такої архітектури.

Програмне забезпечення, яке реалізує управління базами даних і функції пошуку, розроблений як веб-додаток (назвемо його просто "додатком"). За визначенням, веб-додаток є клієнт-серверним додатком, в якому клієнт є браузером, а сервер є веб-сервером. Логіка веб-застосунку розподіляється між сервером і клієнтом, дані зберігаються в основному на сервері, інформація обмінюється по мережі.

Однією з переваг такого підходу є те, що заявники не залежать від операційної системи користувача, тому веб-додатки є крос-платформними службами (Wikipedia).

Клієнтська частина програмного забезпечення реалізує інтерфейс користувача, генерує запити до сервера і обробляє відповіді від нього. Частина сервера отримує запит від заявника, процесів, формує веб-сторінку та надсилає його заявнику через мережу.

Для реалізації Клієнт-серверної архітектури вибрано тріо базового програмного забезпечення, що базується на Linux платформі - віддалений сервер зі стандартним програмним забезпеченням Apache, PHP, MySQL і широко поширений пакет інтерактивного віддаленого управління базою даних phpMyAdmin. Ці програмні засоби є тим середовищем, на основі якої функціонують більшість веб-сайтів і веб-додатків. На поточний рік на цій безкоштовній платформі функціонує близько 70% веб-сайтів.

Ще однією важливою перевагою є відкритий код більшості програмних продуктів, заснованих на цій платформі. Це те, що було використано в цьому додатку.

Більшість веб-додатків базується на веб-серверах загального користування, так званих поділюваних хостах (shared hostings), коли на одному або декількох фізичних серверах розташовується безліч веб-сайтів, веб додатків.

Доступ до таких віртуальним хостам, що знаходяться в користуванні окремих додатків здійснюється, як правило, за допомогою FTP- доступу. Для цього створені спеціальні клієнтські програми типу GuteFTP 8 Professional і безліч інших. За допомогою FTP- Клієнтів можна легко адмініструвати веб-додатки на сервері.

СУБД MySQL - це система керування базами даних, яка, в тому числі, працює в зв'язці безкоштовних потужних продуктів, таких як Unix, Linux ..., Apache, PHP, що забезпечує потужні інструменти підтримки і швидку розробку веб-додатків. MySQL є багатокористувацьким, «багато потоковим» сервером і використовує стандартний, прийнятий програмістські світі структурована мова запитів SQL (Structured Query Language).

Для розробки веб-додатків використовується також широко прийнятий у всьому світі мову високого рівня PHP. Мова PHP працює на стороні сервера і забезпечує створення динамічних веб-додатків, в тому числі і взаємодія з базою даних MySQL, за допомогою безлічі функцій і класів. Програми написані, зокрема, на PHP називаються сценаріями.

Цей проект використовував наступні мови і технології для реалізації веб-додатків: HTML, CSS, PHP, SQL, MySQL. Основним є мова PHP. На ньому написано 90% коду цього додатка.

HTML (від англійського Hyper Text Markup Language – мова розмітки гіпертексту). Це базова мова в області засобів створення веб сторінок. Мова HTML призначений для формування та оформлення веб-сторінок, які

інтерпретується за допомогою спеціальних Клієнтських програм - браузерів (Internet Explorer, Mozilla Firefox, і ін.).

CSS (каскадні таблиці стилів або таблиці каскадних стилів) – набір параметрів форматування, застосованого до елементів веб-сторінки для контролю їх перегляду.

PHP – скриптова мова програмування, створена для генерації HTML-сторінок на Веб-сервері і роботи з базами даних. На даний момент він підтримується 60% - ми постачальників хостингу, входить в «стандартний» набір для створення сайтів (LAMP - Linux, Apache, MySQL, PHP), працює на стороні сервера.

SQL (Structured Query Language) – створено для роботи з реляційними базами даних. Він дозволяє користувачам взаємодіяти з базами даних (переглядати, шукати, додавати, керувати даними). MySQL - розрахований на багато користувачів, багато-сервер бази даних SQL. Має хорошу швидкість і гнучкість, якщо використовувати його для зберігання зображень і файлів. MySQL відповідає специфікації ANSI 92 SQL [2].

Існують різні методи створення веб-додатків, веб-сайтів. Самий тривіальний метод з точки зору технології, але не в тому сенсі, що необхідні знання для створення-це верстка HTML сторінок вручну. Ви можете використовувати будь-який текстовий або спеціалізований редактор. Крім знання HTML, знання CSS також знадобляться. Ось що стосується найпростішого дизайну сайту. Переробляючи сайт, ви можете почати просувати його, поліпшуючи дизайн. Крім того, ви можете використовувати різні онлайн дизайнери сайту або готові шаблони, а потім змінити, поліпшити дизайн сайту. Це стосується статичних сайтів, які мають мінімалістичний дизайн і готові до використання. Крім того, скоро буде зрозуміло, що неможливо обійтися без JavaScript, який, в даному випадку, працює на стороні клієнта і з його допомогою можна домогтися як поліпшеної конструкції сайту, так і його функціональності. Наприклад, Ця типова операція характерна для будь-якого сайту як заповнення різних видів

форм (зворотній зв'язок, реєстрація даних користувача, електронна пошта тощо) неможлива без JavaScript або jQuery.

Ще один спосіб створити сайти, які є найбільш популярними сьогодні є використання інструментів, званих систем управління контентом (CMS). CMS, спрощених союзником, являє собою готову оболонку програмного забезпечення, які можуть бути заповнені з певним змістом, а також змінені, налаштовані на його вимоги останній більш популярним з-за відкритого коду.

- а) Joomla – 38, 95%;
- б) WordPress – 31, 52%;
- в) Двигун DataLife – 10, 16%;
- г) Усог – 7, 08%;
- д) Drupal – 6, 34%.

Таким чином, методи ручного макету при створенні веб-додатків досить складні, так як вони вимагають значних знань в області веб-програмування-знання ряду мов і програмного забезпечення.

Тим не менш, вони мають важливе значення: шляхом створення сайту вручну, ви завжди можете побудувати тільки один з функціями, які розроблені.

Створення сайтів на основі безкоштовних онлайн дизайнерів зручно для тих, хто хоче швидко створити і «запустити свій сайт у світ».. Цей метод підходить для створення невеликих простих сайтів з обмеженою функціональністю і динамічністю.

CMS надає широкий спектр можливостей для створення сайтів будь-якої складності. Такий спосіб створення сайтів вважається одним з найбільш зручних і практичних.

Однак цей проект не використовував CMS для створення сайтів, але використовував ручне компонування HTML-CSS коду.

2.2 Методи вирішення задачі

2.2.1 Проектування бази даних

2.2.1.1 Логічна структура клієнтської бази даних

Процес зберігання інформації полягає в довготривалому зберіганні даних на носіях, у вигляді масивів даних, з певною структурою. В даний час, для зберігання інформації використовуються бази даних.

База даних (БД) може бути визначена як сукупність взаємопов'язаних даних, що використовуються декількома користувачами або програмами і зберігаються з регулярною надлишковістю. Збережені дані не залежать від програм користувачів, для модифікації/внесення змін застосовується загальний управляючий метод, загальне програмне забезпечення.

Обробка інформації полягає в отриманні деяких "об'єктів інформації" з інших "інформаційних об'єктів" за допомогою виконання алгоритмів і є однією з основних операцій, що здійснюються над інформацією.

Наведені нижче основні процедури обробки інформації включають:

- а) пошук інформації;
- б) прийняття рішень;
- в) створення даних;
- г) модифікація даних;
- д) забезпечення цілісності даних.

Інформаційна система складається з наступних складових:

- а) бази даних;
- б) інструменти обробки даних;
- в) інтерфейсі користувача.

За допомогою методу обробки даних інформаційні системи поділяються на системи з централізованою та децентралізованою організацією даних.

Централізована система має найпростішу реалізацію. Використовується одна база даних, а всі транзакції даних здійснюються

одним сервером баз даних. Таким чином, головною перевагою централізованої системи є відсутність необхідності спеціальних інструментів для забезпечення цілісності та актуальності даних. Головним недоліком є обмеження щодо розміру бази даних та кількості запитів, що обробляються можливостями одного серверу.

Коли ви децентралізуєте інформаційну систему, база даних розбита на частини по різних серверах.

Існують два основні підходи до організації децентралізованих інформаційних систем. Користувачі використовують локальну базу даних, яка містить копії частин бази даних спільної інформації. Повна копія інформаційної бази міститься на декількох серверах.

У першому випадку деякі запити можна обробляти лише за допомогою локальної бази даних, що зменшить навантаження на центральному сервері. У другому випадку кожен сервер обробляє запити від своїх клієнтів, розподіляючи тим самим навантаження між серверами.

У будь-якому випадку децентралізовані системи повинні тримати дані в актуальному стані в різних базах даних. Це ставить додаткові навантаження в системі, коли дані зміни.

Традиційно, інформаційні системи використовують WIMP користувальницький інтерфейс, наданий більшістю сучасних операційних систем, але веб-орієнтованих інформаційних систем стали все більш поширеними останнім часом.

Веб-орієнтована Інформаційна система реалізована як клієнт-серверний додаток, в якому клієнт є браузером, а сервер є веб-сервером.

Ця організація пропонує наступні переваги: веб-браузер вбудовано в більшість операційних систем, тому функції розробки, встановлення, оновлення та підтримки клієнтської частини не лежать в основі розробника інформаційної системи. Логіка інформаційної системи орієнтована на сервер. Крім того, клієнти не залежать від конкретної операційної системи користувача, а інформаційна система, таким чином стає між платформною.

Системні функції реалізуються один раз, замість того, щоб розроблятися для кожної операційної системи як: Microsoft Windows, , GNU/Linux, Mac OS X та інших.

Через цю універсальність і відносну простоту розробки веб-додатків, веб-додатки стали широко популярними в кінці 1990-х і початку 2000-х.

Коли об'єкти та атрибути перетворюються, об'єкти переміщуються до таблиць бази даних користувача, а атрибути об'єкта стають атрибутами таблиці.

Структура бази даних може змінюватися в принципі, але всі заголовки табличних заголовків на екрані повинні також змінюватися в синхронізації. Це не суттєво впливає на програмне забезпечення, яке підтримує веб-додатків. Структура таблиць представлена у таблиці 2.1.

Таблиці 2.1. Об'єкти та атрибути

Об'єкт	Атрибут	Ключовим
Клієнти	idКлієнта Номер Адреса Контакти клієнта Телефон E-mail Нотатки	idКлієнта
Відділи	idВідділ Назва Місце розташування Телефон Головний виконуючий Нотатки	idВідділ
Групи	idГрупи	idГрупи

	Ім'я Локація Телефон Менеджер Нотатки	
Замовлення	idЗамовлення Клієнт_id Дата подачі замовлення Дата початку Дата завершення Кодекс виконавчого відділу Ім'я виконавця Коментар	idЗамовлення
Платежі	idПлатежа Дата Клієнт_id Додаток_id Сума платежа	idПлатежа
Послуги	idПослуги Ім'я послуги Деталі Вартість Коментарі	idПослуги
Персонал	idПерсоналу Код співробітника Позиція Сертифікат Нотатки	idПерсоналу

2.2.1.2 ER-модель бази даних заявників

Модель сутностей-відносин, Ерм, ER Model, дозволяє описати концептуальні схеми предметної області.

Модель ER використовується у дизайні бази даних високого рівня. Він може ідентифікувати ключові сутності та визначати підключення, які можуть бути встановлені між цими сутностями. ER модель є офіційним дизайном, який не визначає графічних засобів його подання. У якості стандартного графічного представлення моделі ER була розроблена ER-діаграма сутність-сутність. Коли бази даних розроблено, ER модель перетворюється на конкретну схему бази даних.

Як відомо, основна концепція реляційної бази даних — це таблиця (зв'язок). Таблиця використовується для структури та зберігання інформації. У реляційних базах даних Кожна клітинка таблиці містить одне значення. Крім того, в межах однієї бази даних є зв'язки між таблицями, кожен з яких встановлює спільний доступ до табличних даних.

Діаграма ER графічно являє собою структуру даних проектованої бази даних. Сутності відображаються за допомогою прямокутників, таблиць, які містять ім'я сутності-таблиці БАЗИ даних. Зв'язки сутностей відображаються як лінії, що з'єднують окремі сутності.

Зв'язок показує, що дані з однієї сутності посилаються або зв'язані з даними іншого.

Ступінь кінця з'єднання графічно вказується, множинність зв'язку зображується як «вилка» в кінці з'єднання. Також графічно зображується модальність зв'язку-додаткове з'єднання позначено колом в кінці з'єднання. Назва посилання виражається одним дієсловом: "має", "належить" тощо. (Рис. 2.1) [23].

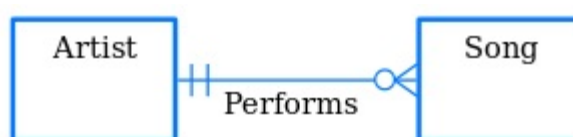


Рис. 2.1. Одне з можливих позначень посилань на графічні зображення

Атрибути суб'єкта господарювання записані всередині прямокутника з зображенням суті і вираженою іменником в сингулярній.

Серед атрибутів виділяється ключ суб'єкта господарювання. Під час створення сутності потрібно вибрати групу атрибутів, які можуть бути первинним ключем, а потім вибрати атрибути для включення до первинного ключа:

- а) Первинний ключ слід вибрати таким чином, щоб значення атрибутів, включених до нього, можна точно визначити копію сутності;
- б) Жоден із атрибутів первинного ключа не повинен дорівнювати нулю;
- в) Значення атрибутів первинного ключа не повинні змінюватися.

Якщо значення змінено, то це ще один екземпляр сутності.

При виборі первинного ключа, як правило, вставляється додатковий атрибут у сутність, яка стає ключем. Наприклад, унікальні числа часто використовуються для визначення первинного ключа, який може автоматично створюватися системою, коли копія сутності додається до бази даних. Використання унікальних номерів полегшує процес індексації та пошуку в базі даних.

Першим кроком у створенні логічної моделі DB є побудова діаграми ER, що складається з трьох частин: сутностей, атрибутів та зв'язків.

Розроблено багато візуальних інструментів для різних платформ. Цей проект використав розвиток MySQL Workbench.

Цей інструмент спрощує проектування і обслуговування баз даних, полегшує трудомісткий і схильний до помилок завдання, і покращує зв'язок між розвитком команди і DB архітекторів. Це дозволяє архітекторам даних візуалізувати вимоги, спілкуватися з зацікавленими сторонами дизайну, перш ніж інвестиції вносяться в проект. Це забезпечує проектування бази даних на основі моделі, яка є найефективнішою методикою для створення реальних і добре розроблених баз даних. На зображенні показано діаграму ER

(control_clients_mid) (діаграма 2.2). Дві вертикальні риси з одного боку і тризуб на інший тим часом «один-до-багатьох».

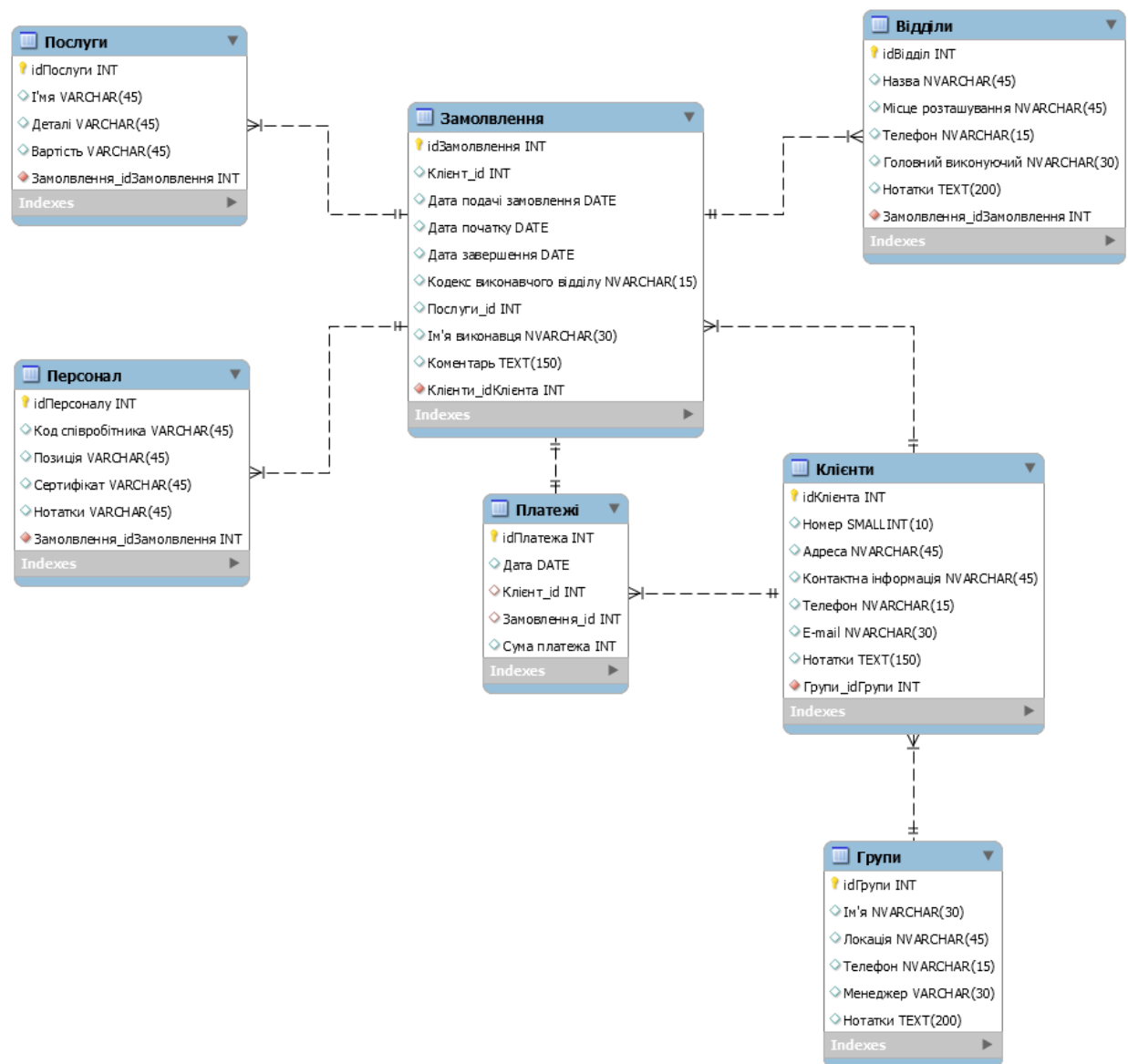


Рис. 2.2. База даних в EER-діаграмі

2.2.1.3 Нормалізація системної бази даних

В реляційній теорії, немає універсальних норм, законів для розробки надійної, призначеної бази даних раз і назавжди. Розробники вибирають свої рішення для проекту бази даних, покладаючись в основному на досвід, інтуїцію, різні інструменти і методи проектування.

Однак, деякі канони, правила існують. Такі правила включають в себе правила нормалізації, тобто приведення відносин до нормального життя.

Як зазначає к. дані, загальна мета процесу нормалізації полягає у ліквідації надмірностей, ліквідації аномалій оновлення, гнучкості у зміні та розширенні проекту бази даних, а також спрощення процедур застосування необхідних обмежень цілісності.

Нормальна форма — це зв'язок в реляційній моделі даних, що характеризує її з точки зору надмірності та цілісності та відповідає певним вимогам.

- а) Перша нормальна форма (1NF);
- б) Друга нормальна форма (2NF);
- в) Третя нормальна форма (3NF).

В основі процесу проектування лежить метод розкладання відносин, що знаходиться в попередній нормальній формі, у двох (або більше стосунках), що відповідають вимогам наступної нормальної форми.

У цьому проекті здійснено часткову нормалізацію. Справа в тому, що в реальному, "промислових" базах даних, розробники вибирають між нормалізації до "переможного кінця" і складність програмного забезпечення, що реалізує запити бази даних і процедури адміністрування баз даних.

2.2.1.4 Прийнята концепція архітектури програмного забезпечення

Програмне забезпечення, яке реалізує управління базами даних і функції пошуку, розроблений як веб-додаток (назвемо його просто "додатком"). За визначенням, веб-додаток є "клієнт-серверним додатком, в якому клієнт є браузером, а сервер є веб-сервером. Логіка веб-застосунку

розподіляється між сервером і клієнтом, дані зберігаються в основному на сервері, інформація обмінюється по мережі.

Однією з переваг цього підходу є те, що клієнти не залежать від конкретної операційної системи користувача, тому веб-додатки є крос-платформними службами.

Клієнтська частина програмного забезпечення реалізує інтерфейс користувача, генерує запити до сервера і обробляє відповіді від нього. Частина сервера отримує запит від клієнта, обробляє його, формує веб-сторінку та надсилає його клієнту через мережу.

Тріо базового програмного забезпечення на базі Linux платформи, віддалений сервер зі стандартними Apache, PHP, MySQL програмного забезпечення і широко розподілений пакет інтерактивного дистанційного керування базою даних PhpMyAdmin, були обрані для реалізації клієнт-серверна архітектура.

Ще однією важливою перевагою є відкритий код більшості програмних продуктів, заснованих на цій платформі. Це те, що було використано в цьому додатку.

Більшість веб-додатків засновані на публічних веб-серверах, так званих спільних хостах, де один або кілька фізичних серверів приймають багато веб-сайтів і веб-додатків.

Доступ до таких віртуальних хостів, що знаходяться в користуванні окремих додатків здійснюється, як правило, за допомогою FTP-доступу. Для цього створені спеціальні клієнтські програми типу GuteFTP 8 Professional і безліч інших. За допомогою FTP-клієнтів можна легко адмініструвати веб-додатки на сервері.

2.2.1.5 Реляційна база даних MySQL

СУБД MySQL - це система керування базами даних, яка, в тому числі, працює в зв'язці безкоштовних потужних продуктів, таких як Unix, Linux,

Apache, PHP, що забезпечує потужні інструменти підтримки і швидку розробку веб-додатків.

MySQL є багатокористувацьким, «багато потоковим» сервером і використовує стандартний, прийнятий програмістські світі структурована мова запитів SQL (Structured Query Language).

Для розробки веб-додатків використовується також широко прийнятий у всьому світі мову високого рівня PHP. Мова PHP працює на стороні сервера і забезпечує створення динамічних веб-додатків, в тому числі і взаємодія з базою даних MySQL, за допомогою безлічі функцій і класів. Програми написані, зокрема, на PHP називаються сценаріями.

