МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет Магістерської та аспірантської підготовки
Кафедра Інформаційних технологій
Рівень вищої освіти магістр
Напрям підготовки 122 Комп'ютерні науки
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри _____
_____
"29" жовтня 2018 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Охрименко Андрію Олексійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи «Автоматизована інформаційна система обліку теплової енергії для КП «Теплопостачання міста Одеси» Частина 1. Розробка інформаційної системи розрахунку споживача теплової енергії
керівники роботи: к.геогр.н., доцент Кузніченко Світлана Дмитрівна, к.т.н. Рихлік Анджей ,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від "05"жовтня 2018 року №271-С
2. Строк подання студентом роботи _____
3. Вихідні дані до роботи технічні вимоги до інформаційної системи, СКБД MySQL, СКБД Firebird, мови програмування PHP JavaScript

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1 Аналіз предметної області
2 Моделювання предметної області
3 Обґрунтування вибору інструментальних засобів
4 Розробка особистого кабінету споживача теплової енергії
4.1 Реалізація процесу реєстрації користувача в системі
4.2 Реалізація особистого кабінету споживача теплової енергії

5. Перелік графічного матеріалу
1 Діаграма прецедентів
2 Структура БД
3 Діаграма класів системи

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання "29"жовтня 2018 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Термін виконання етапів роботи	Оцінка виконання етапу	
			у %	за 4-х бальною шкалою
1	Огляд літературних джерел за темою магістерської роботи			
2	Аналіз предметної області			
3	Проектування інформаційної системи			
4	Обґрунтування вибору інструментальних засобів			
5	Розроблення графічного дизайну			
6	Реалізація особистого кабінету споживача теплової енергії			
7.	**Рубіжна атестація**	19.11.18р.-24.11.18р.		
8	Опис і тестування розробленої системи			
9	Формулювання висновків до магістерської роботи			
11	Здача роботи на кафедру	10.12.18р.		
12	Перевірка на плагіат	19.11.18р.-24.11.18р.		
13	Рецензування	20.10.18р		
	Інтегральна оцінка виконання етапів календарного плану (як середня по етапам)			

Студент _____ Охрименко А.О.
(підпис) (прізвище та ініціали)

Керівники роботи _____ Кузніченко С.Д.
(підпис) (прізвище та ініціали)

_____ Рихлік А.
(підпис) (прізвище та ініціали)

ПОЯСНЮВАЛЬНА ЗАПИСКА

Основною метою комплексної магістерської роботи є створення інформаційного порталу для комунального підприємства «Теплопостачання міста Одеси»

В Одесі, місті з мільйонним населенням, системою централізованого теплопостачання охоплено близько 95% споживачів. Провідним підприємством з виробництва, транспортування та постачання теплової енергії є комунальне підприємство «Теплопостачання міста Одеси», яке надає послуги більш ніж 70% населення міста, вже мало офіційну сторінку в інтернеті дуже давно, але функціонал та дизайн їх сторінки потребував повної перебудови та реконструкції, після початкового огляду було зрозуміло, що створення інформаційного порталу буде з нуля. Зі старого сайту, наш інформаційний портал залишив в собі лише новини.

Проект поділено на 2 частини: перша – створення інформаційного порталу, створення бази даних та налаштування всіх типів користувачів сайту, друга – створення особистого кабінету для користувачів сайту за вимогами замовника, з функціями реєстрації, авторизації, додання та видалення рахунків користувачів послуг КП «ТМО».

Суть першої частини полягає в налаштуванні серверної частини, встановлення необхідного програмного забезпечення на сервер. Також слід провести налаштування системи керування вмістом інформаційного порталу в нашому випадку це – CMS Joomla. Проектування та створення всіх таблиць бази даних для швидкої роботи з вмістом інформації у особистих кабінетах користувачів.

Важливою частиною цієї роботи була оптимізація загрузки бази даних при сильних навантаженнях від великої кількості юзерів. Потрібно було настроїти стандартні засоби оптимізації MySQL та правильно налаштувати процедури які повертають дані з бази даних Oracle.