Коротко можна описати типову спрощену схему взаємодії цих продуктів наступною схемою:

- а) [Браузер] : [Веб-сервер] : [PHP] : [Сервер MySQL];
- б) [Браузер] : [Веб-сервер] : [PHP] : [Сервер MySQL];
- в) Браузер клієнта відправляє HTTP запит на завантаження веб-сторінок веб-сервера. Веб-сервер приймає запит, отримує файл і передає його PHP, який сканує і виконує код PHP, вбудований або динамічно завантажується в веб-сторінку. Якщо в PHP коді присутні функції звернення до бази даних, то виконуються відповідні операції по відкриттю з'єднання з сервером MySQL. Сервер MySQL приймає запит, обробляє його.
- г) Сервер MySQL відправляє результати PHP, який завершує виконання сценарію, формуючи HTML код сторінки. Після цього результати передаються в HTML форматі Веб-серверу. Сервер пересилає HTML файл браузеру, які відображає сторінку на екрані.

В даному проекті використовуються модифікований під додток клас PHP для роботи з MySQL.

2.2.1.6 Використання програми phpMyAdmin для управління базою даних

Для створення бази даних компанії control_clients_mid був використаний пакет phpMyAdmin, що забезпечує веб-інтерфейс для створення і адміністрування бази даних веб-додатків. Цей пакет надає можливість візуального створення бази даних, що становлять таблиць і генерацію SQL запитів до бази даних.

Для цього додатка був створений спеціальний аккаунт, що забезпечує доступ до бази даних тільки цього додатка і обмеження доступу до інших баз даних, що знаходяться під управлінням сервера (малюнку 2.2.1.6.1). Для входу необхідно перейти за посиланням і ввести наступні дані: логін = zao712, пароль = zao712, <http://centr.nuft.edu.ua.com/phpmyadmin/>

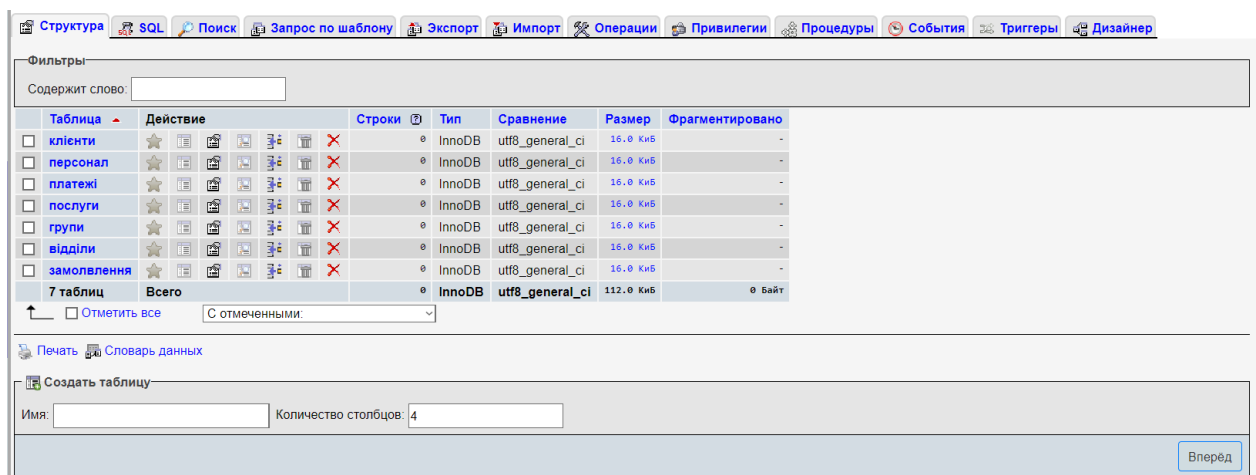


Рис. 2.3. DB структура в phpMyAdmin

Структура таблиці може змінюватися, але потрібно вносити незначні зміни в таблиці подання веб-застосунку.

З програми PHP, відкриття підключення до бази даних забезпечується наступним кодом:

```
<? Php  
define («DB_HOST», «localhost»); // адреса сервера баз даних #  
define («DB_USER», «zao712»); // Ім'я користувача #  
define («DB_PWD», «zao712»); // пароль до бд #
```

```
define («DB_NAME», «control_clients_mid»); // ім'я бази даних #
define («DB_MODE», 2); // 0 - в ручну відкр / закр, #
```

2.2.1.7 Фізична структура бази даних

Для фізичної реалізації логічної структури бази даних використовувався інтерактивний пакет phpMyAdmin. Щоб увійти в адміністративну панель управління базою даних control_clients_mid, необхідно в адресному рядку браузера набрати адресу використовуваного примірника пакета на сервері: <http://www.diofant.com/phpMyAdmin>, а в що з'явилася логін-формі набрати логін - za0712 і пароль - za0712.

Дається назва полів (колонок) в СУБД MySQL, опис полів, типи даних і коротка характеристика і призначення, відповідність табличному поданням полів, викладеному в попередніх розділах.

Ім'я бази дані - це унікальне ім'я, по якому додатки звертаються до бази даних або адміністратор бази даних звертається до бази даних через інтерактивні пакети управління даними типу phpMyAdmin, HS (HeidiSQL) та ін. З метою зміни або доповнення таблиць і полів бази даних. Далі буде описано, яким чином можна змінити структуру таблиць.

На рисунках 2.4-2 представлені найменування таблиць бази даних і їх полів.

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию
☐	1	idКлієнта 	int(11)			Нет	Нет
☐	2	Номер	smallint(10)			Да	NULL
☐	3	Адреса	varchar(45)	utf8_general_ci		Да	NULL
☐	4	Контактна інформація	varchar(45)	utf8_general_ci		Да	NULL
☐	5	Телефон	varchar(15)	utf8_general_ci		Да	NULL
☐	6	E-mail	varchar(30)	utf8_general_ci		Да	NULL
☐	7	Нотатки	text	utf8_general_ci		Да	NULL
☐	8	Групи_idГрупи	int(11)			Нет	Нет

Рис. 2.4. Структура таблиці для клієнтів

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию
☐	1	idВідділ 🗑️	int(11)			Нет	Нет
☐	2	Назва	varchar(45)	utf8_general_ci		Да	NULL
☐	3	Місце розташування	varchar(45)	utf8_general_ci		Да	NULL
☐	4	Телефон	varchar(15)	utf8_general_ci		Да	NULL
☐	5	Головний виконуючий	varchar(30)	utf8_general_ci		Да	NULL
☐	6	Нотатки	text	utf8_general_ci		Да	NULL
☐	7	Замовлення_idЗамовлення	int(11)			Нет	Нет

Рис. 2.5. Структура таблиці для відділів

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию
☐	1	idГрупи 🗑️	int(11)			Нет	Нет
☐	2	Ім'я	varchar(30)	utf8_general_ci		Да	NULL
☐	3	Локація	varchar(45)	utf8_general_ci		Да	NULL
☐	4	Телефон	varchar(15)	utf8_general_ci		Да	NULL
☐	5	Менеджер	varchar(30)	utf8_general_ci		Да	NULL
☐	6	Нотатки	text	utf8_general_ci		Да	NULL

Рис. 2.6. Структура таблиці для груп

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию
☐	1	idЗамовлення 🗑️	int(11)			Нет	Нет
☐	2	Клієнт_id	int(11)			Да	NULL
☐	3	Дата подачі замовлення	date			Да	NULL
☐	4	Дата початку	date			Да	NULL
☐	5	Дата завершення	date			Да	NULL
☐	6	Кодекс виконавчого відділу	varchar(15)	utf8_general_ci		Да	NULL
☐	7	Послуги_id	int(11)			Да	NULL
☐	8	Ім'я виконавця	varchar(30)	utf8_general_ci		Да	NULL
☐	9	Коментарь	text	utf8_general_ci		Да	NULL
☐	10	Клієнти_idКлієнта	int(11)			Нет	Нет

Рис. 2.7. Структура таблиці для замовлень

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию
☐	1	idПлатежа	int(11)			Нет	Нет
☐	2	Дата	date			Да	NULL
☐	3	Клієнт_id	int(11)			Да	NULL
☐	4	Замовлення_id	int(11)			Да	NULL
☐	5	Сума платежа	int(11)			Да	NULL

Рис. 2.8. Структура таблиці для платежів

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию
☐	1	idПослуги	int(11)			Нет	Нет
☐	2	І'мя	varchar(45)	utf8_general_ci		Да	NULL
☐	3	Деталі	varchar(45)	utf8_general_ci		Да	NULL
☐	4	Вартість	varchar(45)	utf8_general_ci		Да	NULL
☐	5	Замовлення_idЗамовлення	int(11)			Нет	Нет

Рис. 2.9. Структура таблиці для послуг

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию
☐	1	idПерсоналу	int(11)			Нет	Нет
☐	2	Код співробітника	varchar(45)	utf8_general_ci		Да	NULL
☐	3	Позиція	varchar(45)	utf8_general_ci		Да	NULL
☐	4	Сертифікат	varchar(45)	utf8_general_ci		Да	NULL
☐	5	Нотатки	varchar(45)	utf8_general_ci		Да	NULL
☐	6	Замовлення_idЗамовлення	int(11)			Нет	Нет

Рис. 2.10. Структура таблиці для персоналу

Щоб змінити структуру таблиць, потрібно зайти на панель управління PhpMyAdmin на <http://www.centruft.edu.ua.com/PhpMyAdmin>, а в з'явилася форма входу для введення логіна -zao712 і Password-zao712.

Якщо змінюється найменування таблиць, додаються або знищуються поля, то після всіх змін, необхідно провести відповідні зміни в заголовках таблиць відображення, тобто змінити відповідним чином, додати або видалити найменування «шапок» таблиць російською мовою. Заголовки уявлень таблиць знаходяться в папці ... / Control_Clients_Mid в файлах

настройки з іменами Table_Headers.php - файл уявлень з російськими заголовками, шапками таблиць і Table_Headers_DB.php - файл імен полів (колонок) в базі даних MySQL. Ці файли, організовані у вигляді асоціативних масивів, використовуються програмним забезпеченням цього додатка, як керуючі мета-таблиці, для управління даними та подання (відображення) таблиць бази даних на екрані дисплея. Крім того необхідно змінити найменування полів в запитах до БД і зокрема, що стосуються внесених змін. У зв'язку з цим передбачена локалізація всіх SQL запитів в певних файлах цього додатка.

2.2.2 Створення інтерфейсу та реалізація функцій системи

2.2.2.1 Групи користувачів і розподілений доступ

Очікується, що застосунок використовуватиметься кількома групами користувачів і є розподілений доступ до цих груп. Ці групи є як керівництвом центру, так і редакторами центру, які в свою чергу можуть також бути поділені на групи, в залежності від рівня доступу до ресурсів і даних системи. Оскільки для розподіленого доступу потрібна система реєстрації та авторизації, цей компонент програмного забезпечення буде реалізований в системі.

Передбачається, що сторінка веб-додатків буде виглядати так у загальній формі веб-додатків, що створюються (рис. 2.12).

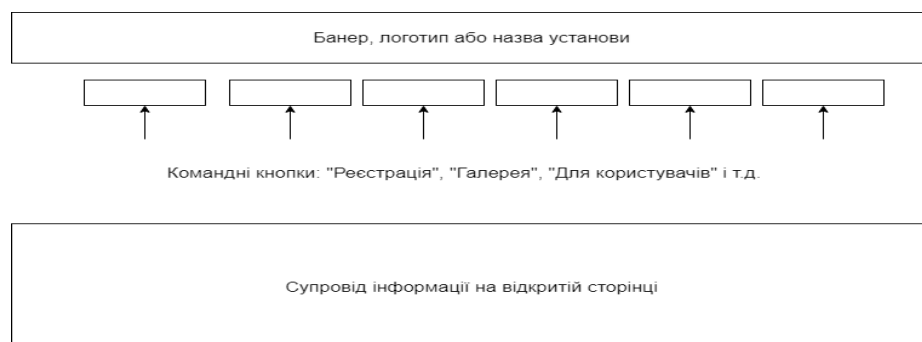


Рис. 2.12. Передбачувані сторінки веб-додатка

На кожній сторінці, яка містить дані з окремої таблиці бази даних, є видалені, оновлені та додані дані. Для цього є кнопки "видалити", "оновлення", "Додати".

Модельна схема процесу подачі заявки (послуга замовлення) для заявника буде проходити в такій послідовності:

- а) відвідувач (потенційний заявник) відвідує сайт організації для перегляду інформації про послуги, перелік можливих досліджень, а також інформацію про ціни (Якщо послуга платна);
- б) якщо відвідувач має намір оформити замовлення-заяву, він зареєстрований, з метою подальшого його авторизації та надання доступу до персонального кабінету і даних, що відповідають його статусу, зокрема, до виданих заявок, форми заповнення платіжних даних. У майбутньому, в особистому кабінеті, він може відслідковувати динаміку застосування;
- в) заявник зазвичай подає заяву в письмовій формі, а потім співробітники центру перевіряє дані і включає в себе заяву в базі даних. В принципі, система, що розробляється, дозволяє заявникові подавати заявку онлайн. У той же час заповнення анкети можна дублювати, відправивши копію на адресу електронної пошти співробітників, який буде обробляти заявку (тому для системи не важливо, хто робить заявку і поміщає його в базу даних-це питання вибору технології проходження додатків, а не можливостей системи);
- г) у подальшому, після затвердження, заповнивши анкету, можна оплатити через платіжні системи Інтернету або готівкою на факт надання заявленого документа.

Таким чином, використовуючи веб-додаток, заявник може скоротити кількість особистих візитів до центру, з метою реалізації його побажань, приймає рішення про вибір того чи іншого варіанту (або декількома

варіантами), подати заявку і контролювати результат її виконання «не залишаючи свого офісу».

2.2.2.2 Використання об'єктно-орієнтованого підходу в додатку

Веб-додаток `Control_Clients_Mid` створений на PHP, з використанням об'єктно-орієнтованого програмування(OOP).php код проекту.

Об'єктно-орієнтоване програмування — сучасне програмування сучасних технологій, напрямок, який неможливо повністю описати навіть в межах однієї книги. Слід зазначити, що розвиток сучасних застосувань практично неможливий без ООП.

Такий підхід забезпечує структуру програмування, дозволяє модифікувати і доповнювати функції, виконані без значних змін до існуючого коду, і просто доповнивши вже розроблені класи додатковими методами і властивостями.

Клас є синтаксичним дизайном коду дорогої і функція, що використовується в програмах за певними угодами, щоб скоротити час розробки і побудувати структурований, легко змінний код. Клас є зразком об'єктів певного типу, які можуть бути використані і використані в програмах різними об'єктами цього типу. Після того, як клас, ви можете використовувати багато додатків, не змінюючи внутрішню структуру класу, але тільки, якщо це необхідно, додаючи нові функції і змінні. Клас створюється так, що програміст не розуміє внутрішню структуру коду, а приймає і використовує готовий код відповідно до домовленостей, викладених в описі занять.

2.2.2.3 Реалізація функцій пошуку

Для реалізації пошукових функцій використовується клас `DB_Class` роботи з базою даних MySQL і стандартна мова запитів SQL. Програмне забезпечення влаштовано таким чином, щоб передбачити можливість зміни структури полів БД. Для цього, по-перше, використовуються файли

настройки Table_Heades.php і Table_Headers_DB.php, а також локалізація пошукових запитів. Так як структура таблиць БД, уявлень таблиць може певною мірою змінюватися при розвитку функцій програми, то такий підхід забезпечить гнучкість, надійність, можливість швидкої адаптації під нові функції або при модифікації пошукових запитів. При зміні найменування, полів таблиць БД, структури таблиць, найменувань полів (шапок полів) табличних подань необхідно відповідним чином змінити файли настройки і найменування полів, атрибути пошуку в пошукових запитах.

Для виконання пошуку по декількох ключах для кожної таблиці розроблені найбільш раціональні, що відповідають реальним потребам вибірки необхідної інформації, форми. Розглянемо, наприклад, форму пошуку в таблиці «Співробітники».

Для того щоб перейти до пошуку по декількох ключах, необхідно в нижній частині сторінки, що представляє форму в табличному вигляді натиснути по посиланню (Рис. 2.13).

#	Статус	Найменування організації	ПІБ замовника	Статус замовника	Назва НДР

Рис. 2.13. Форма пошуку на сайті

Щоб знайти працівників, які відповідають певним умовам пошуку, ви можете задати ідентифікатор працівника на посаді, вибрати працівників, які зараз працюють, і тих, хто працював раніше.

Якщо, наприклад, тільки посада встановлена, система буде побудувати таблицю презентацій з усіма "Інспектор" посад всіх співробітників в даний час працює у всіх підрозділах.

Якщо ж буде додано значення поля Id Підрозділи, то відповідно буде виведена таблиця всіх посад в цьому відділі.

Програма влаштована таким чином, що пошук може здійснюватися не тільки по повному значенню атрибута в шпальтах, а й за вказаною під рядку в стовпці.

2.2.2.4 Структура модуля програмного забезпечення та використання властивостей мови PHP

Розробка програмних модулів не була проведена з нуля, а необхідні програмні модулі були значно змінені, класи PHP взяті з мережі.

DB_Class клас використовувався для підключення бази даних, впровадження базових запитів до бази даних.

Іншим важливим основним компонентом програмного забезпечення є змінений клас таблиці, змінений та доповнене новими методами та функціями. Цей клас використовувався як базовий для створення подань усіх таблиць в одній формі незалежно від вмісту таблиць.

Головна сторінка системи знаходиться за адресою: http://www.centruft.edu.ua.com/Control_Clients_Mid/index.php.

Конструктивно ця веб-сторінка складається з низки спеціально побудованих таблиць в термінології мови HTML, розділивши екран на окремі поля і лінійні, у вигляді стрічки, меню з можливими варіантами роботи з системою.

Дана версія веб-сторінки називається системою «панель керування». Кожна кнопка в цьому меню відповідає посиланню на конкретний модуль програми. Натиснувши на кнопку, користувач ініціює виконання конкретного модуля, кожен модуль є кілька взаємопов'язаних PHP модулів, які виконуються на сервері, і результат у вигляді HTML, CSS, JavaScript веб-сторінка відправляється в браузері.

Структура програмного забезпечення певною мірою відображає технологію MVC(модель-вид-контролер)і складається з трьох рівнів (Рис. 2.14). У схемі представлені не всі модулі, частина з них описується нижче. Як

вже було сказано раніше, базові класи були розроблені в компанії MitriDAT і в процесі роботи над проектом адаптувалися під вимоги проекту.

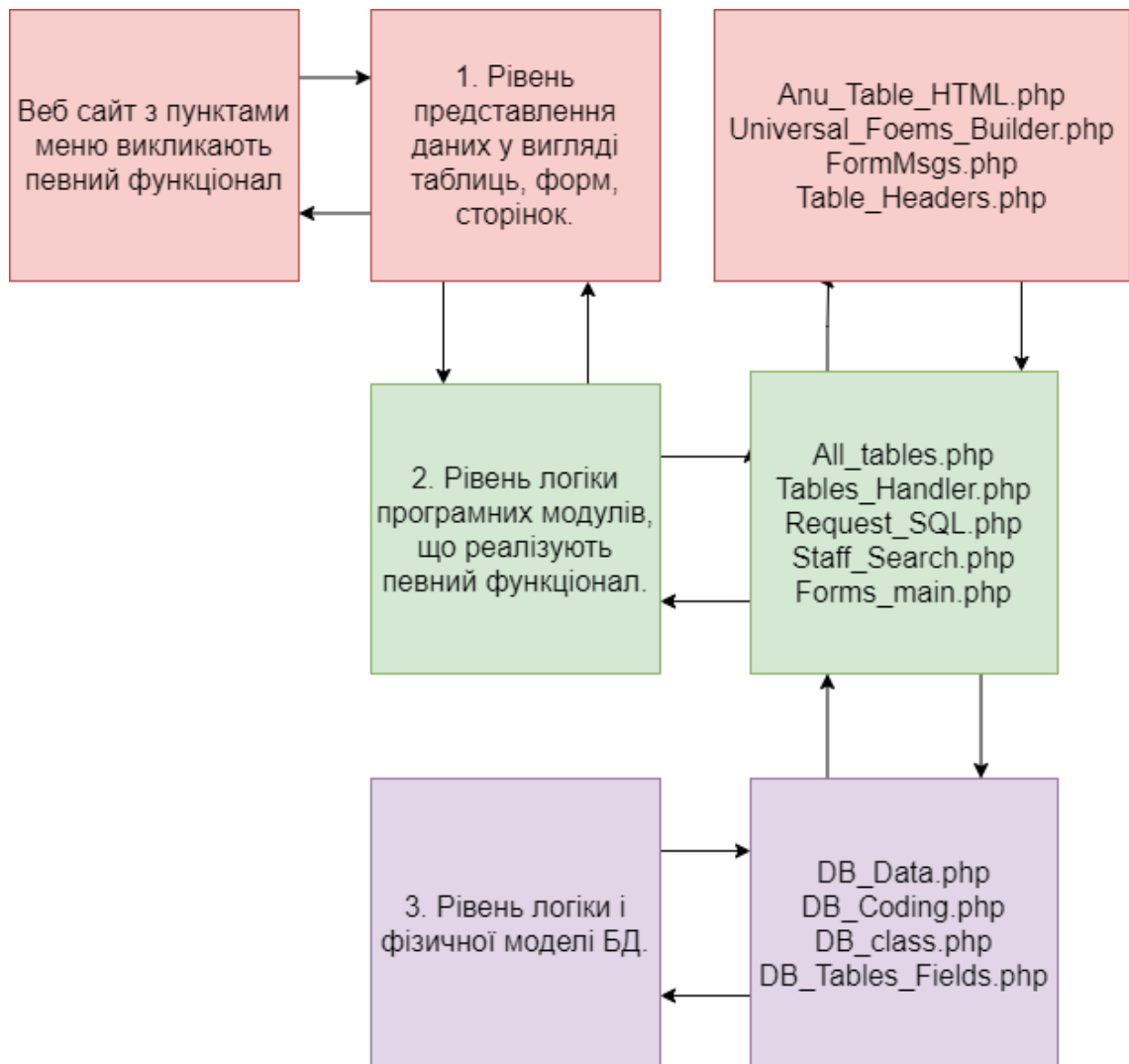


Рис. 2.14. Структура програмного забезпечення веб-застосунків

Кожен рівень відповідає програмних модулів, представлених в крайньому правому стовпчику діаграми. Звичайно, не всі модулі другого рівня звертаються до бази даних і не всі модулі представлені на цій схемі. Опис всіх можливих послідовностей модулів штабів займе досить великий обсяг дипломного проекту, тому давайте подивимося, як програмне забезпечення працює відповідно до цієї схеми за прикладом.

При натисканні кнопки верхне-меню «Веб-Схема БД», викликається модуль `All_Tables.php`. Цей модуль звертається до нижчого рівня, викликаючи модуль `DB_Tables_Fields.php`, який сканує поточну логіку бази даних («на льоту», саме тому можна міняти структуру і зміст полів сторінок,

не замислюючись про модифікацію коду), зчитує таблиці і поля. Потім модуль All_Tables.php формує запит до бази даних, звертається до модуля DB_Class.php, який оформлений як клас, отримує значення полів бази даних (рядків). Після цього за допомогою модулів табличного представлення вищого рівня Any_Table_HTML.php формується HTML-код таблиць, додається заголовок («шапка») Header.php сторінки, яка містить логотип, HTML-CSS код сторінки і повністю сформована сторінка відправляється в браузер користувача у відповідь на запит . Нагадаємо, що весь програмний код працює на стороні сервера, який і відправляє вже готову для відображення HTML - сторінку. Аналогічно відбувається формування табличних даних для будь-якої обраної таблиці, але при цьому працює інший модуль - Tables_Handler.php. Короткий опис роботи кожного модуля буде дано нижче.

2.2.2.5 Програмна композиція та модульна функція

Нижче представлена композиція програмного забезпечення та функцій, виконаних модулями.

Control_Clients_Mid	Головна папка, що містить всі модулі
All_Tables_short.php	Головна сторінка з таблицями БД
Any_Table_HTML_New.php	Клас таблиць відображення
DB_Class.php	Клас роботи з базою даних
DB_Coding.php	Модуль перекодування в кирилицю
DB_data.php	Параметри з'єднання з БД
Depart_Handler.php	Оброблювач «Підрозділи»
Device_Handler.php	Оброблювач уявлення
Form_Buttons.php	Форми видалення, пошуку, оновлення
Form_Buttons_Staff.php	Форма пошуку по декількох ключах
FormMsgs.php	Повідомлення
Forms_Main.php	Форма додавання даних в усі таблиці
Header.php	Стандартна шапка сторінок додатку

Prod_Handler.php	Оброблювач БД та подання
Staff_Handler.php	Оброблювач БД «Співробітники»
Staff_Search.php	Модуль пошуку по декількох ключах
Table_Headers.php	Файли настройки «шапок» таблиць
Table_Headers_DB.php	Файли настройки полів таблиць БД
Universal_Forms_Builder.php	Будівник форм додавання в таблиці.

Кожна таблиця має свій власний обробник, який використовує базові класи DB_Class.php, DB_Coding.php. Модуль перекодування кирилицею є стандартом, що використовується для всіх додатків, які використовують кирилицю в базах даних.

Додавання даних конструктор форм у Universal_Forms_Builder.php є універсальним модулем для всіх таблиць для створення фігур цього типу, незалежно від змісту таблиці.

Вище, є тільки короткий опис модулів, які складають програмне забезпечення веб-додатків. Структура програмного забезпечення, вміст папок і файлів представлені на рисунку 3002E4. Вся програма знаходиться в папці Control_Clients_Mid.

Додатком до дипломного проекту є zip файл, в якому представлений весь код справжнього програмного забезпечення, який включає HTML сторінки веб-сайту, SQL код генерації бази даних, SQL-PHP код управління базою даних і всього веб-додатки в цілому.

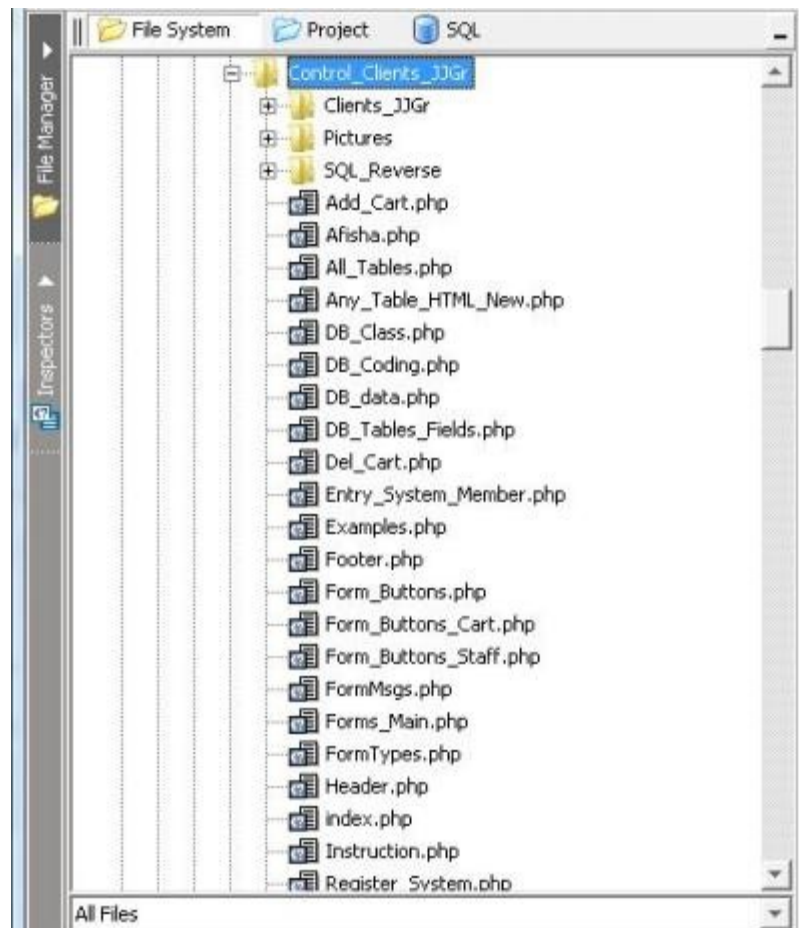


Рис. 2.15. Структура програмного забезпечення додатка в поданні PHPStorm

2.2.2.6 Розробка функціональної моделі

Аналіз і опис деяких бізнес-процесів центру. Секція 1.2 містить загальний опис робочих процесів кафедри.

Структурний метод проектування. Два типи моделей: діаграми IDEF0 та діаграми потоків даних (DFD, схеми обробки даних) зазвичай використовуються в задачах моделювання бізнес-процесів.

Діаграма потоків даних показує як етапи бізнес-процесу, так і потік документів і керування.

Діаграми даних DFD:

- а) Це засіб моделювання функціональних вимог до проектованої системи;
- б) Призначений для імітації існуючого руху інформації;
- в) Подавати документи та обробку інформації;

г) Використовується як додаткові моделі для моделі IDEFO для візуалізації поточного документообігу та обміну інформацією.

DFD діаграм доповнюють те, що вже ПРЕДСТАВЛЕНІ в моделі IDEFO, тому що вони описують потоки даних, що відстежує рух і перетворення інформації в системі, між процесами, і між процесами з зовнішнім інформаційним середовищем.

Діаграми DFD створено як інструмент для проектування програмних систем.

З діаграмами DFD вимоги до проектованої інформаційної системи розбиваються на функціональні компоненти. Функції DFD демонструють, як кожен процес перетворює свій внесок у вихідні дані. На схемах процесу відображаються:

- а) Функція процесу;
- б) Вхідна та вихідна інформація;
- в) Зовнішні процеси, описані в інших діаграмах;
- г) точка розбиття при переміщенні на інші сторінки.

При моделюється за методикою IDEFO , система вважається мережею взаємопов'язаних функцій. Під час створення діаграми DFD система представлена як набір пов'язаних функцій, як набір сутностей (об'єктів).

З вищесказаного ми можемо зробити висновки: при моделюванні системи "як є" найкращим використанням специфікації SADT, як вона традиційно використовується для моделювання процесів на підприємстві;

При будівництві "як вона повинна" модель системи, краще використовувати DFD специфікацій. Ця методологія дає більше переваг при проектуванні інформаційної системи.

2.3 Інструкція користувача

2.3.1 Технологічна послідовність роботи з системою

Є три основні етапи, три технологічні послідовності роботи:

а) Встановлення підсистеми аутентифікації користувачів; створення таблиць бази даних.

б) Реєстрація користувачів, в першу чергу претендентів. Заповнення бази даних.

в) Після цього система готова до регулярної роботи.

Підсистема автентифікації призначена для реєстрації користувачів, організації розподіленого доступу до захищених папок, компонентів системи та системних баз даних. Установка системи дуже проста. По суті, це сама установка, в якій підсистема сама створює всі керуючі файли. Установка тільки включає в себе оригінальний логін і пароль системного адміністратора, так що він може увійти в панель управління підсистеми аутентифікації. Надалі адміністратор може додати інших адміністраторів, які мають однакові права. У файлах підсистеми зберігатися інформація про логіни і паролі користувачів, захищених папках, дані про реєстрацію користувачів.

Крім адміністратора передбачається дві групи реєструються користувачів системи - заявників та персоналу центру. У цій системі прототипів передбачається, що співробітники мають доступ до всіх функцій і даних системи, хоча цей постулат може бути змінений в залежності від фактичної функції, що виконуються кожним працівником. Щоб зареєструвати користувачів: По-перше, вам потрібно увійти. В даний час вам потрібно набрати наступну адресу в адресному рядку браузера для входу: http://www.centru.nuft.edu.ua.com/control_clients_mid/index.php

Перша сторінка системи з'явиться (давайте поговоримо про веб-додатків або просто додатків). Ви повинні натиснути на пункт реєстрації меню. Там буде сторінка поруч з реєстрацією, яка надає посилання на початок реєстрації. Реєстрація передбачає створення логіна і пароля для входу та введення деяких особистих даних. Реєстрація займе якийсь час. Після того, як реєстраційні форми були завершені, дані будуть автоматично відправлені на електронну адресу системного адміністратора, який

ідентифікує дані і підтверджує право реєстрації в системі. Після підтвердження реєстрації, повідомлення, яке підтверджує логін і пароль, буде надіслано до вказаної форми. Система прийняла угоду про те, що логін може бути, наприклад ANS або номер паспорта і т. д.

Розглянемо форми і процедури реєстрації претендентів. Для співробітників, вся процедура аналогічна. Деякі відмінності існують тільки у формі представлення персональних даних. Є дві форми для реєстрації. Перша форма призначена для реєстрації логіна і пароля користувача, а також включає в себе поле для підтвердження обраного паролю і адреси електронної пошти.

Після успішного проходження цього кроку при заповненні форми і натисканні на «наступний» ключ в нижній частині форми, відбувається перехід до наступної форми реєстрації, деякі персональні дані про користувача.

За персональних даних адміністратор системи, ідентифікує користувача і вирішує, підтвердити чи ні реєстрацію користувача, іншими словами, «за кого він себе видає», заявник це або співробітник.

Після успішного проходження другого етапу заповнення форми і натискання кнопки «Реєстрація» в нижній частині форми, відбувається перехід до наступної форми-повідомлення про те, що дані введено правильно і реєстраційний запит відправлений адміністратору системи.

Дані користувача автоматично надсилаються до системного адміністратора. Системний адміністратор отримує повідомлення на його поштову адресу у формі, зображуваного на рисунку 2.16.

```
Привет Admin! Пользователь Иван Иванович Иванов просит зарегистрировать его на сайте.  
  
Он успешно прошел регистрацию.  
После проверки данных пользователя,  
кликните по ссылке для подтверждения регистрации.  
Пользователь получит доступ к защищенной области Clients_JJGr:  
  
http://www.diofant.com/aladin-m/au\_register\_auto\_mem.php?passwd=IvanovJJGr  
  
данные для входа пользователя:  
Логин      = IvanovJJGr;  
Пароль     = IvanovJJGr  
  
Имя Отчество = Иван Иванович;  
Фамилия      = Иванов;  
Место Работы = ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, ЭКОНОМИКИ И МЕНЕДЖМЕНТА;  
Статус       = заявитель;  
Отдел/Группа = ОТК;  
Улица        = Ивановская, 13;  
Город        = Москва;  
Область      = Московская область;  
Страна       = Россия;  
Телефон=999999999  
  
Система он-лайн регистрации
```

Рис. 2.16. Повідомлення системному адміністратору про реєстрацію нового користувача

Системний адміністратор отримує повідомлення на його поштову адресу, ідентифікує користувача. Опція ідентифікації керівництва тепер реалізована.

Якщо ви є заявником і намагаєтеся зареєструватися як працівник, системний адміністратор не дозволить вам і ви не будете зареєстровані в системі. У подальшому можна автоматично визнати і підтвердити реєстрацію.

Для підтвердження реєстрації користувача, адміністратор просто натискає на посилання, включене в повідомлення. Потім дані вводяться в підсистему аутентифікації і на адресу електронної пошти користувача, які він вказав в першій формі реєстрації, повідомлення про його успішну реєстрацію, а також зареєстрований логін і пароль (той, який він вказав в першій формі реєстрації, цифра) (2.17).

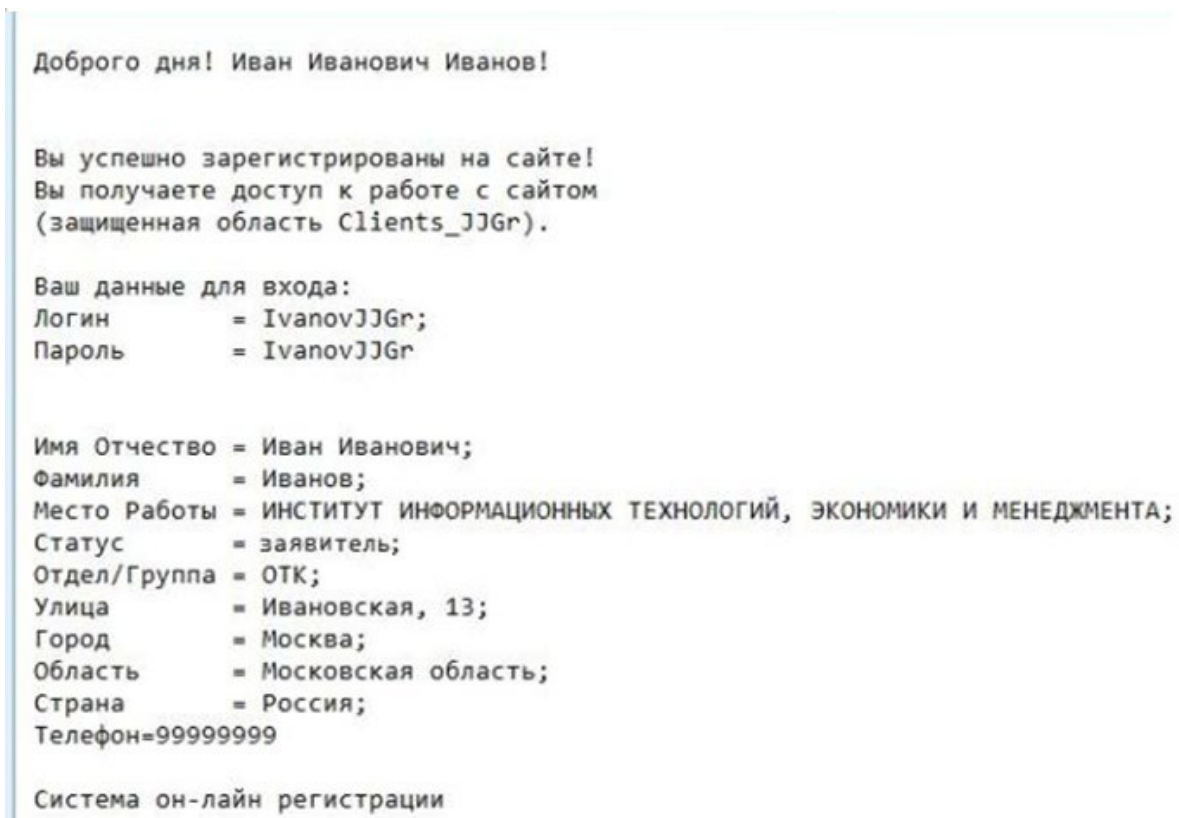


Рис.

2.17. Повідомлення про реєстрації на адрес користувача

У цьому прикладі довільний логін використовувався для демонстрації процедури реєстрації та участі у формах, але відповідно до програмних угод, необхідно ввести номер паспорта або інший логін. Після реєстрації в системі, працівник повинен ввести дані заявника в системну базу даних. Також можна сполучити підсистему реєстрації з базою даних системи. Потім дані будуть введені автоматично.

Як тільки заявники і співробітники зареєстровані, ви можете почати працювати з системою. Вхід до системи для користувача і співробітників здійснюється через кнопки працівника і користувача в меню (в даній версії тільки для заявників). Після натискання цих кнопок запрошення буде здійснено для входу до зареєстрованих користувачів із посиланням для входу та отримання доступу. Після натискання посилання ви отримуєте форму, щоб авторизувати користувача за допомогою полів логіна і пароля.

Як тільки клієнти та працівники уповноважені, ви можете почати працювати з системою на повний робочий підхід.

Розроблена інформаційна система дозволить здійснювати пошук інформації за критеріями, отримувати звіти про виконання робіт, надавати інформацію про співробітників, замовлення, виконані роботи.

Організація екранних форм. Розроблюване програмне забезпечення передбачається для роботи в операційній системі Windows, в зв'язку з цим призначений для користувача інтерфейс повинен задовольняти стандартам прийнятим фірмою «Microsoft».

Програма повинна мати інтуїтивно зрозумілий користувацький інтерфейс, що робить програму легкою для роботи.

Організація випуску друкованих документів. Основним видом вихідного друкованого документа будуть звіти про виконані роботи, подані та оброблені додатки, сформовані на основі обраних користувачем параметрів у пресі.

Вимоги для захисту від неавторизованих дій. Вступ до програми повинен передуватися аутентифікацією користувача за допомогою логіна і пароля, після введення, який визначається профілем доступу користувача, що окреслюючи доступ до певних пунктів меню і функцій програми.

Вимоги до безпеки інформації. Для того, щоб зберегти інформацію, необхідно передбачити можливість періодичного резервного копіювання бази даних. Частота резервних копій щоденна або щотижнева, в залежності від швидкості і обсягу інформації введення.

2.3.2 Виконання функцій адміністрації бази даних

Базу даних можна вводити двома способами: використовуючи інтерактивний пакет PhpMyAdmin і безпосередньо через інтерфейс цієї програми за допомогою перегляду таблиць і вхідних форм.

Адміністрація бази даних у цьому додатку посилається на повний набір інструментів, реалізованих у вигляді програмних модулів, які забезпечують керування даними (маніпуляції): додавання нових даних до таблиць, видалення даних, оновлення даних.

Дані управляються через браузер через веб-інтерфейс-спеціально розроблені види таблиць, форм, послідовностей команд.

Робота адміністратора починається з головної сторінки, яка відображає всі таблиці бази даних у формі "усіченого". Натискаючи посилання на потрібну таблицю, ви можете виконувати всі доступні транзакції з даними таблиці.

Щоб додати дані до таблиць, була розроблена Уніфікована форма, яка базується на універсальному коді РНР, але заповнюється назвами полів, що відповідають типу таблиць. Для того, щоб додати дані в потрібну таблицю, вам потрібно натиснути на посилання на головній сторінці зі списком всіх таблиць для доступу до цієї таблиці, наприклад в таблиці, в зверху справа є кнопка створення ного запису (Рис. 2.18).

Центер

ДомашняЗамовленняЗвітиКалендарьКористувачіРегістраціяВихід (admin)

Головна / Замовлення / Створити замовлення

Створити замовлення

Найменування організації *

Не вибрано

ПІБ замовника *

Статус замовника *

Керівник теми

Назва НДР

Роки виконання НДР

Номер держреєстрації НДР

Характер роботи *

Контактні дані особи зі сторони замовника

Найменування та мета роботи

Опис проведення досліджень

Необхідність повернення зразків

Необхідне обладнання

немає

Дата початку досліджень

Select operating time ...

Дата закінчення досліджень

Select operating time ...

Додаткова інформація

Дата подання заявки *

Select operating time ...

Зберегти

© Центер 2020

Рис. 2.18. Форма для додавання даних в таблиці

Поля розташовуються в послідовності-видаляти, знаходити, оновлювати. Для цього вам потрібно буде ввести ідентифікаційний код відповідної лінії (працівника в нашому прикладі). Якщо вводиться Id і натискається кнопка «Оновити», то з'являється форма для поновлення з усіма даними підрозділу (Рис. 2.19). Теж для інших таблиць, наприклад, «Працівників».

Центер

Домашня
Замовлення
Звіт
Календарь
Користувачі
Регістрація
Вихід (admin)

Головна / Замовлення / 20

20

Оновити
Видалити

ID	20
Найменування організації	Одеська національна академія харчових технологій
ПІБ замовника	Пономарчук Максим Васильович
Статус замовника	Старший науковий робітник
Назва НДР	Інноваційний менеджмент
Роки виконання НДР	2020-06-16
Номер держреєстрації НДР	65159
Характер роботи	Прикладна
Контактні дані особи зі сторони замовника	380(451)556-65-17
Найменування та мета роботи	
Опис проведення досліджень	
Необхідність повернення зразків	1
Необхідне обладнання	Mastersizer 3000E
Дата початку досліджень	2020-06-22
Дата закінчення досліджень	2020-05-27
Додаткова інформація	
Дата подання заявки	2020-06-26
Статус	Відхилений

© Центер 2020

Рис. 2.19. Форма для поновлення з усіма даними підрозділу

Після додавання даних і натискання кнопки «оновити», дані будуть занесені в таблицю, з'явиться список всіх співробітників з супутніми полями меню. Тут можна проконтролювати правильність введених даних.

Таким же чином відбувається видалення запису про підрозділ, якщо це необхідно. Аналогічні операції доступні для всіх таблиць бази даних, незалежно від змісту таблиць.

2.4 Розрахунок ефективності витрат

У цьому розділі обчислюється економічний ефект і рентабельність капітальних вкладень у розвиток інформаційної системи.

У ринковій економіці, коли підприємства конкурують, перший пріоритет полягає в обладнанні підприємств, компаній, корпорацій з сучасним високопродуктивним обладнанням і засобами автоматизації, які повинні забезпечувати високу продуктивність і якість продукції, покращувати маркетингові можливості організації, збільшувати обсяги

продажів, кількість і якість послуг. Найважливішим способом збільшення прибутків є створення та ефективне використання нових високоякісних технологій, прогресивних технологій, впровадження організаційних, технічних та проектних рішень. Найвищий економічний вплив за найнижчою вартістю може бути досягнуто, якщо розрахунок, комплексний аналіз, обґрунтування та оцінка ефективності нових технологій (в даному випадку база даних) розраховується.