Суть другої частини полягає в наданні споживачеві інформації про кількість спожитого їм тепла за певні проміжки часу, а також суму яку йому необхідно сплатити у вигляді електронної сторінки.

Для ідентифікації користувача і надання необхідних йому даних буде розроблений персональний особистий кабінет споживача теплової енергії.

Особистий кабінет – це особливий розділ сайту в якому користувач може отримати доступ до особистої інформації, маніпулювати цією інформацією, також здійснювати операції від свого імені.

Обрана мова написання скриптів – PHP.

У ході виконання цієї частини було реалізовано:

- реєстрація, з підтвердженням на електрону пошту користувача;
- авторизація, з можливістю запам’ятати користувача;
- функція відновлення паролю;
- функція додання та видалення особистого рахунку користувача;
- функція друку квитанції для кожного із доданих до особистого кабінету рахунків користувача.

АНОТАЦІЯ

на магістерську роботу «Автоматизована інформаційна система обліку теплової енергії для КП «Теплопостачання міста Одеси»,
частина 1. Розробка інформаційної системи розрахунку споживання теплової енергії
студента Охрименко Андрія Олексійовича

Актуальність дослідження полягає в необхідності надання споживачами теплової енергії зручного он-лайн ресурсу для можливість отримувати інформацію і проводити певні дії зі своїми рахунками за тепло.

Мета дослідження – розробка автоматизованої інформаційної системи обліку теплової енергії для комунального підприємства «Теплопостачання міста Одеси» (КП «ТМО»), а саме розрахунку споживання теплової енергії та розробку інформаційної системи для підприємства КП «ТМО», що має сучасний і зручній інтерфейс та надає користувачу можливість отримувати інформацію про спожите тепло.

Задачі дослідження: провести аналіз предметної області; проектування підсистеми за допомогою UML-засобів; аналіз систем управління базами даних для створення надійної бази даних інформаційного порталу; аналіз сучасних веб-технологій для реалізації основного функціоналу інформаційного порталу; розробити базу даних та основний функціонал інформаційного порталу і забезпечити конфіденційність особистої інформації.

Об'єкт дослідження – інформаційна система обліку теплової енергії.

Предмет дослідження – алгоритмізація і моделювання он-лайн ресурсу для надання споживачам інформації про підприємство КП «ТМО».

Методи дослідження: UML моделювання, об'єктно-орієнтоване моделювання, концептуальне моделювання.

Результати, їх новизна, теоретичне та практичне значення: розроблено нове оригінальне програмне забезпечення, яке реалізує інформаційний портал за вимогами КП «ТМО» з можливістю реєстрації, авторизації і захисту даних.

Структура магістерської роботи складається з вступу, чотирьох розділів, висновків, переліку посилань на 15 найменувань, додатків. Повний обсяг роботи становить 67 сторінки, містить 26 рисунка.

Ключові слова: ІНФОРМАЦІЙНА СИСТЕМА, ОБЛІК ТЕПЛОВОЇ ЕНЕРГІЇ, ОСОБИСТИЙ КАБІНЕТ.

SUMMARY

A Comprehensive Qualification Paper (Thesis) on Computer-Aided Information System for Heat Energy Accounting for the Municipal Enterprise of ‘Heat Supply for the City of Odessa’, part 1. Development of information system for calculation of heat supply energy energy consumption made by a student Okhrymenko Andrii

The urgency of the study is the need to provide the consumer with heat energy a convenient online resource for the opportunity to receive information and carry out certain actions with their accounts for the warmth.

The purpose of the research is to develop an automated information system for heat energy accounting for the utility company "Heat supply of the city of Odesa" (KP "TMO"), namely the calculation of heat energy consumption and the development of an information system for the enterprise KP "TMO", which has modern and user-friendly interface and provides the user with the opportunity to receive information about consuming heat.

Tasks of the research: to carry out the analysis of the subject area; designing a subsystem using UML tools; database management database analysis to create a reliable database of information portals; analysis of modern web technologies for the implementation of the main functionality of the information portal; to develop the database and the main function of the information portals and to ensure the confidentiality of personal information.

The object of research is the information system of heat energy accounting.