Ефективність ІС це ступінь її відповідності своєму призначенню. Прийнято розрізняти функціональну і економічну ефективність.

Критерій ефективності - не числом, як показник, а тенденція (наприклад, звести до мінімуму витрати це критерій).

Найчастіше оцінка економічної ефективності заснована на зіставленні витрат і результатів. Для оцінки економічної ефективності проекту застосовуються статичні і динамічні показники. Статичні показники не враховують різний час витрат на ІС та результатів від її функціонування і використовуються при невеликих термінах створення системи (до року).

Динамічні показники враховують розподіл витрат і результатів з плином часу, виходячи з поняття вартості часу грошей. Їх використання корисне протягом довгого життя системи.

Можна виділити три основні групи методів, що дозволяють визначити ефект від впровадження ІС:

- а) фінансовий (кількісний);
- б) якісне;
- в) ймовірність.

Кожен з них має свої плюси і мінуси, тому для того, щоб повністю проілюструвати остаточний ефект від впровадження ІС, крім фінансових методів, необхідно використовувати методи нефінансового аналізу. Використання всіх трьох груп методів призводить до оптимальної оцінки реалізації проекту.

В якості основного статичного показника економічної ефективності може бути використаний річний економічний ефект (економічний прибуток) (2.4.1):

$$\Xi = \Delta \Xi_{\text{рік}} - C - E * K = \Delta \Xi_{\text{рік}} - \Pi \quad (2.4.1)$$

де $E_{\text{год}}$ - річна економія (прибуток), без урахування експлуатаційних витрат;

C - грн;

K - одноразові витрати (капіталовкладення);

E - норма прибутку на капітал (нормативна прибутковість);

Π - річні приведені витрати: $\Pi = C + EK$

Одиниці виміру у формулі (розмірність величин):

E - грн./рік.; $E_{\text{год}}$ - грн./рік.; Z - грн./рік., K - грн.; E - 1/рік .; Π - грн./рік.

$(\Delta E - C)$ - це господарська або бухгалтерський прибуток. Вона являє собою різницю між виручкою і явними витратами. За ринкової термінології, явні витрати (Explicit Cost) - це всі грошові витрати організації, включаючи амортизацію.

З точки зору економічного змісту, величина E складається з норми віддачі на капітал і норми підприємницького доходу. Величина E в ринкових умовах не повинна бути менше річної банківської процентної ставки.

Так, якщо Центральний банк встановив з 1 грудня 2020р ставку рефінансування 13%, норму прибутку на капітал необхідно встановити рівний 13%.

Розглянемо складові витрат формули (1).

Капітальні витрати (K)

Капітальні витрати є одновідними. Ті, які надсилаються до основних засобів обробки інформації передачі їх вартість товарів і послуг в частинах через амортизаційні відрахування. Капіталу їх називають тому, що вони не втрачені, а відтворюються.

Капітальні витрати включають:

- а) витрати на обслуговування (комп'ютерне обладнання, Оргтехніка, комунікації, технічні засоби захисту і т. д.);
- б) витрати на програмне забезпечення, включаючи функціональні та сервісні витрати;
- в) вартість встановлення приміщень, включаючи робочі місця працівника;
- г) витрати на послуги експертів та консультантів тощо.

Будемо вважати, що компанія володіє необхідною обчислювальною технікою і засобами зв'язку. Базове програмне забезпечення серверів, як уже зазначалося, є безкоштовним, код відкритим і вільно поширюваним. Тому з усіх складових капітальних витрат залишимо тільки ті, які перераховані нижче у формулі (2.4.2).

Капітальні видатки можна підсумувати таким чином (2.4.2):

$$K = K_{\text{пр}} + K_{\text{іб}} + K_{\text{уч}} + K_{\text{неврах.}} \quad (2.4.2)$$

Де

$K_{\text{пр}}$ - витрати на проектування ІС і розробку програм;

$K_{\text{іб}}$ - витрати на фахівців беруть участь у формуванні БД і програмного забезпечення;

$K_{\text{уч}}$ - витрати на навчання персоналу;

$K_{\text{неучт}}$ - невраховані витрати, зазвичай складають 8-10% від загальних витрат.

Давайте порахуємо капітальні видатки:

$K_{\text{пр}} = \text{з/пл програмістів} * \text{к-ть програмістів} * \text{термін проектування та програмування} = 20\text{тис.грн} * 1 * 1 \text{ міс} = 20 \text{ тис. грн.}$

$K_{\text{іб}} = \text{з/пл} * \text{к-ть фахівців} * \text{термін роботи. бази} = 20\text{тис. грн} * 1 * 0,5 \text{ міс} = 10 \text{ тис. грн.}$

$K_{\text{уч}} = \text{з/пл інженера, який курирує БД} * \text{термін навчання} = 20 * 0,2 \text{ міс} = 3 \text{ тис. грн.}$

Разом отримуємо: $K = (20 + 10 + 3) + 8\% \text{ неврахованих витрат} = 36 \text{ тис. грн.}$

Загальна сума капітальних витрат представлена на Рис. 4.1.

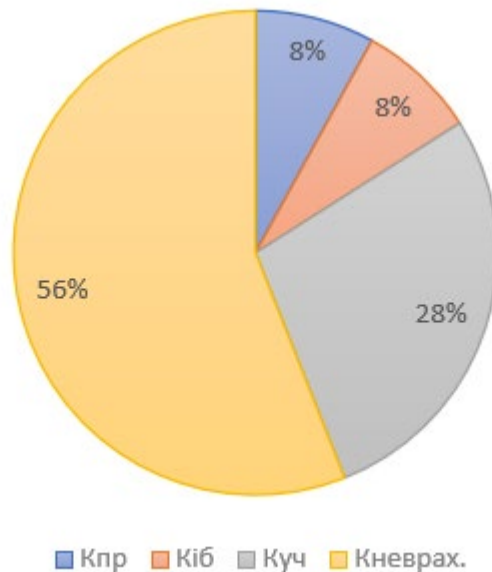


Рис. 2.4.1. Капітальні витрати

Експлуатаційні витрати (С)

Експлуатаційні витрати повторюються. Вони повторюються в кожному циклі доставки послуг, і розраховані в загальній складності за рік. Експлуатаційні витрати здійснюються синхронізуватися з експлуатацією організації. Експлуатаційні витрати-це вартість послуг:

- а) вартість заробітної плати, виплаченої головному та допоміжних персоналу;
- б) вартість операційних обчислень та іншого обладнання;
- в) витрати на експлуатацію приміщень та обслуговування робочих місць і т. д.

Ці витрати включають всі витрати, враховані відповідно до прийнятої процедури собівартості (за виключенням амортизації і амортизації). Крім того, на ранніх стадіях розробки та впровадження нових технологій, при відсутності конкретної (звітної та нормативної) інформації, для розрахунку витрат на послуги можуть застосовуватися розширені методи собівартості,

зокрема метод специфічних показників, методи регресійного аналізу, метод структурної аналогії, агрегування і метод скорингу і т. д.

Експлуатаційні витрати на інформаційну систему включають наступні витрати (2.4.3):

$$C = C_{ав} + C_{то} + C_{ін} + C_{ел} + C_{ін}. \quad (2.4.3)$$

Де

$C_{ав}$ - амортизаційні відрахування;

$C_{то}$ - витрати на технічне обслуговування, включаючи заробітну плату персоналу;

$C_{ін}$ - витрати, пов'язані з використанням глобальних обчислювальних мереж (Інтернету та ін.);

$C_{ел}$ - витрати на електроенергію;

$C_{ін}$ - інші витрати становлять приблизно 8%.

Найбільша частка в операційних витратах належить до заробітних плат, амортизації та амортизації, технічного обслуговування.

Введення в експлуатацію ІС не вплине на витрати, такі як:

- а) амортизаційні відрахування ($C_{ав}$) - їхня вартість не зміниться, так як автопарк обладнання не зміниться;
- б) вартість використання глобальних обчислювальних мереж ($C_{ін}$) - ці витрати будуть постійними, тому при розрахунку їх ми також нижче їх.

Ми розраховуємо інші експлуатаційні витрати.

Витрати на обслуговування, включаючи заробітну плату співробітників:

$$C_{то1} = (\text{з/пл інженера БД} * 12 \text{ міс}) * 3\% = 10 * 12 * 0,03 = 3,6 \text{ тис. грн/рік}$$

$$C_{то2} = (\text{з/пл персоналу} * 12 \text{ міс}) * 25\% = 12 * 12 * 0,25 = 36 \text{ тис. грн/рік}$$

Витрати на електроенергію:

$$C_{ел} = 150 \text{ грн. мес.} * K\text{-ть} * 12 \text{ міс} = 1,8 \text{ тис. грн/рік}$$

$$C = 3,6 + 36 + 1,8 + 8\% = 41,4 + 8\% = 44,7 \text{ тис. грн./рік}$$

Розподіл експлуатаційних витрат показано на рисунку 2.4.2.

Як було відзначено, ΔE год - річна економія (прибуток), викликана введенням ІС, без урахування експлуатаційних витрат на ІС, буде рахуватися за формулою (1):

$$\Pi = C + E * K = 44,7 + 36 * 8,25\% = 47,7 \text{ тис. грн.}$$

Хоча вартість витрат, як правило, не є особливо складними і в значній мірі чисто технічні, економічні показники (особливо непрямі) може бути важко оцінити. У зв'язку з цим, експерт (експерт) метод експертних оцінок часто використовується для оцінки індивідуальних показників, які є частиною загального економічного ефекту, в якому експерти (експерти) розглядають оптимістичний, песимістичний і, швидше за все, значення, а не розрахунку будь-якого з композитних показників.

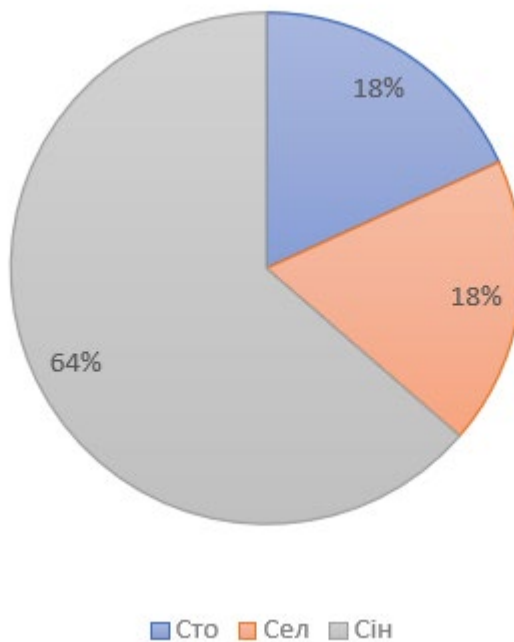


Рис. 2.4.2 - Розподіл експлуатаційних витрат

Методика визначення економічної ефективності на основі статичних показників полягає в розрахунку річного економічного ефекту як суми прямих і непрямих ефектів.

Прямий економічний вплив можна виразити за природними, затратними і трудовими показниками, а також у своїх комбінаціях, при впровадженні нових інформаційних технологій:

- а) підвищити продуктивність працівників державного сектора;
- б) дозволяє розширити спектр послуг;
- в) знижує вартість послуг з продажу.

Іншими словами, прямий економічний ефект є результатом будь-яких змін у природі реалізації функціональної складової процесу роботи, як правило, безпосередньо пов'язаних з особливостями предметної області державної установи. У той же час, ріст продуктивності може бути досягнуто за рахунок скорочення обсягів ручних операцій або більш швидка обробка інформації через обчислювальні засоби.

Непрямий економічний вплив впровадження нових інформаційних технологій є результатом чинників, які, як правило, не мають безпосереднього відношення до специфіки предметної області і, як правило, є соціальними, ергономічними, екологічними та іншими.

Для цілей аналізу та методичної зручності розрахунку річного економічного ефекту доцільно визначити як кількість прямих, так і непрямих ефектів (2.4.4):

$$\Xi = \Xi_{\text{прям.}} + \Xi_{\text{непрям.}} \quad (2.4.4)$$

Розглянемо розрахунок прямих економічних наслідків. Він зводиться до наступного: різниця в щорічних витрат, наданих основними і запропонованими варіантами ІС:

$$\Xi_{\text{прям.}} = P_6 - P = \Delta C_{\text{зп}} - C_{\Sigma} - EK \quad (2.4.5)$$

Де – $\Delta C_{\text{зп}}$ скорочення заробітної плати персоналу державної установи при впровадженні;

C_{Σ} - сумарні експлуатаційні витрати за винятком заробітної плати персоналу державної установи.

$\Delta C_{зп} = C_{зпб} - C_{зп} = 0$, тому що не передбачається знижувати зарплату працівникам або звільняти їх при введенні автоматизації робіт.

Де $C_{зпд}$ - заробітна плата персоналу державної установи в базовому варіанті;

$C_{зп}$ - заробітна плата персоналу державної установи в запропонованому варіанті.

$$\Delta_{\text{прям.}} = \Delta C_{зп} - 47,7 = -47,7 \text{ тис. грн } C_{зп}$$

Величина прямого економічного ефекту є недостатньою (навіть негативною) для виправдання витрат на впровадження. Це пояснюється тим, що ІС створюється з нуля і відсутністю економії на заробітній платі персоналу державної установи.

В цьому випадку впровадження ІС доцільно, тільки якщо є впевненість в досить великому непрямому економічній ефект.

Розглянемо розрахунок непрямого економічного ефекту. Даний розрахунок передбачає визначення наступних складових:

$$\Delta_{\text{непрям.}} = \Delta \Delta_{\text{рік}} = \Delta A + \Delta C_{\text{соб}} + \Delta \text{Ш} + \text{ЕДО} \quad (2.4.6)$$

Де ΔA - річний приріст виручки від реалізації послуг, що надаються, іншої реалізації або поза реалізаційної діяльності;

$\Delta C_{\text{соб}}$ - річна економія на собівартості послуг, що надаються державною установою;

$\Delta \text{Ш}$ - скорочення штрафів й інших не планованих втрат за рік; загальна сума штрафів за рік, викликана помилками в документах з вини співробітників, дорівнює приблизно 50 тис. грн.

ІС дозволить знизити ці втрати на 80% в рік.

Склад статей, за якими розраховується економія на собівартість товарів і послуг за рахунок ІС (2.4.7):

$$\Delta C_{\text{соб}} = \Delta C_e + \Delta C_{зп} + \Delta C_{\text{уео}} \quad (2.4.7)$$

Де ΔC_e - економія на електроенергії на технологічні цілі;

$\Delta C_{зп}$ - економія на заробітній платі працівників;

$\Delta C_{сeo}$ - економія на утриманні та експлуатації обладнання.

У структурі вартості суспільства основну частку займають матеріальні витрати - 38%, а витрати на оплату праці з відрахуваннями - 36%.

Заплануємо 3% скорочення витрат на оплату праці за рахунок скасування деяких функцій і 20% скорочення витрат на обладнання, матеріали. Для простоти розрахунку об'єднаємо економію по енергії, утримання обладнання та витрат і заплануємо 1% економії.

Для проведення розрахунків необхідна собівартість робіт, умовно приймаємо її за 100 тис. грн, для зручності проведення розрахунків.

Отримуємо:

$$\Delta C_{м} = 100 * 38\% - (38\% * 100 - 20\%) = 30,4 \text{ тис. грн/міс.}$$

$$\Delta C_{зп} = 100 * 36\% - (36\% * 100 - 3\%) = 34,92 \text{ тис. грн/міс.}$$

$$\Delta C_{проч} = \Delta C_{сe} + \Delta C_{уeo} = 26\% * 100 (26\% * 100 - 1\%) = 25,74 \text{ тис. грн/міс.}$$

$$\Delta C_{соб} = 12 * (\Delta C_{см} + \Delta C_{зп} + \Delta C_{інш}) = 12 * (30,4 + 34,92 + 25,74) = 91,06 \text{ тис. грн/рік.}$$

Якщо впровадження ІС не впливає на будь-яку статтю витрат у складі собівартості, то ця стаття, очевидно, не фігурує в розрахунку непрямого економічного ефекту.

$$E_{непрям.} = \Delta A + \Delta C_{соб} + \Delta Ш = 0 + 91,06 + 40 = 131,06 \text{ тис. ргрн.}$$

$$\Delta E_{рік} = E_{непрям.} + E_{прям} = 131,06 - 47,7 = 83,36 \text{ тис. грн.}$$

$$E = \Delta E_{рік} - П = 83,36 - 47,7 = 35,66 \text{ тис. грн.}$$

Річний економічний ефект являє собою абсолютний показник ефективності. Система вважається ефективною, якщо $E > 0$.

Допоміжними показниками економічної ефективності є:

- Розрахункова прибутковість (рентабельність):

$$E_p = \Delta E_{\text{річ}} / K = 83,36 / 36 = 2,32$$

- Термін окупності:

$$T_{oK} = \frac{1}{E_p} = \frac{K}{\Delta E_{\text{річ}}} = 0,43 - \text{проект окупиться через 4,3 місяці.}$$

Розробка ІС буде проводитися протягом 1 місяця, в одну зміну тривалістю 8 годин. Заробітна плата розробника за 1 місяць, з урахуванням районного коефіцієнта і єдиного соціального податку, складе

20 тис. грн. У підсумку витрати на розробку ІС, включаючи формування початкової інформаційної бази і навчання персоналу, складуть 36 тис. грн.

Витрати на технічне обслуговування протягом року і витрати на заробітну плату співробітника, який працює з ІС, складе 47,7 тис. грн. на рік.

Після впровадження даної ІС скоротяться витрати по матеріальним витратам на 20%. Це призведе до зниження умовної загальної собівартості робіт, дана економія дозволить окупити створення і впровадження ІС. Річний економічний ефект становить 35,66 тис. грн., Так як дана величина більше нуля, отже проект вважається ефективним. Передбачуваний час окупності проекту становить 4,3 місяця.

Висновки до розділу 2

У цьому розділі розглянуто використання веб-додатки декількома групами користувачів і організація розподіленого доступу для цих груп, що дозволяє забезпечити доступ різних груп користувачів до різних можливостей системи.

Розпізнавання груп користувачів відбувається в процесі реєстрації в системі «Співробітника» або «Заявника».

Для функціонального моделювання бізнес-процесів Центру пропонується використання діаграми потоків даних DFD, так як дана методологія дає більше переваг при проектуванні інформаційної системи.

При створенні логічної моделі бази даних в цьому проекті використовується розробка MySQL Workbench. Цей інструмент спрощує проектування і обслуговування баз даних, економить багато часу, спрощує схильні до помилок завдання і покращує зв'язок серед команд розробників і архітекторами БД.

Так само проведена часткова нормалізація бази даних системи, тому що в реальних, «промислових» базах даних розробники вибирають між нормалізацією до «переможного кінця» і ускладненням програмного забезпечення, що реалізує запити до бази даних і процедури адміністрування баз даних.

У цьому проекті використовувалася ручна верстка html та css коду. Це пов'язано з тим, що основною метою проекту стало не створення повнофункціонального інтернет-сайту організації, а проектування та реалізація окремих складових бази даних, окремих програмних компонент з оперативного управління базою даних, представлення даних на екрані дисплея, інтерфейсу з користувачем.

Для реалізації клієнт-серверної архітектури вибрано тріо базового програмного забезпечення, що базується на Linux платформі - віддалений сервер зі стандартним програмним забезпеченням Apache, PHP, MySQL і широко поширений пакет інтерактивного віддаленого управління базою даних phpMyAdmin.

Для розробки веб-додатків використовується модифікований під додаток клас PHP для роботи з MySQL.

Для створення бази даних компанії control_clients_mid був використаний пакет phpMyAdmin, що забезпечує веб-інтерфейс для створення і адміністрування бази даних веб-додатків.

Веб-додаток Control_Clients_Mid створено на мові PHP, з використанням об'єктно-орієнтованого програмування (ООП).

Розробка програмних модулів не велася з нуля, а були істотно модифіковані необхідні програмні модулі, функції класи PHP, взяті з мережі.

Незважаючи на це, відсоток нових доробок адаптованих під даний додаток становить близько 90%.

Для з'єднання з базою даних був використаний клас `DB_Class`, який реалізує основні запити до бази даних.

Адміністрування бази даних (додавання нових даних в таблиці, видалення даних, оновлення даних) може проводитися двома шляхами: з використанням інтерактивного пакета `phpMyAdmin` і безпосередньо через інтерфейс цього додатка з використанням таблиць уявлення і форм введення.

Управління даними здійснюється за допомогою браузера через веб-інтерфейс - спеціально розробленими уявленнями таблиць, форм, послідовностей команд.

Для реалізації пошукових функцій використовується клас `DB_Class` роботи з базою даних `MySQL` і стандартна мова запитів `SQL`. Програмне забезпечення влаштовано таким чином, щоб передбачити можливість зміни структури полів БД. Для цього, по-перше, використовуються файли настройки `Table_Heades.php` і `Table_Headers_DB.php`, а також локалізація пошукових запитів.

Для виконання пошуку по декількох ключах для кожної таблиці розроблені найбільш раціональні, що відповідають реальним потребам вибірки необхідної інформації, форми.

Для фізичної реалізації логічної структури бази даних використовувався інтерактивний пакет `phpMyAdmin`.

Для оцінки економічної ефективності проекту застосовуються статичні і динамічні показники. Статичні показники не враховують різночасовість витрат на ІС та результатів від її функціонування і використовуються при невеликих термінах створення системи (до року).

Динамічні показники враховують розподіл витрат і результатів у часі, ґрунтуючись на концепції тимчасової цінності грошей. Їх використання доцільне при тривалих термінах створення системи.

Після впровадження даної ІС скоротяться витрати по матеріальним витратам на 20%. Це призведе до зниження умовної загальної собівартості робіт, дана економія дозволить окупити створення і впровадження ІС. Річний економічний ефект становить 35,66 тис. грн., Так як дана величина більше нуля, отже проект вважається ефективним. Передбачуваний час окупності проекту становить 4,3 місяця.

РОЗДІЛ 3 ОХОРОНА ПРАЦІ ТА ТЕХНІКА БЕЗПЕКИ

Проектування робочих місць відео терміналу є однією з найважливіших проблем ергономічного дизайну в області обчислень. Ергономічними аспектами роботи відео терміналу є:

- а) висота робочої поверхні;
- б) розміри ноги простору;
- в) вимоги до місця розташування документів на робочому місці (наявність і розмір стенду для документів, можливість розміщувати документи, відстань від користувача до екрану, документ, клавіатура;
- г) характеристика робочого крісла;
- д) вимоги до робочого столу;
- е) регулювання робочого місця та його елементів;

Робочі місця ПЕОМ з відеотерміналами по відношенню до світлових прорізів повинні розташовуватися так, щоб природне світло падало збоку переважно зліва. При цьому слід дотримуватись таких умов:

- а) рівномірне освітлення всієї робочої області;
- б) прилади, за можливості встановлювати в місцях, віддалених від вікон;
- в) вибирати на пряме освітлення або вкривати корпусу світильників;
- г) світло , що надходить через вікна пом'якшувати за допомогою штор;
- д) робоче місце організувати так, щоб напрям погляду було по можливості паралельно фронту вікон.

При організації робочого місця, рекомендується враховувати дані антропометрії. Рух працівника повинен бути таким, щоб його групи м'язів були завантажені рівномірно, а непотрібні непродуктивні рухи ліквідовані. Конструкція робочого місця повинна забезпечувати, щоб трудові операції

виконувалися в межах легкої досяжності і оптимальної площі автомобільного поля, що відображається на рисунку 3.1.

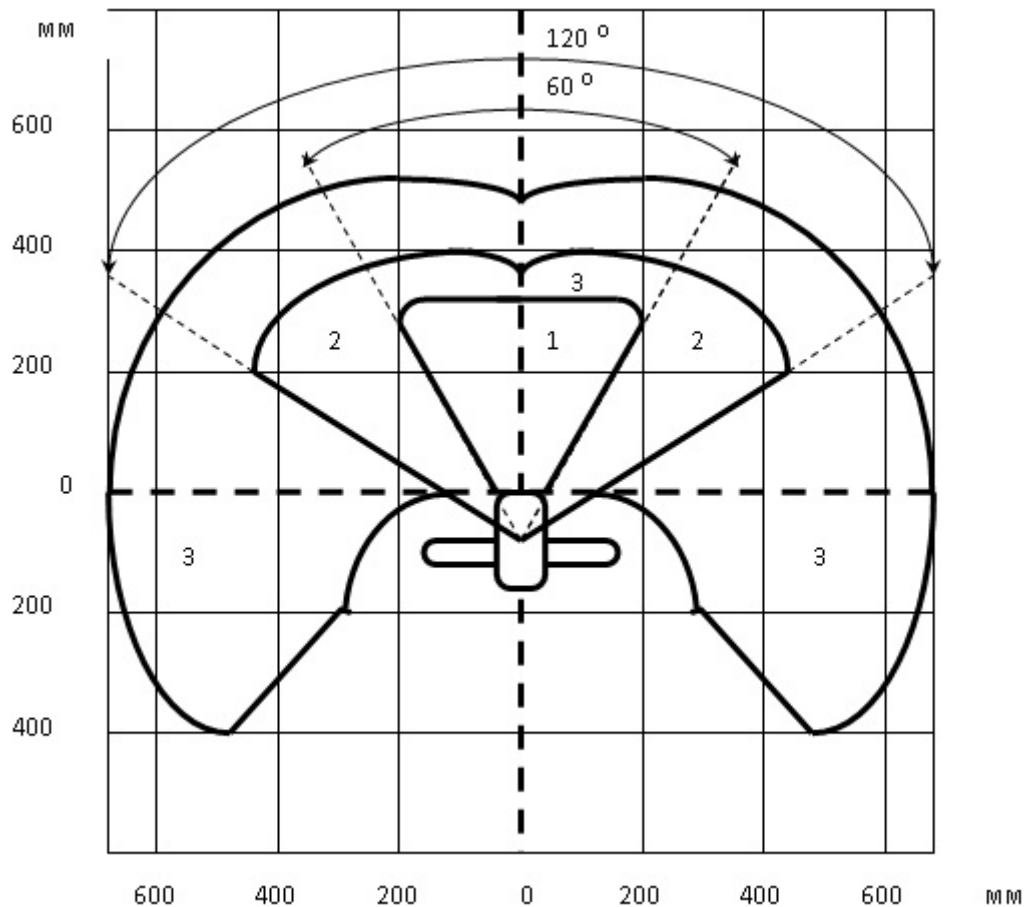


Рис. 3.1. Области для виконання ручних операцій та розміщення органів управління

1 - область для найважливіших і дуже часто використовуваних елементів управління (Оптимальна площа автомобільного поля);

- а) зона для розміщення широко використовуваних органів (зона легкого доступу);
- б) зона для розміщення рідко використовуваних органів управління (зона досяжності моторного поля);

При організації робочих місць слід врахувати наступні умови:

- а) відстань між стільницями з відеотерміналами має бути менше не менше 1,2 метрів;

- б) висота робочої поверхні рекомендована між 680-760 мм, а поверхня, на якій, на встановлена клавіатура – 650;
- в) висота сидіння над пола рівнем підлоги повинна знаходитися в діапазоні 420-550 мм, а кут сидіння. Поверхня сидіння рекомендується бути м'якою, ріжучої кромки округлою, а кут нахилу - регульований;
- г) на робочих місцях рекомендується передбачати підставку для ніг, причому її довжина повинна бути 400 мм, а ширина - 350 мм, регулювання висоти від 0 до 150 мм, кут нахилу від 0 до 200.

Також на екранах моніторів потрібна наявність антистатичне покриття.
Вимоги до вентиляції, опалення та кондиціонування повітря.

У офісному просторі, де робота з монітором і ПЕОМ є основою, необхідно надати оптимальні параметри мікроклімату, а також наступні умови:

- а) для підвищення вологості повітря в приміщеннях з моніторами і ПЕОМ слід застосовувати зволожувачі повітря, що заправляють щодня дистильованої;
- б) система опалення повинна забезпечувати достатню постійне і рівномірне нагрівання повітря в холодну пору року. При цьому коливання температури протягом доби не повинно перевищувати 2 - 30 С;
- в) в приміщеннях з надлишками тепла необхідно передбачати регулювання подачі теплоносія.

Наступні обсяги зовнішнього повітря повинні постачатися в офісний простір:

- а) при обсязі приміщення до 20 м³ на одного працюючого не менше 30 м³ / год на людину;
- б) при обсязі приміщення 20 - 40 м³ на одного працюючого не менше 20 м³/год на людину;

в) при обсязі приміщення більше 40 м³ на одного працюючого при наявності;

У офісах без вікон і легких ліхтарів, подача повітря для одного працівника повинна бути не менше 60 м³/год, при дотриманні стандартів мікроклімату і максимально допустимих концентраціях шкідливих речовин.

Таблиці 3.1 Нормативи температури в приміщенні з ПЕОМ

	Холодний сезон року	Теплий сезон року
	Оптимальне Значення	Оптимальне Значення
Температура, С _о	22 – 24	23 – 25
Вологість, %	40 – 60	40 – 60
Швидкість повітря, м/с	0,1	0, 1 – 02

Для забезпечення належної повітряної композиції необхідні наступні умови:

- а) систематичне провітрювання;
- б) вологе щоденне прибирання;
- в) щомісячне протирання спиртом клавіатури і екрану з метою усунення мікроорганізмів;
- г) встановити зволожуючі засоби і кондиціонери.

Кондиціонер повинен гарантувати, що параметри мікроклімату автоматично зберігаються в межах необхідних лімітів протягом всіх сезонів року. Автономні кондиціонери встановлюються в віконних рамах, а їх кількість визначається розрахунком, в залежності від надлишку тепла, що випромінюється машинами, людьми і сонячним опроміненням.

Вимоги до висвітлення в приміщеннях обчислювальних центрів.

Природне освітлення залежить від розмірів світлових прорізів, світлотехнічних якостей світлопрозорих загороджень, світлового клімату

місцевості, орієнтації приміщень і світлових прорізів відносно сторін світу. При виконанні роботи високої зорової точності коефіцієнт природної освітленості (КПО) повинен бути не нижче 1,5%, а при виконанні робіт середньої точності - не нижче 1%. Орієнтація світлових прорізів для приміщень ПЕОМ повинна бути північній, північно-західній або північно-східній.

Штучне освітлення в приміщеннях повинно здійснюватися у вигляді комбінованої системи освітлення з використанням люмінесцентних джерел світла, генеральних світильників, які повинні розташовуватися на робочих поверхнях в рівномірно прямокутному порядку.

Локальне освітлення забезпечується лампами, встановленими безпосередньо на поверхні столу або його вертикальної панеллю. Джерела світла повинні бути поміщені таким чином, щоб запобігти пряме світло від введення очей. При цьому захисний кут світильників в цих ламп повинен бути не менше 30⁰, а пульсація легких використовуваних люмінесцентних ламп не повинна перевищувати 10%.

Для виключення підсвічування відображення екранів прямими світловими потоками загальної освітлювальної лампи розташовані на стороні робочого місця, паралельно лінії зору оператора. Для тих самих цілей використовуються анти світловідбиваючі решітки та спеціальні фільтри для екранів. Коли освітлювальна апаратура в рядках, дисплеї не допускаються до розташування один в одному. При установці відеотерміналів у великих приміщеннях, щоб зменшити яскравість зміни, слід використовувати мобільні вертикальні перегородки, висота яких буде захищати вигляд тих, хто працює з сусідніх районів з різною яскравістю. У полі зору оператора слід забезпечити відповідне розподілення яскравості, а відношення яскравості екрану до яскравості навколишніх поверхонь не повинно перевищувати 3:1.

Вимоги до шуму і вібрації в приміщеннях обчислювальних центрів.

Допустимі рівні шуму і вібрації на робочому місці повинні відповідати вимогам «санітарних стандартів допустимого на робочому місці». Вібрація (загальне) обладнання не повинна перевищувати максимальних допустимих значень, встановлених "санітарними нормами вібрації на робочому місці".

Для зниження рівня шуму і вібрації в офісних приміщеннях, обладнання, обладнання та прилади повинні встановлюватися на спеціальних фундаментах і амортизаційних трибунах, передбачених нормативними актами. Стіни і стелі офісного приміщення, де встановлюються комп'ютери, повинні бути облицьовані звукопоглинаючим матеріалом і, крім того, слід використовувати підвісні акустичні стелі. На висоті більше 3,5 метрів до стелі, необхідно повісити звукопоглинач у вигляді хрестика і поздовжніх діафрагм, оброблених з обох сторін звукопоглинаючим матеріалом.

Додатковим звукопоглинанням є штори з щільної тканини, що гармонізує з кольором стін і підвішений в складку на відстані 15-20 см від огорожі. Ширина фіранки повинна бути вдвічі ширшою, як у віконці. У таблиці показано значення рівнів шуму, які не повинні бути перевищені.

Таблиці 3.2 Рівні шуму для різних категорій робітників

Категорії робітників	Рівень шуму, дБ
Математики, програмістів, термінальних операторів	50
Інженери, які здійснюють аналітичний та вимірний контроль	60
Комп'ютери (без дисплея)	65
Персонал, що працює в кімнатах з галасливими підрозділами	75

Захист від статичної електрики і електромагнітної радіації.

Статична електроенергія являє собою набір явищ, пов'язаних з появою, збереженням і рухом безкоштовного електричного заряду на поверхні або в об'ємі діелектриків або на ізольованих проводах.

У офісному просторі, розряд статичної електрики відбувається, коли персонал доторкнувся до будь-якого з комп'ютерів. Такі розряди не становлять небезпеки для людини, однак, крім неприємних відчуттів, вони можуть привести до виходу з ладу комп'ютерів.

Захист від статичної електрики має здійснюватися відповідно до санітарно-технічним нормам допустимого натягу електростатичного поля. Допустимі рівні напруженості електростатичного поля не повинні перевищувати 20 кВ протягом однієї години.

Для зменшення обсягу статичної електрики в офісних приміщень необхідно проводити з одношарового полівінілхлоридного лінолеуму. Іншим способом захисту є нейтралізації заряду з іонізованим газом. Загальний захист від статичної електрики включає в себе загальне і місцеве зволоження повітря.

У офісному просторі необхідно контролювати рівень аероіонізації. Слід врахувати, що світлові рентген, що виникає при напрузі на аноду 20-22 кВ, а також висока напруга на нинішніх ділянках ланцюга викликають іонізацію повітря з утворенням позитивних іонів, несприятливої для людини. Оптимальним рівнем аероіонізації в зоні дихання працюючого вважається з'єднання легких аероіонів обох знаків від $1,5 \cdot 10^2$ до $5 \cdot 10^3$ см³ повітря.

Питання електромагнітної радіації монітора дуже важливий, а діапазон випромінювання комп'ютера включає в себе рентгенівський, Ультрафіолетовий і інфрачервоний регіони спектру, а також широкий діапазон електромагнітних хвиль інших частот.

Щоб зменшити потенційно небезпечні радіації, необхідно використовувати відеотермінали для виконання спеціальних заходів захисту від низькочастотних полів. Джерело високовольного дисплея-невеликий

трансформатор-укладається в задній або бічній частині терміналу, а стінки корпусу не захищають радіацію, тому користувачі повинні бути не ближче, ніж 1,2 м від задньої і бічних поверхонь сусідніх терміналів.

Пожежна безпека в офісах офісних центрів.

Пожежі в офісних центрах (ОЦ) є особливо небезпечними, оскільки вони передбачають великі матеріальні збитки. Характерною особливістю ЦК є невеликі ділянки приміщення. Як відомо, пожежа може виникати при горючі речовини, взаємодії окислювача і запалення. Всі три основні чинники, необхідні для виникнення пожежі, присутні в приміщеннях ОЦ.

Горючі компоненти в ОЦ є: будівельні матеріали для акустичного і естетичного оздоблення приміщень, перегородок, дверей, підлог, перфокарт і пучків, ізоляції потужності, сигнальних кабелів, обмотувальних радіотехнічних деталей, ізоляції сполучних кабелів клітин, блоків, підблоків, панелей, стійок, шаф, рідин для очищення елементів і вузлів від забруднення.

Для відводу тепла з комп'ютерів в офісних приміщеннях ЦК постійно працює потужна система кондиціонування повітря. Як правило, кондиціонування повітря здійснюється як в допоміжних, так і в офісних установах. Тому кисень, як окислювач процесів горіння, доступний в будь-якому місці в приміщеннях ОЦ.

Джерелами запалювання в ОЦ можуть бути: електронні комп'ютерні ланцюги, пристрої, що використовуються для технічного обслуговування, електроприладів, кондиціонерів, де в результаті різних порушень знаходяться перегріті елементи, електричні іскри і дуги, які можуть викликати горючі матеріали для займання.

Пожежна безпека систем вентиляції та кондиціонування повітря в офісах офісних центрів.

Однією з характерних особливостей сучасних ЕОМ є велика поверхнева щільність потоку розсіюється теплової енергії: 5-60 Вт / см². Для відводу надлишкової теплоти від обчислювальних засобів і створення

оптимальних метеорологічних умов на робочих місцях в приміщеннях ОЦ широко використовуються системи вентиляції і кондиціонування повітря.

Для тепло і звукопоглинаючої ізоляції систем вентиляції необхідно застосовувати такі негорючі матеріали, як мати з мінеральної вати, скловолокна, стіліта.

При виявленні пожежі електроприводи вентиляторів необхідно негайно вимкнути. Вимкнення здійснюється автоматично по команді системи пожежної сигналізації або в ручну. Одночасно відключається електроживлення електронних пристроїв.

Первинна система пожежогасіння в офісах офісних центрів.

Первинні засоби гасіння пожеж, які призначені для використання для локалізації дрібних пожеж, включають пожежні бочки, внутрішні пожежні тгрни, Вогнегасники, сухі пісок, азбестоцементні ковдри і т. д.

У приміщеннях ОЦ пожежної установки встановлюються в коридорах, на ділянках сходових свердловин і входів. Вода використовується для гасіння пожеж в приміщеннях програмістів, допоміжних і офісних приміщень.

У офісних приміщеннях(ОС) застосовують переважно вуглекислотний Вогнегасники, переваги яких-висока ефективність гасіння пожежі, безпека електронного обладнання, діелектричні властивості двоокису вуглецю, що дозволяє використовувати ці вогнегасники навіть тоді, коли неможливо деактивувати електричну установку відразу.

Ручні вогнегасники встановлюються в приміщеннях з обчислювальним обладнанням зі швидкістю одного вогнегасника на 40-50 м² області, але не менше двох приміщеннях.

Масу вуглекислотного вогнегасників тестується не рідше одного разу на три місяці, а огляд з гідравлічним тестом-через п'ять років.

Висновки до розділу 3

В даному розділі розглянуто заходи та вимоги до безпеки праці , адже забезпечення безпеки праці кожного працівника є завданням великого соціально-політичного і економічного значення.

Для того щоб людина могла плідно працювати, їй необхідно створити оптимальні умови для виконання виробничих обов'язків. Існує цілий ряд норм, виконання яких дозволяє багато в чому забезпечити такі умови, а також зменшити негативний вплив техніки на здоров'я працівника. Для успішної роботи дуже важливо правильно спланувати робоче місце, яке повинно задовольняти вимогам зручності виконання робіт, раціонального використання площ і обсягів, дотримання правил техніки безпеки.

ВИСНОВОК

На підставі аналізу роботи центру колективного користування науковим обладнанням було зроблено висновки про створення веб-орієнтованої ІС.

В роботі розглянуто створення та розвиток програмного забезпечення інформаційних систем у лабораторії, що базується на сучасних програмних платформах, сучасних концепціях в області систем управління базами даних (БД), сучасних концепціях прикладного програмування, інтернет-технологій. Це використання поєднання цих технологій, що дозволяє отримати доступ до бази даних і компонентів програмного забезпечення не тільки для співробітників закладу, але і для всіх зацікавлених сторін у цій сфері діяльності, кожна зі своєю власною, певною інформацією.

Розроблена база даних, програмні компоненти представлених веб-додатків є однією з компонентів, що забезпечує швидке збереження та управління інформацією, оперативний пошуку інформації, накопичення інформації з метою отримання статистичних даних та проведення аналітичних досліджень.

Розроблений веб-орієнтованої ІС має клієнт-серверну архітектуру.

Розроблено інтуїтивно зрозумілий веб-інтерфейс для дистанційної роботи з компонентами бази даних та програмного забезпечення.

Використання системи надає принципово нові можливості у взаємодії працівників Центру та замовників через Інтернет та забезпечує доступ до інформації. Використання системи дозволяє: розробляти нові технологічні моделі взаємодії замовників з клієнтами і персоналу установи; отримати актуальну інформацію різного виду по пошуковим запитам; накопичувати, класифікувати інформацію про заявників; приймати швидкі рішення; удосконалює існуючі бухгалтерські документи; скорочення та уніфікація документів; зменшити ймовірність помилок і втрати документів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Фролов А.В. Створення веб-додатків: практичний посібник/ А.В. Фролов, Г.В., «Фролов. -М.: Видавничо-торговий дім «Російська редакція», 2001. - 420с.
2. Дрібас В.П . Реляційні моделі баз даних/ В.П. Дрибас. -М.: Машинобудування, 2005. -304с.
3. Кормен Т. Алгоритми: будівництво та аналіз/ Т. Кормен, Ч. Лейзерсон, Р. Рівест. -М.: МЦНМО, 2001. -450с.
4. Томсон Л. Розробка веб-додатків на PHP і MySQL/ Л. Томсон, Л. Веллінг. - М.: Видавничий дім DiaSoft, 2001. -560с.
5. Карпова Т. Бази база даних. Моделі, розробка, реалізація/ Т. Карпова. -М.: Видавництво «Вільямс», 2001. -300с.
6. Конноллі Т. Бази даних. Проектування, реалізація, технічне обслуговування. Теорія і практика/ Т. Конноллі, К. Бегг, Страчан А. -М.: Видавництво «DiaSoft», 2009. -460с.
7. Мартін Д. Бази база даних. Практичні методи/ Д. Мартін. -М.: Видавництво "Мир", 2005. -580с.
8. Мейер Д. Теорія реляційної бази даних/ Д. Мейер. -М.: Статистика, 2007. - 380с.
9. Олле Т. Пропозиція КОДАСИЛ управління базами/ Т. Олле. -М.: Видавництво «Вільямс», 2010. -379с.
10. Потоцький В.К. Технологія створення та управління роботою реляційної бази даних на персональних комп'ютерах/ В.К. Потоцький. -М.:
11. Чудінов І.Л. Системи управління базами даних/ І.Л. Чудінов. -М.: Машинобудування, 2009. -410с.
12. Зандстра М.php4 в 24 години/ М. Зандстра. -М.: Видавництво «Вільямс», 2008. -230с.
13. Дімітрієва Л.Л. Методичний посібник з проектування курсового та дипломного проекту/ Л. Л . Дімітрієва. - Ростов-на-Дону: РДЕУ «РІНХ», 2009. -35с.

14. Смирнова Г.Н. Проектування економічних інформаційних систем/ Г.Н. Смирнова, Ю.Ф. Тельнов. -М.: МЕСІ, 2004. -223с.
15. Трофімов С.А. Case-технології: практична робота в Rational Rose. Візуальне моделювання/ С.А. Трофімов. -М.: Біном-Пресс, 2002. -228с.
16. Томсон Л. Розробка веб-додатків на PHP і MySQL/ Л. Томсон, Л. Веллінг. - Спб.: ООО «ДиаСофтЮП», 2003. -672с.
17. Шполянська І. Ю. Проектування інформаційних систем з використанням UML і Rational Rose: методичні указівки для вивчення дисципліни для студентів заочної форми навчання/ І.Ю. Шполянська. -Ростов-на-Дону: РДЕУ «РІНХ», 2003. -45с.
18. Фаулер М. UML. Основи/ М. Фаулер, К. Скотт. - СПб.: Символ-Плюс, 2002. -192с.
19. Зеленський С.Е. Автоматизація обліку персоналу: практична допомога/ С.Е. Зеленський. - Київ: "Центр освітньої літератури", 2003. -679с.
20. Веб-сайт розробників Rational Software [Електронний ресурс] - URL-адреса: <https://ibm.com/software/rational/> (Дата адреси: 18.03.2020).
21. Веб-сайт розробників веб-сервера разработчиков Apache [Електронний ресурс] - URL-адреса: <http://httpd.apache.org/> (Дата адреси: 14 травня 2022).
22. Вікіпедія Model-View-Controller [Електронний ресурс] - URL: <http://ru.Wikipedia.org/Wiki/Model-View-Controller> (Дата адреси: 10.04.2020).
23. ER-модель [Електронний ресурс] / - // Wikipedia. – 2019. – Режим доступу до ресурсу: <https://ru.wikipedia.org/wiki/ER-%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C>.

ДОДАТКИ

Додаток А

Функціональна модель діяльності лабораторії біобезпеки, якості харчової продукції та безпеки харчування НУХТ

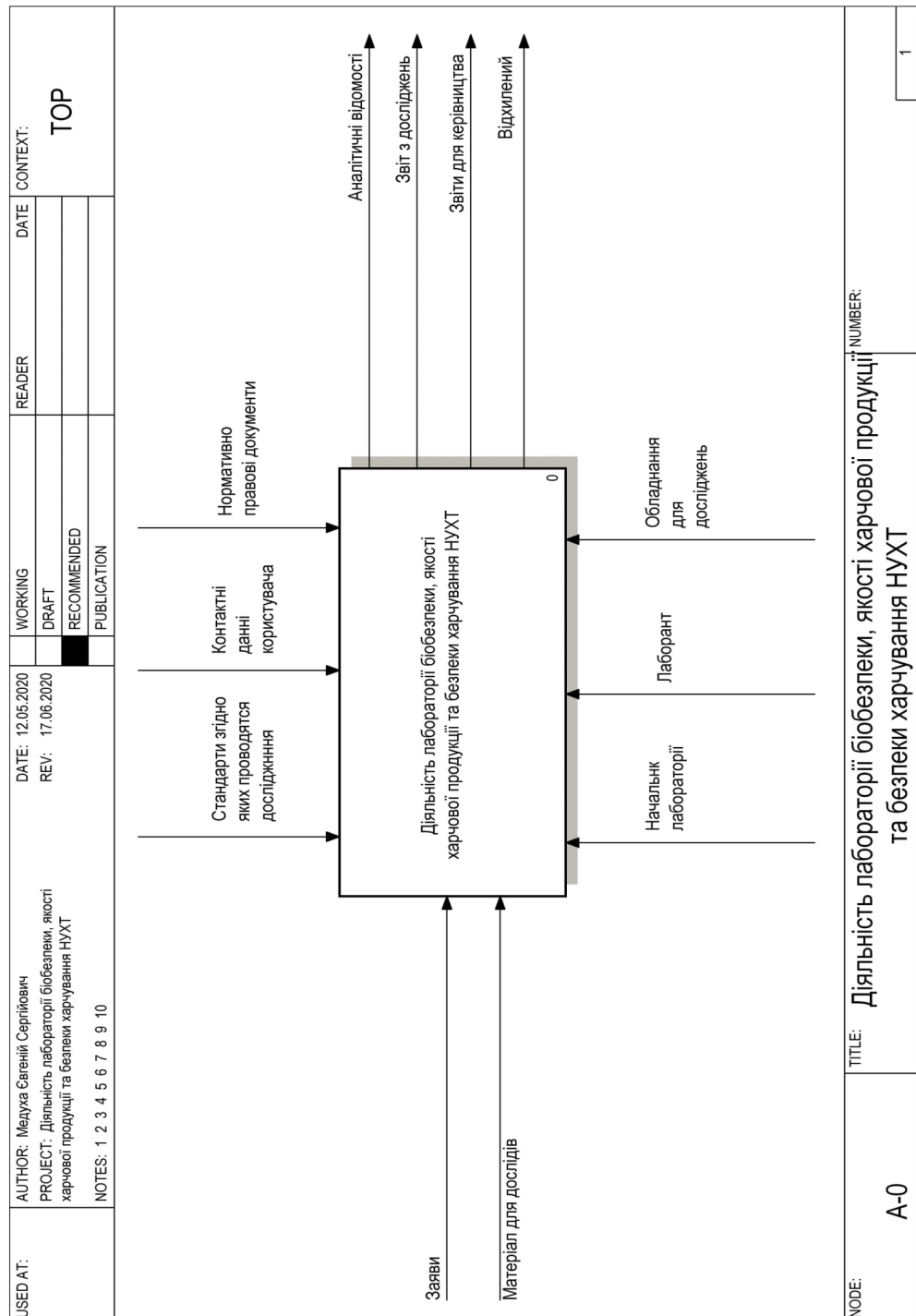


Рис 1. Контекстна діаграма діяльності Лабораторії

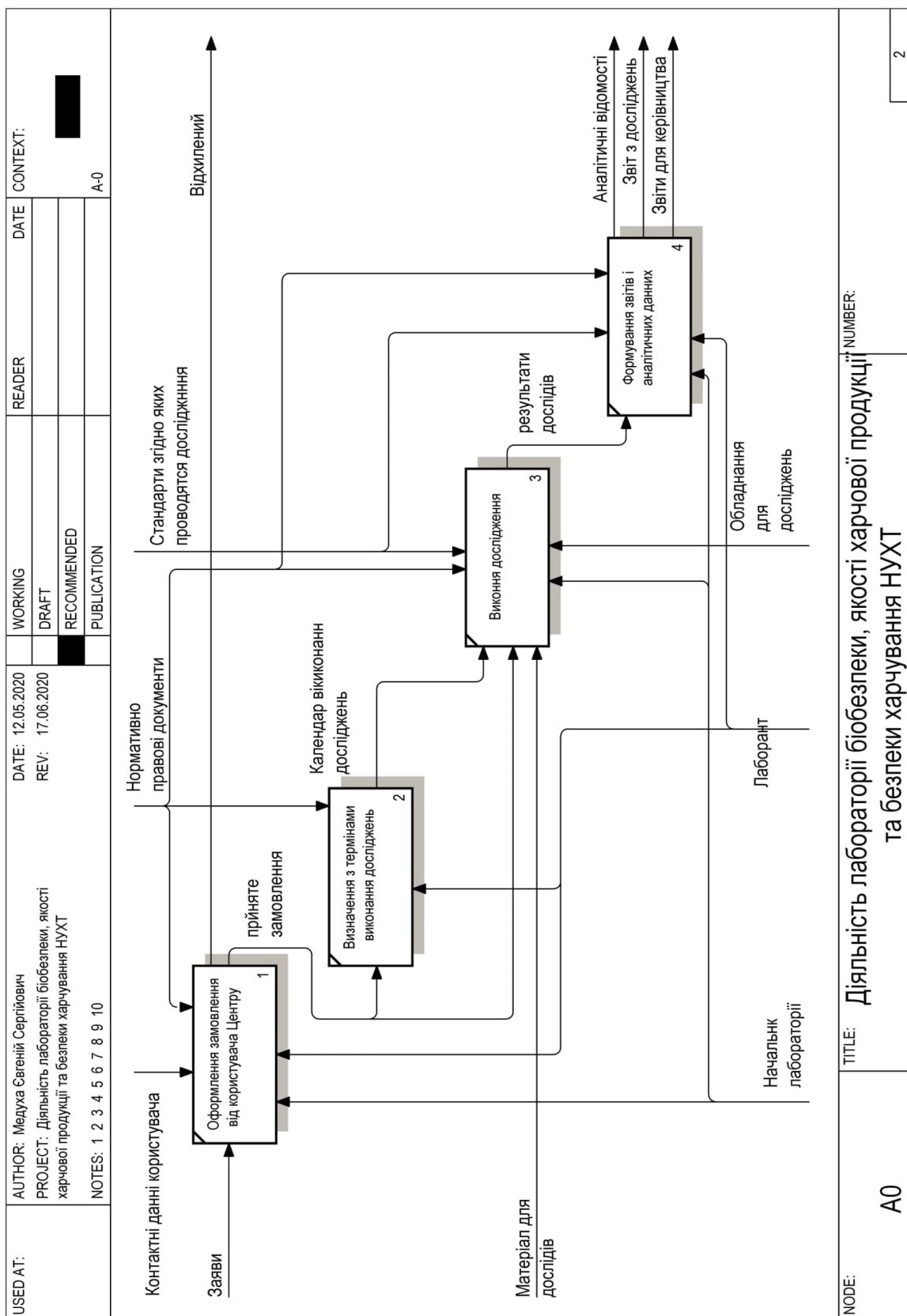


Рис 2. Діаграма декомпозиції першого рівня

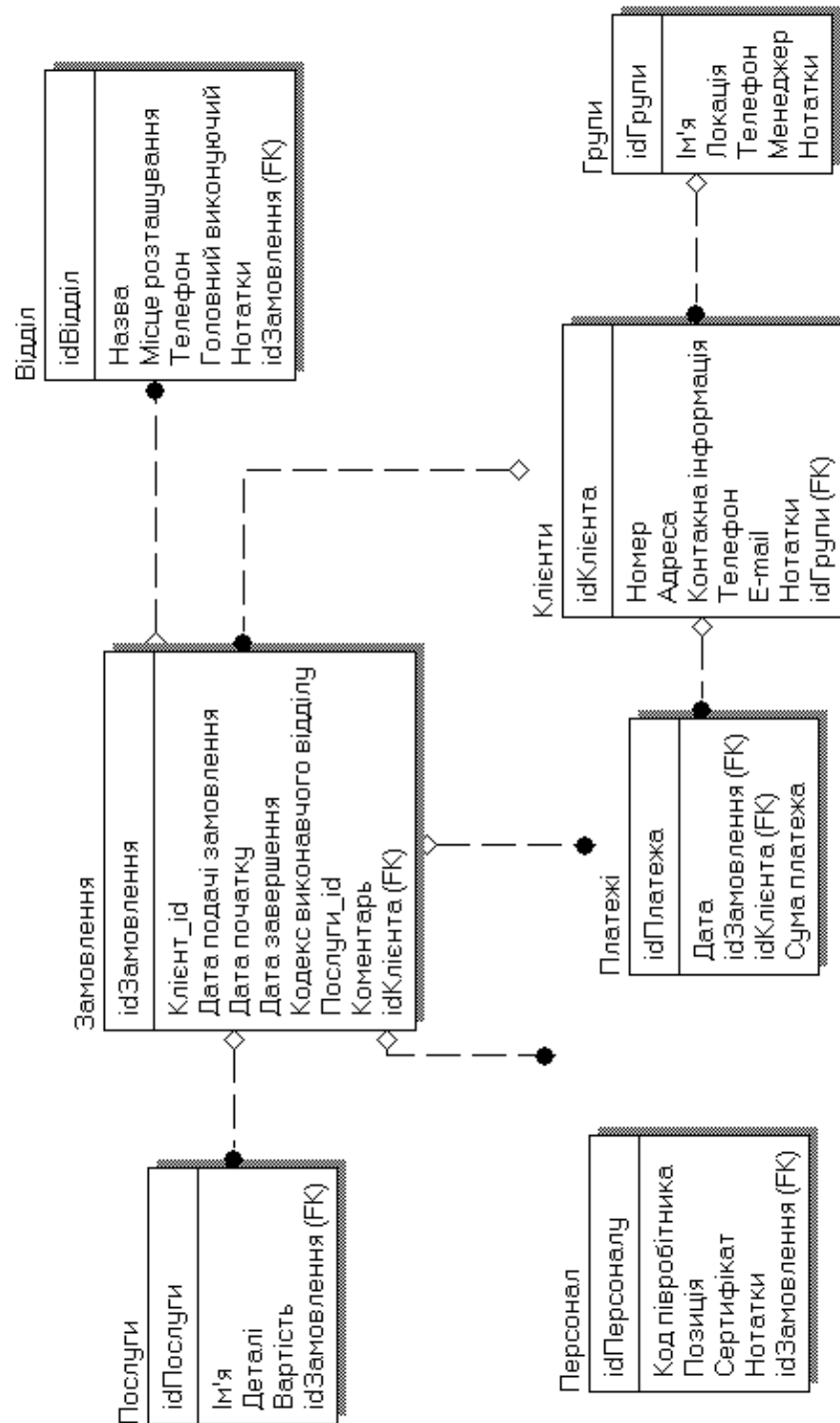


Рис 3. Логічна модель бази даних

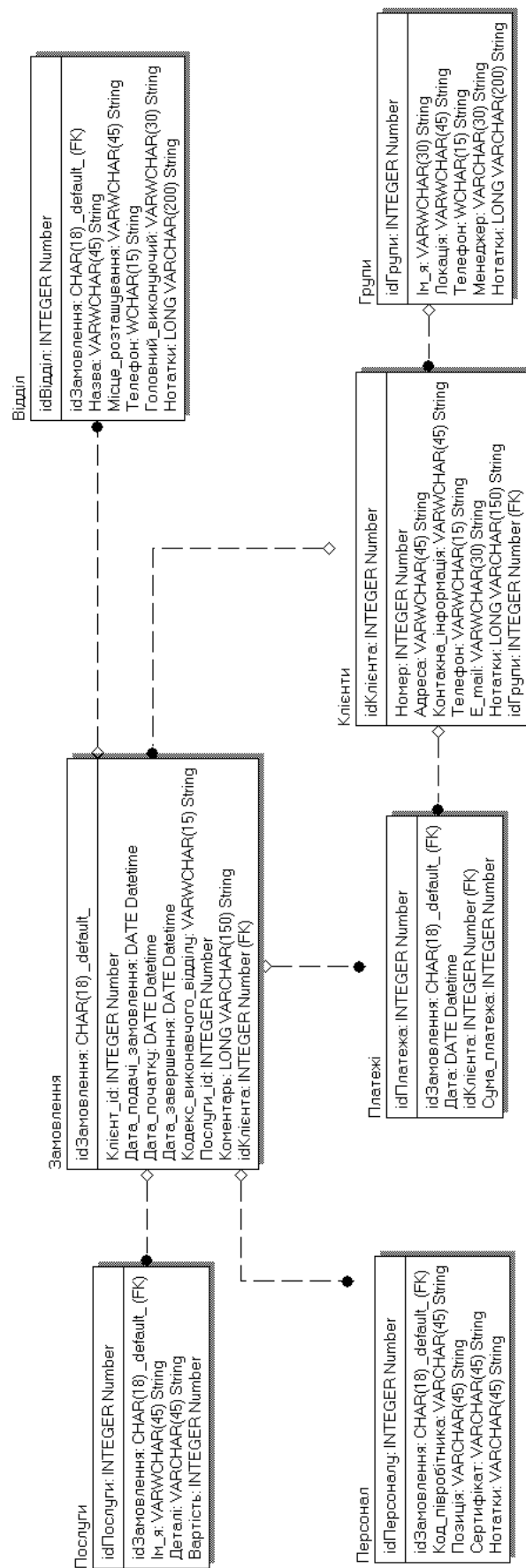


Рис 4. Фізична модель бази



Рис. 5. ER-діаграма згенерована в MySQL Workbench

ReportingController

```

1.<?php
2.
3.namespace backend\controllers;
4.
5.use app\models\NameOrganization;
6.use app\models\Requests;
7.use yii\web\Controller;
8.
9./**
10. * ReportingController implements the CRUD actions for Requests model.
11. * * @property string $name
12. */
13.class ReportingController extends Controller
14.{
15.    /**
16.     * Lists all Requests models.
17.     * @return mixed
18.     */
19.    //TODO:
20.    public function actionIndex()
21.    {
22.        $sql = /** @lang text */
23.            'SELECT T1.dt ,sum(case when T2.name=\'Не вибрано\' then
24.                IFNULL(T2.rq_count, 0)
25.                else 0
26.                end) as "not necessary"
27.            ,sum(case when T2.name=\'Проблемна науково дослідна
28.                лабораторія НУХТ\' then IFNULL(T2.rq_count, 0)
29.                else 0
30.                end) as "NUHT"
31.            ,sum(case when T2.name=\'Одеська національна
32.                академія харчових технологій\' then IFNULL(T2.rq_count, 0)
33.                else 0
34.                end) as "ONAHT"
35.            ,sum(case when T2.name=\'Харківський державний
36.                університет харчування та торгівлі\' then IFNULL(T2.rq_count, 0)
37.                else 0
38.                end) as "XDUHT"
39.            ,sum(case when T2.name=\'Львівський національний
40.                університет ветеринарної медицини\' then IFNULL(T2.rq_count, 0)
41.                else 0
42.                end) as "LNUVM"
43.            ,sum(case when T2.name=\'Херсонський національний
44.                технічний університет\' then IFNULL(T2.rq_count, 0)
45.                else 0
46.                end) as "HNTU"

```

```

41.             ,sum(case when T2.name=\ 'Київський національний
університет технологій та дизайну\ ' then IFNULL(T2.rq_count, 0)
42.                 else 0
43.             end) as "KNUND"
44.             ,sum(case when T2.name=\ 'Одеський державний
екологічний університет\ ' then IFNULL(T2.rq_count, 0)
45.                 else 0
46.             end) as "ODEU"
47.
48.         FROM
49.             ((SELECT 1 dt)
50.             UNION ALL (SELECT 2 dt)
51.             UNION ALL (SELECT 3 dt)
52.             UNION ALL (SELECT 4 dt)
53.             UNION ALL (SELECT 5 dt)
54.             UNION ALL (SELECT 6 dt)
55.             UNION ALL (SELECT 7 dt)) T1
56.         LEFT JOIN
57.
58.         (
59.             SELECT no.name , DAYOFWEEK(rq.date_submission) - 1
day, COUNT(rq.id) rq_count
60.             FROM name_organization no
61.             ,requests rq
62.             WHERE no.id = rq.name_organization_id
63.             AND YEAR(rq.date_submission) = YEAR(NOW()) AND
WEEK(rq.date_submission, 1) = WEEK(NOW(), 1)
64.             GROUP BY no.name , DAYOFWEEK(rq.date_submission) - 1
65.             ) T2 ON T1.dt = T2.day
66.         group by T1.dt';
67.
68.     $model = Requests::findBySql($sql)->asArray()->all();
69.     $names = NameOrganization::find()->all();
70.     $data[0][0] = 'День';
71.     $i = 1;
72.     foreach ($names as $n) {
73.         $data[0][] = $n->name;
74.     }
75.     foreach ($model as $m){
76.         $data[$i] = array_map('intval', array_values($m));
77.         $i++;
78.     }
79.     $data[1][0] = 'Понеділок';
80.     $data[2][0] = 'Вівторок';
81.     $data[3][0] = 'Середа';
82.     $data[4][0] = 'Четверг';
83.     $data[5][0] = 'П\'ятниця';
84.     $data[6][0] = 'Субота';
85.     $data[7][0] = 'Неділя';
86.     return $this->render('index', [
87.         'data' => $data,
88.     ]);
89. }
90. }

```

SiteController

```
1.<?php
2.namespace backend\controllers;
3.
4.use Yii;
5.use yii\helpers\Url;
6.use yii\web\Controller;
7.use yii\filters\VerbFilter;
8.use yii\filters\AccessControl;
9.use common\models\Signup;
10.use common\models\LoginForm;
11.
12./**
13. * Site controller
14. */
15.class SiteController extends Controller
16.{
17.    /**
18.     * {@inheritdoc}
19.     */
20.    public function behaviors()
21.    {
22.        return [
23.            'access' => [
24.                'class' => AccessControl::class,
25.                'only' => ['login', 'logout', 'index', 'error'],
26.                'rules' => [
27.                    [
28.                        'actions' => ['login', 'error'],
29.                        'allow' => true,
30.                    ],
31.                    [
32.                        'actions' => ['logout', 'index'],
33.                        'allow' => true,
34.                        'roles' => ['@'],
35.                    ],
36.                ],
37.            ],
38.            'verbs' => [
39.                'class' => VerbFilter::class,
40.                'actions' => [
41.                    'logout' => ['post'],
42.                ],
43.            ],
44.        ];
45.    }
46.
47.    /**
48.     * {@inheritdoc}
49.     */
50.    public function actions()
51.    {
52.        return [
```

```

53.         'error' => [
54.             'class' => 'yii\web\ErrorAction',
55.         ],
56.     ];
57. }
58.
59. /**
60.  * Displays homepage.
61.  *
62.  * @return string
63.  */
64. public function actionIndex()
65. {
66.     return $this->render('index');
67. }
68.
69. /**
70.  * Login action.
71.  *
72.  * @return string
73.  */
74. public function actionLogin()
75. {
76.     if (!Yii::$app->getUser()->isGuest) {
77.         return $this->goHome();
78.     }
79.
80.     $model = new LoginForm();
81.     if ($model->load(Yii::$app->getRequest()->post()) && $model->login()) {
82.         return $this->goBack();
83.     } else {
84.         return $this->render('login', [
85.             'model' => $model,
86.         ]);
87.     }
88. }
89.
90. /**
91.  * Signup new user
92.  *
93.  * @return string
94.  */
95. public function actionSignup()
96. {
97.     $model = new Signup();
98.     if ($model->load(Yii::$app->getRequest()->post())) {
99.         if ($user = $model->signup()) {
100.             return $this->goHome();
101.         }
102.     }
103.
104.     return $this->render('signup', [
105.         'model' => $model,
106.     ]);

```

```
106.         }
107.
108.         /**
109.          * Logout action.
110.          *
111.          * @return string
112.          */
113.         public function actionLogout()
114.         {
115.             Yii::$app->user->logout();
116.
117.             return $this->redirect(Yii::$app->urlManagerFrontend-
                >createUrl(''));
118.         }
119.     }
```

RequestsController

```
1.<?php
2.
3.namespace backend\controllers;
4.
5.use app\models\Files;
6.use common\models\User;
7.use kartik\grid\EditableColumnAction;
8.use Yii;
9.use app\models\Requests;
10.use app\models\RequestsSearch;
11.use yii\helpers\ArrayHelper;
12.use yii\helpers\Url;
13.use yii\web\UploadedFile;
14.use yii\web\Controller;
15.use yii\web\NotFoundHttpException;
16.use yii\filters\VerbFilter;
17.
18./**
19. * RequestsController implements the CRUD actions for Requests model.
20. */
21.class RequestsController extends Controller
22.{
23.    public $file;
24.    /**
25.     * {@inheritdoc}
26.     */
27.    public function behaviors()
28.    {
29.        return [
30.            'verbs' => [
31.                'class' => VerbFilter::class,
32.                'actions' => [
33.                    'delete' => ['POST'],
34.                ],
35.            ],
36.        ];
37.    }
38.
39.    // TODO: Ned fix formatter not validate date
40.
41.    /**
42.     * @return array
43.     */
44.    public function actions()
45.    {
46.        return ArrayHelper::merge(parent::actions(), [
47.            'editrequests' => [
48.                'class' => EditableColumnAction::class,
49.                'modelClass' => Requests::class,
50.                'outputValue' => function
51.                    ($model, $attribute, $key, $index) {
52.                        $fmt = Yii::$app->formatter;
```

```

52.         $value = $model->$attribute;
53.         if ($attribute === 'date_submission') {
54.             return $fmt->asDate($value, 'yyyy-MM-dd');
55.         }
56.         return '';
57.     },
58. //     'outputMessage' => function ($model, $attribute, $key,
59. //         $index) {
60. //         return 'Not correct date';
61. //     },
62. //     'postOnly' => true,
63. ];
64. }
65.
66. /**
67.  * Lists all Requests models.
68.  * @return mixed
69.  */
70. public function actionIndex()
71. {
72.     $searchModel = new RequestsSearch();
73.     $dataProvider = $searchModel->search(Yii::$app->request-
74. >queryParams);
75.     return $this->render('index', [
76.         'searchModel' => $searchModel,
77.         'dataProvider' => $dataProvider,
78.     ]);
79. }
80.
81. /**
82.  * Displays a single Requests model.
83.  * @param integer $id
84.  * @return mixed
85.  * @throws NotFoundHttpException if the model cannot be found
86.  */
87. public function actionView($id)
88. {
89.     return $this->render('view', [
90.         'model' => $this->findModel($id),
91.     ]);
92. }
93.
94. /**
95.  * Creates a new Requests model.
96.  * If creation is successful, the browser will be redirected to the
97.  * 'view' page.
98.  * @return mixed
99.  */
100. public function actionCreate()
101. {
102.     $model = new Requests();

```

```

103.         if ($model->load(Yii::$app->request->post()) && $model-
>save()) {
104.             $id = $model->id;
105.             if ($model->sendEmailCreate($id)) {
106.                 Yii::$app->session->setFlash('success', 'An email has
been sent.');
```

```

107.
108.                 return $this->redirect(['view', 'id' => $model->id]);
109.             } else {
110.                 Yii::$app->session->setFlash('error', 'Sorry, we
could not send the message to your email address.');
```

```

111.             }
112.         }
113.
114.         return $this->render('create', [
115.             'model' => $model,
116.         ]);
117.     }
118.
```

```

119.     /**
120.      * Updates an existing Requests model.
121.      * If update is successful, the browser will be redirected to the
'view' page.
122.      * @param integer $id
123.      * @return mixed
124.      * @throws NotFoundHttpException if the model cannot be found
125.      */
```

```

126.     public function actionUpdate($id)
127.     {
128.         $model = $this->findModel($id);
129.         if ($model->load(Yii::$app->request->post()) && $model-
>save()) {
```

```

130.             $model->file =
UploadedFile::getInstances($model, 'file');
131.             if ($model->uploadFilesMany($id))
132.             {
133.                 Yii::$app->session->setFlash('success', 'File
uploaded successfully.');
```

```

134.
135.                 return $this->redirect(['view', 'id' => $model->id]);
136.             } elseif ($model->checkNumberFiles($id) > 4) {
137.                 Yii::$app->session->setFlash('error', 'Error loading
files, wrong format or too many.');
```

```

138.                 return $this->redirect(['update', 'id' => $model-
>id]);
139.             } else
140.                 return $this->redirect(['view', 'id' => $model->id]);
141.         }
142.
```

```

143.         return $this->render('update', [
144.             'model' => $model,
145.         ]);
146.     }
147.
```

```

148.      /**
149.      * Deletes an existing Requests model.
150.      * If deletion is successful, the browser will be redirected to
151.      * the 'index' page.
152.      * @param integer $id
153.      * @return mixed
154.      * @throws NotFoundHttpException if the model cannot be found
155.      */
156.      public function actionDelete($id)
157.      {
158.          $storagePath = Yii::getAlias('@webroot/uploads');
159.          $files = Files::findAll(['id_FK_requests' => $id]);
160.          foreach ($files as $f) {
161.              $filename = $f->filename;
162.              if ($files !== null) {
163.                  // check filename for allowed chars (do not allow ../
164.                  // to avoid security issue: downloading arbitrary files)
165.                  if (preg_match('/^requests_[0-9]{10}_[\w\-. ]+\. [a-
166.                  z0-9]+$ /iu', $filename) && is_file("$storagePath/$filename")) {
167.                      unlink("$storagePath/$filename");
168.                  }
169.              }
170.          }
171.          Files::deleteAll(['id_FK_requests' => $id]);
172.          $this->findModel($id)->delete();
173.
174.          return $this->redirect(['index']);
175.      }
176.
177.      /**
178.      * Finds the Requests model based on its primary key value.
179.      * If the model is not found, a 404 HTTP exception will be
180.      * thrown.
181.      * @param integer $id
182.      * @return Requests the loaded model
183.      * @throws NotFoundHttpException if the model cannot be found
184.      */
185.      protected function findModel($id)
186.      {
187.          if (($model = Requests::findOne($id)) !== null) {
188.              return $model;
189.          }
190.
191.          throw new NotFoundHttpException(Yii::t('app', 'The requested
192.          page does not exist.));
193.      }
194.
195.      /**
196.      * @param $filename
197.      * @return \yii\console\Response|\yii\web\Response
198.      * @throws NotFoundHttpException
199.      */
200.      public function actionFileDownload($filename)

```

```

197.         {
198.             $storagePath = Yii::getAlias('@webroot/uploads');
199.             // check filename for allowed chars (do not allow ../ to
                avoid security issue: downloading arbitrary files)
200.             if (!preg_match('/^requests_[0-9]{10}_[\w\-. ]+\.[a-z0-
                9]+$/iu', $filename) || !is_file("$storagePath/$filename")) {
201.                 throw new NotFoundException(Yii::t('app', 'The file
                    does not exists.'));
202.             }
203.             return Yii::$app->response-
                >sendFile("$storagePath/$filename", $filename);
204.         }
205.
206.         // TODO: Write error handling
207.         /**
208.          * @param $id
209.          * @param $filename
210.          * @return \yii\web\Response
211.          * @throws NotFoundException
212.          */
213.         public function actionFileDelete($id, $filename)
214.         {
215.             $model = $this->findModel($id);
216.             $model->fileDelete($filename);
217.             return $this->redirect(Url::to('admin'));
218.         }
219.     }

```

CalendarController

```
1.<?php
2.
3.namespace backend\controllers;
4.
5.use Yii;
6.use app\models\Requests;
7.use \yii2fullcalendar\models\Event;
8.use \yii\helpers\Url;
9.use \yii\web\Response;
10.use yii\web\Controller;
11.
12.
13./**
14. * Calendar controller
15. */
16.class CalendarController extends Controller
17.{
18.    /**
19.     * @return string
20.     */
21.    public function actionIndex()
22.    {
23.        return $this->render('index');
24.    }
25.
26.
27.
28.    /**
29.     * @param null $start
30.     * @param null $end
31.     * @param null $_
32.     * @return array
33.     */
34.    public function actionJson($start=NULL,$end=NULL,$_=NULL)
35.    {
36.        \Yii::$app->response->format = Response::FORMAT_JSON;
37.
38.        $times = Requests::find()
39.            ->where(['between', 'research_start_date', $start,$end])
40.            ->orWhere(['between', 'research_end_date', $start,$end])
41.            ->all();
42.        $events = array();
43.
44.        foreach ($times AS $time) {
45.
46.            $Event = new Event();
47.            $Event->id = $time->id;
48.            $Event->title = $time->name_customer;
49.            $Event->allDay = true;
50.            $Event->className = 'event-default';
51.
```

```
52.         $Event->start = date('Y-m-d\TH:i:s\Z',strtotime($time-
    >research_start_date));
53.         $Event->end = date('Y-m-d\TH:i:s\Z',strtotime($time-
    >research_end_date));
54. //         $Event->color = '#' . str_pad(dechex(mt_rand(0, 0xFFFFFFFF)),
    6, '0', STR_PAD_LEFT);
55.
56.         $Event->url = Url::to(['/requests/view', 'id'=>$time->id]);
57.         $events[] = $Event;
58.
59.     }
60.
61.     return $events;
62. }
63. }
```

Requsts.php

```
1.<?php
2.
3.namespace app\models;
4.
5.use common\models\User;
6.use \yii\db\ActiveRecord;
7.use Yii;
8.use yii\web\NotFoundHttpException;
9.
10.
11./**
12. * This is the model class for table "requests".
13. *
14. * @property int $id
15. * @property int $name_organization_id
16. * @property string $name_customer
17. * @property int $customer_status_id
18. * @property string|null $name_srw
19. * @property string|null $years_execution_srw
20. * @property int|null $registration_number_srw
21. * @property string $nature_work
22. * @property string|null $contact_details_customer
23. * @property string|null $name_purpose_work
24. * @property string|null $description_research
25. * @property int|null $return_samples
26. * @property int|null $equipment_id
27. * @property string|null $research_start_date
28. * @property string|null $research_end_date
29. * @property string|null $additional_information
30. * @property string $date_submission
31. * @property int|null $status_id
32. *
33. * @property CustomerStatus $customerStatus
34. * @property Equipment $equipment
35. * @property NameOrganization $nameOrganization
36. * @property Status $status
37. * @property Files[] $files
38. */
39.class Requests extends ActiveRecord
40.{
41.
42.     public $file;
43.     /**
44.      * {@inheritdoc}
45.      */
46.     public static function tableName()
47.     {
48.         return 'requests';
49.     }
50.
51.     /**
52.      * {@inheritdoc}
```

```

53.     */
54.     public function rules()
55.     {
56.         return [
57.             [['name_organization_id', 'name_customer', 'customer_status_id',
58.              'nature_work', 'date_submission'], 'required'],
59.             [['name_organization_id', 'customer_status_id', 'registration_
60.              number_srw', 'return_samples', 'equipment_id', 'status_id'], 'integer'],
61.             [['research_start_date', 'research_end_date', 'date_submission
62.              '], 'safe'],
63.             [['additional_information'], 'string'],
64.             [['name_customer', 'name_srw', 'years_execution_srw', 'nature_
65.              work', 'contact_details_customer', 'name_purpose_work', 'description_resea
66.              rch'], 'string', 'max' => 255],
67.             [['customer_status_id'], 'exist', 'skipOnError' =>
68.              true, 'targetClass' => CustomerStatus::class, 'targetAttribute' =>
69.              ['customer_status_id' => 'id']],
70.             [['equipment_id'], 'exist', 'skipOnError' =>
71.              true, 'targetClass' => Equipment::class, 'targetAttribute' =>
72.              ['equipment_id' => 'id']],
73.             [['name_organization_id'], 'exist', 'skipOnError' =>
74.              true, 'targetClass' => NameOrganization::class, 'targetAttribute' =>
75.              ['name_organization_id' => 'id']],
76.             [['status_id'], 'exist', 'skipOnError' => true, 'targetClass'
77.              => Status::class, 'targetAttribute' => ['status_id' => 'id']],
78.             [['file'], 'file', 'extensions'=>'doc,docx,xls,xlsx,odt,pdf', 's
79.              kipOnError' => true, 'maxFiles' => 4],
80.         ];
81.     }
82.
83.     /**
84.      * {@inheritdoc}
85.      */
86.     public function attributeLabels()
87.     {
88.         return [
89.             'id' => Yii::t('app', 'ID'),
90.             'name_organization_id' => Yii::t('app', 'Name Organization'),
91.             'name_customer' => Yii::t('app', 'Name Customer'),
92.             'customer_status_id' => Yii::t('app', 'Customer Status'),
93.             'name_srw' => Yii::t('app', 'Name Srw'),
94.             'years_execution_srw' => Yii::t('app', 'Years Execution Srw'),
95.             'registration_number_srw' => Yii::t('app', 'Registration
96.              Number Srw'),
97.             'nature_work' => Yii::t('app', 'Nature Work'),
98.             'contact_details_customer' => Yii::t('app', 'Contact Details
99.              Customer'),
100.             'name_purpose_work' => Yii::t('app', 'Name Purpose Work'),
101.             'description_research' => Yii::t('app', 'Description
102.              Research'),
103.             'return_samples' => Yii::t('app', 'Return Samples'),
104.             'equipment_id' => Yii::t('app', 'Equipment'),
105.             'research_start_date' => Yii::t('app', 'Research Start Date'),
106.             'research_end_date' => Yii::t('app', 'Research End Date'),

```

```

91.         'additional_information' => Yii::t('app', 'Additional
    Information'),
92.         'date_submission' => Yii::t('app', 'Date Submission'),
93.         'status_id' => Yii::t('app', 'Status'),
94.         'customerStatusName' => Yii::t('app', 'Customer Status'),
95.         'equipmentName' => Yii::t('app', 'Equipment'),
96.         'nameOrganizationName' => Yii::t('app', 'Name Organization'),
97.         'statusName' => Yii::t('app', 'Status'),
98.         'file' => Yii::t('app', 'File'),
99.         'FileName' => Yii::t('app', 'Files Name'),
100.    ];
101.    }
102.
103.    /**
104.     * Gets query for [[CustomerStatus]].
105.     *
106.     * @return \yii\db\ActiveQuery
107.     */
108.    public function getCustomerStatus()
109.    {
110.        return $this->hasOne(CustomerStatus::class, ['id'
    => 'customer_status_id']);
111.    }
112.
113.    /**
114.     * @return string
115.     */
116.    public function getCustomerStatusName()
117.    {
118.        $customerStatus = $this->customerStatus;
119.        return $customerStatus ? $customerStatus->name : '';
120.    }
121.
122.    /**
123.     * Gets query for [[Equipment]].
124.     *
125.     * @return \yii\db\ActiveQuery
126.     */
127.    public function getEquipment()
128.    {
129.        return $this->hasOne(Equipment::class, ['id'
    => 'equipment_id']);
130.    }
131.
132.    /**
133.     * @return string
134.     */
135.    public function getEquipmentName()
136.    {
137.        $equipment = $this->equipment;
138.        return $equipment ? $equipment->name : '';
139.    }
140.
141.    /**

```

```

142.         * Gets query for [[NameOrganization]].
143.         *
144.         * @return \yii\db\ActiveQuery
145.         */
146.         public function getNameOrganization()
147.         {
148.             return $this->hasOne(NameOrganization::class, ['id'
=> 'name_organization_id']);
149.         }
150.
151.         /**
152.         * @return string
153.         */
154.         public function getNameOrganizationName()
155.         {
156.             $nameOrganization = $this->nameOrganization;
157.             return $nameOrganization ? $nameOrganization->name : '';
158.         }
159.
160.         /**
161.         * Gets query for [[Status]].
162.         *
163.         * @return \yii\db\ActiveQuery
164.         */
165.         public function getStatus()
166.         {
167.             return $this->hasOne(Status::class, ['id' => 'status_id']);
168.         }
169.
170.         /**
171.         * @return string
172.         */
173.         public function getStatusName()
174.         {
175.             $status = $this->status;
176.             return $status ? $status->name : '';
177.         }
178.
179.         /**
180.         * Gets query for [[Files]].
181.         *
182.         * @return \yii\db\ActiveQuery
183.         */
184.         public function getFiles()
185.         {
186.             return $this->hasMany(Files::class, ['id_FK_requests'
=> 'id']);
187.         }
188.
189.         /**
190.         * @return string
191.         */
192.         public function getFilesName() {
193.             $files = $this->files;

```

```

194.         foreach ($files as $f) {
195.             $filename[] = $f->filename;
196.         }
197.         return $filename ? $filename : '';
198.     }
199.
200.     /**
201.      * @param bool $insert
202.      * @return bool
203.      * @throws \yii\base\InvalidConfigException
204.      */
205.     public function beforeSave($insert)
206.     {
207.         $this->research_start_date = Yii::$app->formatter-
>asDate($this->research_start_date, 'yyyy-MM-dd');
208.         $this->research_end_date = Yii::$app->formatter-
>asDate($this->research_end_date, 'yyyy-MM-dd');
209.         $this->date_submission = Yii::$app->formatter->asDate($this-
>date_submission, 'yyyy-MM-dd');
210.         parent::beforeSave($insert);
211.         return true;
212.     }
213.
214.
215.     /**
216.      * @return bool|void
217.      * @throws \yii\base\InvalidConfigException
218.      */
219.     public function afterFind()
220.     {
221.         $this->research_start_date = Yii::$app->formatter-
>asDate($this->research_start_date, 'yyyy-MM-dd');
222.         $this->research_end_date = Yii::$app->formatter-
>asDate($this->research_end_date, 'yyyy-MM-dd');
223.         $this->date_submission = Yii::$app->formatter->asDate($this-
>date_submission, 'yyyy-MM-dd');
224.         parent::afterFind();
225.         return true;
226.     }
227.
228.     /**
229.      * @param $id
230.      * @return bool
231.      */
232.     public function sendEmailCreate($id)
233.     {
234.
235.         /* @var $user User */
236.         $user = User::findOne([
237.             'status' => User::STATUS_ACTIVE,
238.             'id' => Yii::$app->user->getId(),
239.         ]);
240.
241.         if (!$user) {

```

```

242.         return false;
243.     }
244.
245.         return Yii::$app->mailer->compose()
246.             ->setFrom([Yii::$app->params['supportEmail'] =>
                Yii::$app->name . ' robot'])
247.             ->setTo($user->email)
248.             ->setBcc(Yii::$app->params['adminEmail']) //Send copy to
                admin
249.             ->setSubject('Вае замовлення №' . $id . 'прийнято до
                розгляду')
250.             ->setTextBody('Ваше замовлення прийнято до розгляду')
251.             ->setHtmlBody('<b>Ваше замовлення прийнято до
                розгляду</b>')
252.             ->send();
253.     }
254.
255.     /**
256.      * @param null|string $strings
257.      * @return string
258.      */
259.     public function FileNameBase($strings = null)
260.     {
261.         $string = (string)$strings;
262.         $pattern = "/^requests_[0-9]{10}_(?P<match>[\w\-. ]+\. [a-z0-
                9]+)$/iu";
263.         preg_match_all($pattern, $string, $matches);
264.         return $matches['match'][0];
265.     }
266.
267.     /**
268.      * @param int $id
269.      * @return bool
270.      */
271.     public function uploadFilesMany($id)
272.     {
273.         $timestamp = time();
274.         // VarDumper::dump($this->checkNumberFiles($id));
275.         if ($this->file && $this->validate() && $this->
                checkNumberFiles($id) <= 4) {
276.             foreach ($this->file as $file) {
277.                 $filename = 'requests_' . $timestamp . '_' . $file->
                baseName . '.' . $file->extension;
278.                 $fileSave = new Files();
279.                 $fileSave->id_FK_requests = $id;
280.                 $fileSave->filename = $filename;
281.                 $fileSave->save();
282.                 $file->saveAs('uploads/' . $filename);
283.                 $timestamp++;
284.             }
285.             return true;
286.         } else {
287.             return false;
288.         }

```

```

289.     }
290.
291.     /**
292.      * @param $id
293.      * @return bool|int|string|null
294.      */
295.     public function checkNumberFiles($id)
296.     {
297.         $quantity = Files::find()
298.             ->where(['id_FK_requests' => $id])
299.             ->count();
300.         foreach ($this->file as $file) {
301.             $number[] = $file->baseName;
302.         }
303.         $numbers = 0;
304.         $quantify = 0;
305.         if (!empty($number)) {
306.             $numbers = count($number);
307.         }
308.         $quantify = $quantity + $numbers;
309.         return $quantify;
310.     }
311.
312.     public function fileDelete ($filename)
313.     {
314.         $storagePath = Yii::getAlias('@webroot/uploads');
315.         $file = Files::findOne(['filename' => $filename]);
316.         if ($file !== null) {
317.             // check filename for allowed chars (do not allow ../ to
318.             // avoid security issue: downloading arbitrary files)
319.             if (!preg_match('/^requests_[0-9]{10}_[\w\-. ]+\.[a-z0-9]+$/iu', $filename) || !is_file("$storagePath/$filename")) {
320.                 throw new NotFoundHttpException(Yii::t('app', 'The
321.                 file does not exists to patch.));
322.             }
323.             $file->delete();
324.             unlink("$storagePath/$filename");
325.             return true;
326.         }
327.         return false;
328.     }

```

Files.php

```
1.<?php
2.
3.namespace app\models;
4.
5.use Yii;
6.
7./**
8. * This is the model class for table "files".
9. *
10. * @property int $id
11. * @property int $id_FK_requests
12. * @property string $filename
13. *
14. * @property Requests $fkRequests
15. */
16.class Files extends \yii\db\ActiveRecord
17.{
18.    /**
19.     * {@inheritdoc}
20.     */
21.    public static function tableName()
22.    {
23.        return 'files';
24.    }
25.
26.    /**
27.     * {@inheritdoc}
28.     */
29.    public function rules()
30.    {
31.        return [
32.            [['id_FK_requests', 'filename'], 'required'],
33.            [['id_FK_requests'], 'integer'],
34.            [['filename'], 'string', 'max' => 255],
35.            [['id_FK_requests'], 'exist', 'skipOnError' =>
true, 'targetClass' => Requests::className(), 'targetAttribute' =>
[['id_FK_requests' => 'id']],
36.        ];
37.    }
38.
39.    /**
40.     * {@inheritdoc}
41.     */
42.    public function attributeLabels()
43.    {
44.        return [
45.            'id' => Yii::t('app', 'ID'),
46.            'id_FK_requests' => Yii::t('app', 'Id Fk Requests'),
47.            'filename' => Yii::t('app', 'Filename'),
48.        ];
49.    }
50.
```

```
51.     /**
52.      * Gets query for [[FKRequests]].
53.      *
54.      * @return \yii\db\ActiveQuery
55.      */
56.     public function getFKRequests()
57.     {
58.         return $this->hasOne(Requests::class, ['id' => 'id_FK_requests']);
59.     }
60. }
```

Equipment.php

```
1.<?php
2.
3.namespace app\models;
4.
5.use Yii;
6.
7./**
8. * This is the model class for table "equipment".
9. *
10. * @property int $id
11. * @property string $name
12. *
13. * @property Requests[] $requests
14. */
15.class Equipment extends \yii\db\ActiveRecord
16.{
17.    /**
18.     * {@inheritdoc}
19.     */
20.    public static function tableName()
21.    {
22.        return 'equipment';
23.    }
24.
25.    /**
26.     * {@inheritdoc}
27.     */
28.    public function rules()
29.    {
30.        return [
31.            [['name'], 'required'],
32.            [['name'], 'string', 'max' => 255],
33.        ];
34.    }
35.
36.    /**
37.     * {@inheritdoc}
38.     */
39.    public function attributeLabels()
40.    {
41.        return [
42.            'id' => Yii::t('app', 'ID'),
43.            'name' => Yii::t('app', 'Name'),
44.        ];
45.    }
46.
47.    /**
48.     * Gets query for [[Requests]].
49.     *
50.     * @return \yii\db\ActiveQuery
51.     */
52.    public function getRequests()
```

```
53.     {
54.         return $this->hasMany(Requests::class, ['equipment_id' => 'id']);
55.     }
56.
57.     /**
58.      * @return array
59.      */
60.     public static function listMap()
61.     {
62.         return Equipment::find()->select(['name', 'id'])->indexBy('id')-
            >column();
63.     }
64. }
```

CustomerStatus.php

```
1.<?php
2.
3.namespace app\models;
4.
5.use Yii;
6.
7./**
8. * This is the model class for table "customer_status".
9. *
10. * @property int $id
11. * @property string $name
12. *
13. * @property Requests[] $requests
14. */
15.class CustomerStatus extends \yii\db\ActiveRecord
16.{
17.    /**
18.     * {@inheritdoc}
19.     */
20.    public static function tableName()
21.    {
22.        return 'customer_status';
23.    }
24.
25.    /**
26.     * {@inheritdoc}
27.     */
28.    public function rules()
29.    {
30.        return [
31.            [['name'], 'required'],
32.            [['name'], 'string', 'max' => 255],
33.        ];
34.    }
35.
36.    /**
37.     * {@inheritdoc}
38.     */
39.    public function attributeLabels()
40.    {
41.        return [
42.            'id' => Yii::t('app', 'ID'),
43.            'name' => Yii::t('app', 'Name'),
44.        ];
45.    }
46.
47.    /**
48.     * Gets query for [[Requests]].
49.     *
50.     * @return \yii\db\ActiveQuery
51.     */
52.    public function getRequests()
```

```
53.     {
54.         return $this->hasMany(Requests::class, ['customer_status_id'
    => 'id']);
55.     }
56.
57.     /**
58.      * @return array
59.      */
60.     public static function listMap()
61.     {
62.         return CustomerStatus::find()->select(['name', 'id'])-
    >indexBy('id')->column();
63.     }
64. }
```

```

1.NameOrganization.php <?php
2.
3.namespace app\models;
4.
5.use Yii;
6.
7./**
8. * This is the model class for table "name_organization".
9. *
10. * @property int $id
11. * @property string $name
12. *
13. * @property Requests[] $requests
14. */
15.class NameOrganization extends \yii\db\ActiveRecord
16.{
17.    /**
18.     * {@inheritdoc}
19.     */
20.    public static function tableName()
21.    {
22.        return 'name_organization';
23.    }
24.
25.    /**
26.     * {@inheritdoc}
27.     */
28.    public function rules()
29.    {
30.        return [
31.            [['name'], 'required'],
32.            [['name'], 'string', 'max' => 255],
33.        ];
34.    }
35.
36.    /**
37.     * {@inheritdoc}
38.     */
39.    public function attributeLabels()
40.    {
41.        return [
42.            'id' => Yii::t('app', 'ID'),
43.            'name' => Yii::t('app', 'Name'),
44.        ];
45.    }
46.
47.    /**
48.     * Gets query for [[Requests]].
49.     *
50.     * @return \yii\db\ActiveQuery
51.     */
52.    public function getRequests()
53.    {

```

```
54.         return $this->hasMany(Requests::class, ['name_organization_id'
    => 'id']);
55.     }
56.
57.     /**
58.      * @return array
59.      */
60.     public static function listMap()
61.     {
62.         return NameOrganization::find()->select(['name', 'id'])-
    >indexBy('id')->column();
63.     }
64. }
```

RequestsSearch.php

```
1.<?php
2.
3.namespace app\models;
4.
5.use yii\base\Model;
6.use yii\data\ActiveDataProvider;
7.use app\models\Requests;
8.
9./**
10. * RequestsSearch represents the model behind the search form of
11. * `app\models\Requests`.
12. */
13.class RequestsSearch extends Requests
14.{
15.    public $customerStatusName;
16.    public $statusName;
17.    public $nameOrganizationName;
18.    public $equipmentName;
19.
20.    /**
21.     * {@inheritdoc}
22.     */
23.    public function rules()
24.    {
25.        return [
26.            [['id', 'name_organization_id', 'customer_status_id', 'registr
27.            ation_number_srw', 'return_samples', 'equipment_id', 'status_id'], 'intege
28.            r'],
29.            [['name_customer', 'name_srw', 'years_execution_srw', 'nature_
30.            work', 'contact_details_customer', 'name_purpose_work', 'description_resea
31.            rch', 'research_start_date', 'research_end_date', 'additional_information'
32.            , 'date_submission', 'statusName', 'nameOrganizationName', 'equipmentName'
33.            , 'customerStatusName'], 'safe'],
34.        ];
35.    }
36.
37.    /**
38.     * {@inheritdoc}
39.     */
40.    public function scenarios()
41.    {
42.        // bypass scenarios() implementation in the parent class
43.        return Model::scenarios();
44.    }
45.
46.    /**
47.     * Creates data provider instance with search query applied
48.     *
49.     * @param array $params
50.     *
51.     * @return ActiveDataProvider
52.     */
53.}
```

```

46.     public function search($params)
47.     {
48.         $query = Requests::find();
49.         // add conditions that should always apply here
50.
51.         $dataProvider = new ActiveDataProvider([
52.             'query' => $query,
53.         ]);
54.
55.         $dataProvider->setSort([
56.             'attributes' => [
57.                 'name_customer',
58.                 'customer_status_id',
59.                 'name_srw',
60.                 'years_execution_srw',
61.                 'registration_number_srw',
62.                 'nature_work',
63.                 'contact_details_customer',
64.                 'name_purpose_work',
65.                 'description_research',
66.                 'return_samples',
67.                 'research_start_date',
68.                 'research_end_date',
69.                 'date_submission',
70.                 'customerStatusName' => [
71.                     'asc' => ['customer_status.name' => SORT_ASC],
72.                     'desc' => ['customer_status.name' => SORT_DESC],
73.                 ],
74.                 'equipmentName' => [
75.                     'asc' => ['equipment.name' => SORT_ASC],
76.                     'desc' => ['equipment.name' => SORT_DESC],
77.                 ],
78.                 'nameOrganizationName' => [
79.                     'asc' => ['name_organization.name' => SORT_ASC],
80.                     'desc' => ['name_organization.name' => SORT_DESC],
81.                 ],
82.                 'statusName' => [
83.                     'asc' => ['status.name' => SORT_ASC],
84.                     'desc' => ['status.name' => SORT_DESC],
85.                 ],
86.             ]
87.         ]);
88.
89.         $this->load($params);
90.
91.         if (!$this->validate()) {
92.             // uncomment the following line if you do not want to return
             any records when validation fails
93.             // $query->where('0=1');
94.             return $dataProvider;
95.         }
96.
97.         // grid filtering conditions
98.         $query->andWhere([

```

```

99.         'id' => $this->id,
100.         'customer_status_id' => $this->customer_status_id,
101.         'registration_number_srw' => $this-
            >registration_number_srw,
102.         'return_samples' => $this->return_samples,
103.         'research_start_date' => $this->research_start_date,
104.         'research_end_date' => $this->research_end_date,
105.         'date_submission' => $this->date_submission,
106.     ];
107.
108.     $query->andFilterWhere(['like', 'name_customer', $this-
        >name_customer])
109.         ->andFilterWhere(['like', 'name_srw', $this->name_srw])
110.         ->andFilterWhere(['like', 'years_execution_srw', $this-
        >years_execution_srw])
111.         ->andFilterWhere(['like', 'nature_work', $this-
        >nature_work])
112.         -
        >andFilterWhere(['like', 'contact_details_customer', $this-
        >contact_details_customer])
113.         ->andFilterWhere(['like', 'name_purpose_work', $this-
        >name_purpose_work])
114.         ->andFilterWhere(['like', 'description_research', $this-
        >description_research])
115.         -
        >andFilterWhere(['like', 'additional_information', $this-
        >additional_information])
116.         ->joinWith(['customerStatus'])->andFilterWhere(['like',
        CustomerStatus::tableName().'.name', $this->customerStatusName])
117.         ->joinWith(['equipment'])->andFilterWhere(['like',
        Equipment::tableName().'.name', $this->equipmentName])
118.         ->joinWith(['nameOrganization'])->andFilterWhere(['like',
        NameOrganization::tableName().'.name', $this->nameOrganizationName])
119.         ->joinWith(['status'])->andFilterWhere(['like',
        Status::tableName().'.name', $this->statusName]);
120.
121.     return $dataProvider;
122. }
123. }

```

Status.php

```
1.<?php
2.
3.namespace app\models;
4.
5.use Yii;
6.
7./**
8. * This is the model class for table "status".
9. *
10. * @property int $id
11. * @property string $name
12. *
13. * @property Requests[] $requests
14. */
15.class Status extends \yii\db\ActiveRecord
16.{
17.    /**
18.     * {@inheritdoc}
19.     */
20.    public static function tableName()
21.    {
22.        return 'status';
23.    }
24.
25.    /**
26.     * {@inheritdoc}
27.     */
28.    public function rules()
29.    {
30.        return [
31.            [['name'], 'required'],
32.            [['name'], 'string', 'max' => 255],
33.        ];
34.    }
35.
36.    /**
37.     * {@inheritdoc}
38.     */
39.    public function attributeLabels()
40.    {
41.        return [
42.            'id' => Yii::t('app', 'ID'),
43.            'name' => Yii::t('app', 'Name'),
44.        ];
45.    }
46.
47.    /**
48.     * Gets query for [[Requests]].
49.     *
50.     * @return \yii\db\ActiveQuery
51.     */
52.    public function getRequests()
```

```
53.     {
54.         return $this->hasMany(Requests::class, ['status_id' => 'id']);
55.     }
56.
57.     /**
58.      * @return array
59.      */
60.     public static function listMap()
61.     {
62.         return Status::find()->select(['name', 'id'])->indexBy('id')-
            >column();
63.     }
64. }
```

Requests/index.php

```
1.<?php
2.
3.use mdm\admin\components\Helper;
4.use yii\helpers\Html;
5.use kartik\grid\GridView;
6.use yii\widgets\Pjax;
7./* @var $this yii\web\View */
8./* @var $searchModel app\models\RequestsSearch */
9./* @var $dataProvider yii\data\ActiveDataProvider */
10.
11.$this->title = Yii::t('app', 'Requests');
12.$this->params['breadcrumbs'][] = $this->title;
13.$heading = Yii::t('app', 'Requests');
14.$gridColumns = [
15.    ['class' => '\kartik\grid\SerialColumn'],
16.    'statusName',
17.    'nameOrganizationName',
18.    'name_customer',
19.    'customerStatusName',
20.    'name_srw',
21.    'years_execution_srw',
22.    'registration_number_srw',
23.    'nature_work',
24.    'contact_details_customer',
25.    'name_purpose_work',
26.    'description_research',
27.    'return_samples',
28.    'equipmentName',
29.    'research_start_date',
30.    'research_end_date',
31.    'date_submission',
32.    /*TODO: Need fix search date show */
33.    [
34.        'class' => '\kartik\grid\ActionColumn',
35.        'template' =>
36.            Helper::filterActionColumn('{view}{update}{delete}'),
37.    ],
38.];
39.<div class="requests-index" style="overflow-x:hidden;">
40.    <p><?= Html::a(Yii::t('app', 'Create Requests'), ['create'], ['class'
41.        => 'btn btn-success']) ?></p>
42.    <?php Pjax::begin(); ?>
43.    <?php // echo $this->render('_search', ['model' => $searchModel]); ?>
44.
45.    <?= GridView::widget([
46.        'dataProvider' => $dataProvider,
47.        'filterModel' => $searchModel,
48.        'responsive'=>true,
49.        'hover'=>true,
50.        'persistResize'=>true,
```

```

51.         'resizeStorageKey'=>Yii::$app->user->id . '-' . date("m"),
52.         'floatHeader' => true,
53.         'floatHeaderOptions' => [
54.             'scrollingTop' => '0',
55.             'position' => 'absolute',
56.             'top' => 50,
57.         ],
58.         'panel' => [
59.             'type' => GridView::TYPE_PRIMARY,
60.             'heading' => $heading,
61.         ],
62.         'tableOptions'=>['class'=>'table-striped table-bordered table-
condensed'],
63.         'options'=>['style' => 'white-space:nowrap;'],
64.         'rowOptions' => function ($data)
65.         {
66.             switch ($data['statusName']) {
67.                 case 'Новий':
68.                     return ['style' => 'background-color: #84B4DB;'];
69.                     break;
70.                 case 'Прийнятий':
71.                     return ['style' => 'background-color: #896CDF;'];
72.                     break;
73.                 case 'В роботі':
74.                     return ['style' => 'background-color: #FFC16B;'];
75.                     break;
76.                 case 'Завершений':
77.                     return ['style' => 'background-color: #62E96B;'];
78.                     break;
79.                 case 'Відхилений':
80.                     return ['style' => 'background-color: #FF6F6B;'];
81.                     break;
82.                 default:
83.                     return ['style' => 'unset'];
84.             }
85.         },
86.
87.         'columns' => $gridColumns,
88.     ]); ?>
89.
90.     <?php Pjax::end(); ?>
91.
92.</div>

```

requests/view.php

```
1.<?php
2.
3.use yii\helpers\Html;
4.use yii\widgets\DetailView;
5.use mdm\admin\components\Helper;
6.use \yii\web\YiiAsset;
7.
8./* @var $this yii\web\View */
9./* @var $model app\models\Requests */
10.
11.$this->title = $model->id;
12.$this->params['breadcrumbs'][] = ['label' =>
    Yii::t('app', 'Requests'), 'url' => ['index']];
13.$this->params['breadcrumbs'][] = $this->title;
14.YiiAsset::register($this);
15.
16.if (!empty($model->FileName)) {
17.    $downloadFileName = $model->FileName;
18.} else $downloadFileName = '';
19.??>
20.<div class="requests-view">
21.
22.    <h1><?= Html::encode($this->title) ?></h1>
23.
24.    <p>
25.        <?php if (Helper::checkRoute('update')) {
26.            echo Html::a(Yii::t('app', 'Update'), ['update', 'id'
27.=> $model->id], ['class' => 'btn btn-primary',]);
28.        }
29.        if (Helper::checkRoute('delete')) {
30.            echo Html::a(Yii::t('app', 'Delete'), ['delete', 'id'
31.=> $model->id], [
32.                'class' => 'btn btn-danger',
33.                'data-confirm' => Yii::t('app', 'Are you sure to delete
34.this item?'),
35.                'data-method' => 'post',
36.            ]);
37.        } ??>
38.    </p>
39.
40.    <?= DetailView::widget([
41.        'model' => $model,
42.        'attributes' => [
43.            'id',
44.            'nameOrganizationName',
45.            'name_customer',
46.            'customerStatusName',
47.            'name_srw',
48.            'years_execution_srw',
49.            'registration_number_srw',
50.            'nature_work',
51.            'contact_details_customer',
```

```

49.         'name_purpose_work',
50.         'description_research',
51.         'return_samples',
52.         'equipmentName',
53.         'research_start_date',
54.         'research_end_date',
55.         'additional_information:text',
56.         'date_submission',
57.         'statusName',
58.     ],
59. ])?>
60. <div class="container">
61.     <div class="row">
62.         <?php if(!empty($downloadFileName)) foreach ($downloadFileName
as $Name): ?>
63.             <div class="col-md-3">
64.                 <b><?php echo $model->FileNameBase($Name);?></b>
65.                 <?= Html::a('Download', [
66.                     'requests/file-download',
67.                     'filename' => $Name,
68.                 ], [
69.                     'class' => 'btn btn-primary',
70.                     'target' => '_blank',
71.                 ]);
72.                 ?>
73.             </div>
74.             <?php endforeach; ?>
75.         </div>
76.     </div>
77.
78.</div>

```

requests/update.php

```
1.<?php
2.
3.use yii\helpers\Html;
4.
5./* @var $this yii\web\View */
6./* @var $model app\models\Requests */
7.
8.if (!empty($model->FileName)) {
9.    $downloadFileName = $model->FileName;
10.} else $downloadFileName = '';
11.
12.$this->title = Yii::t('app', 'Update Requests: {name}', [
13.    'name' => $model->id,
14.]);
15.$this->params['breadcrumbs'][] = ['label' =>
    Yii::t('app', 'Requests'), 'url' => ['index']];
16.$this->params['breadcrumbs'][] = ['label' => $model->id, 'url' =>
    ['view', 'id' => $model->id]];
17.$this->params['breadcrumbs'][] = Yii::t('app', 'Update');
18.??
19.<div class="requests-update">
20.
21.    <h1><?= Html::encode($this->title) ?></h1>
22.
23.    <?= $this->render('_formUpdate', [
24.        'model' => $model,
25.    ]) ?>
26.    <br><div class="container">
27.        <div class="row">
28.            <?php if(!empty($downloadFileName)) foreach ($downloadFileName
                as $Name): ?>
29.                <div class="col-md-3">
30.                    <b><?php echo $model->FileNameBase($Name); ?></b>
31.                    <?= Html::a('Delete', [
32.                        'requests/file-delete',
33.                        'id' => $model->id,
34.                        'filename' => $Name,
35.                    ], [
36.                        'class' => 'btn btn-primary',
37.                    ]);
38.                    ?>
39.                </div>
40.                <?php endforeach; ?>
41.            </div>
42.        </div>
43.
44.
45.</div>
```

requests/_form.php

```
1.<?php
2.
3.use app\models\CustomerStatus;
4.use app\models\Equipment;
5.use app\models>NameOrganization;
6.use kartik\datetime\DateTimePicker;
7.use yii\helpers\Html;
8.use yii\widgets\ActiveForm;
9.
10./* @var $this yii\web\View */
11./* @var $model app\models\Requests */
12./* @var $form yii\widgets\ActiveForm */
13.??>
14.
15.<div class="requests-form">
16.
17.    <?php $form = ActiveForm::begin(); ?>
18.
19.    <?= $form->field($model, 'name_organization_id')->
        >dropDownList(NameOrganization::listMap()) ?>
20.
21.    <?= $form->field($model, 'name_customer')->textInput(['maxlength' =>
        true]) ?>
22.
23.    <?= $form->field($model, 'customer_status_id')->
        >dropDownList(CustomerStatus::listMap()) ?>
24.
25.    <?= $form->field($model, 'name_srw')->textInput(['maxlength' => true])
        ?>
26.
27.    <?= $form->field($model, 'years_execution_srw')->
        >textInput(['maxlength' => true]) ?>
28.
29.    <?= $form->field($model, 'registration_number_srw')->textInput() ?>
30.
31.    <?= $form->field($model, 'nature_work')->textInput(['maxlength' =>
        true]) ?>
32.
33.    <?= $form->field($model, 'contact_details_customer')->
        >textInput(['maxlength' => true]) ?>
34.
35.    <?= $form->field($model, 'name_purpose_work')->textInput(['maxlength'
        => true]) ?>
36.
37.    <?= $form->field($model, 'description_research')->
        >textInput(['maxlength' => true]) ?>
38.
39.    <?= $form->field($model, 'return_samples')->textInput() ?>
40.
41.    <?= $form->field($model, 'equipment_id')->
        >dropDownList(Equipment::listMap()) ?>
42.
```

```

43.     <?= $form->field($model, 'research_start_date', ['inputOptions' =>
    ['autocomplete' => 'off']])->widget(
44.         DateTimePicker::class, \Yii::$app->params['DateTime']); ?>
45.
46.     <?= $form->field($model, 'research_end_date', ['inputOptions' =>
    ['autocomplete' => 'off']])->widget(
47.         DateTimePicker::class, \Yii::$app->params['DateTime']); ?>
48.
49.     <?= $form->field($model, 'additional_information')->textarea(['rows'
    => 6]) ?>
50.
51.     <?= $form->field($model, 'date_submission', ['inputOptions' =>
    ['autocomplete' => 'off']])->widget(
52.         DateTimePicker::class, \Yii::$app->params['DateTime']); ?>
53.
54. <!--      --><?/*= $form->field($model, 'status_id')-
    >dropDownList(Status::listMap()) */?>
55.
56.     <div class="form-group">
57.         <?= Html::submitButton(Yii::t('app', 'Save'), ['class' => 'btn
    btn-success']) ?>
58.     </div>
59.
60.     <?php ActiveForm::end(); ?>
61.
62.</div>

```

calendar/index.php

```
1.<?php
2.use app\assets\AppAsset;
3.use yii\helpers\Url;
4.
5./* @var $this yii\web\View */
6.
7.$this->title = Yii::$app->name;
8.?>
9.<div class="calendar-index">
10.    <?= yii2fullcalendar\yii2fullcalendar::widget([
11.        'id'=>'calendar',
12.        'options' => [
13.            'lang' => 'uk',
14.            //... more options to be defined here!
15.        ],
16.        'clientOptions' => [
17.            'eventLimit'=> true,
18.            'fixedWeekCount'=>false,
19.            //          Dont click to more
20.            'eventLimitClick'=>new JsExpression('function (cellInfo,
21.                jsEvent) {showEvents(cellInfo.date, cellInfo.segs);}')
22.        ],
23.        'header' => [
24.            'center'=>'',
25.            'left'=>'title',
26.            'right'=>'prev,next today'
27.        ],
28.        'ajaxEvents' => Url::to(['/calendar/json', 'type'=>'all'])
29.    ]);?>
30.</div>
```

```

1.<?php
2.
3.use yii\helpers\Html;
4.use bsadnu\googlecharts\ColumnChart;
5.
6.
7./* @var $this yii\web\View */
8./* @var $searchModel app\models\RequestsSearch */
9.
10.$this->title = Yii::t('app', 'Reporting');
11.$this->params['breadcrumbs'][] = $this->title;
12.$heading = Yii::t('app', 'Reporting');
13.??>
14.<h1>Звіт за тиждень</h1>
15.
16.<?= ColumnChart::widget([
17.    'id' => 'my-column-chart-id',
18.    'data' => $data /*[
19.        ['Day', 'Sales', 'Expenses', 'Var'],
20.        ['1', 1000, 400, 40],
21.        ['2', 1170, 460, 30],
22.        ['3', 660, 1120, 50],
23.        ['4', 1030, 540, 60],
24.        ['5', 1030, 540, 80],
25.        ['6', 1030, 540, 10],
26.        ['7', 1030, 540, 5],
27.    ]*/,
28.    'options' => [
29.        'fontName' => 'Verdana',
30.        'height' => 400,
31.        'fontSize' => 12,
32.        'chartArea' => [
33.            'left' => '5%',
34.            'width' => '90%',
35.            'height' => 350
36.        ],
37.        'tooltip' => [
38.            'textStyle' => [
39.                'fontName' => 'Verdana',
40.                'fontSize' => 13
41.            ]
42.        ],
43.        'vAxis' => [
44.            'title' => 'Кількість замовлень/день тижня',
45.            'titleTextStyle' => [
46.                'fontSize' => 13,
47.                'italic' => false
48.            ],
49.            'gridlines' => [
50.                'color' => '#e5e5e5',
51.                'count' => 10
52.            ],

```

```
53.         'minValue' => 0
54.     ],
55.     'legend' => [
56.         'position' => 'top',
57.         'alignment' => 'center',
58.         'textStyle' => [
59.             'fontSize' => 12
60.         ]
61.     ]
62. ]
63.]) ?>
```