The subject of the study is the algorithmization and modeling of on-line resources to provide consumers with information about the enterprise of the CM "TMO".

Research methods: UML modeling, object-oriented modeling, conceptual modeling.

Results, their novelty, theoretical and practical significance: the new original software was developed, which implements the information portal according to the requirements of the "TMO" CP with the possibility of registration, authorization and data protection.

The structure of the master's work consists of an introduction, four roles, conclusions, a list of references to 15 titles, applications. Full work is 67 pages, contains 26 pictures.

Key words: INFORMATION SYSTEM, HEAT SUPPLY ENERGY ACCOUNTING, PERSONAL CABINET.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ

БД – база даних

КП «ТМО» – комунальне підприємство «Теплопостачання міста Одеси»

СКБД – система керування базами даних

CSS – Cascading Style Sheets

JS – JavaScript

ВСТУП

В наш час у всьому світі не має будинку в якому не було би лічільника теплової енергії. Для спрощення слідкування та оплати комунальних послуг користувачів КП «ТМО», яка включає в себе дуже багато різноманітних послуг був створений у ході магістерської роботи інформаційний портал.

Інтернет-портал (або портал, інформаційний портал) – сукупність поєднаних безпосередньо та через мережу "Інтернет" апаратних засобів, що включають комп'ютери та машинозчитувані електронні носії інформації із заздалегідь записаною на них інформацією та/або виконані з можливістю запису та зчитування інформації у вигляді комп'ютерних програм, баз даних і т.п., виконана з можливістю обробки зазначеної інформації та команд користувача системи та надання йому Інтернет-сервісів як результатів обробки зазначеної інформації і команд.

Інтернет-портал для користувачів – сайт, що надає користувачеві Інтернету різні інтерактивні сервіси (Інтернет-сервіси), які працюють в рамках єдиного сайту.

Мета роботи – розробити систему для підприємства КП «ТМО» за останніми технологіями HTML5 та CSS3, вивчити основні принципи взаємодії сайту з БД на скриптовій мові програмування PHP, спроектувати базу даних для записів КП «ТМО», на прикладі якої продемонструвати можливості MySQL, а також ознайомитися с принципом роботи JavaScript та JQuery.

Основними задачами магістерської роботи є:

Поставлені задачі:

- створити інформаційний потрал для користувачів послуг КП «ТМО»;
- створити бази даних;
- створити інтерфейс для роботи користувачів з сайтом;
- проаналізувати предметне середовище;
- навчитися створювати SQL запити на вибірку, вставку, редагування даних безпосередньо з інтерфейсу сайта;
- заповнити сторінки інформаційного порталу, додавши інформацію про місцезнаходження підприємства КП «ТМО», сторінку новин і всі інші сторінки;
- зробити аккаунти адміністраторів на інформаційному порталі.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис підприємства КП «ТМО».

Історія централізованої системи теплопостачання (ЦСТ) в місті Одесі налічує вже понад 60 років. Поява ЦСТ в місті з мільйонним населенням було справою часу. Прийшла пора позбавлятися від пічного опалення з його енергетичної неефективністю і незручністю для мешканців, коли кожна квартира мала свій сарай з дровами і вугіллям, запас яких необхідно було поповнювати щорічно.

У місті настав час масового будівництва багатоповерхових жи-лих будинків. При цьому теплопостачання 5-ти і 9-ти поверхових будинків можна обидві-печити тільки домовик системою водяного опалення. До того ж наспів введення в експлуатацію Одеської ТЕЦ, в 1950 році – перша черга, а в 1953 році – друга черга. На базі централізованого джерела теплоенергії – Одеської ТЕЦ – і почалася теплофікація міста.

1.2 Технічна характеристика обладнання

System:

- Host: teplo.od.ua Kernel – 4.4.0-139-generic x86_64 (64 bit);
- Console: tty 0 Distro – Ubuntu 16.04 xenial;
  Machine;
- System: QEMU product – Standard PC (i440FX + PIIX 1996) v: pc-i440fx-2.7;
- Mobo: N/A model – N/A Bios: Sea v: rel-1.9.3-0-ge2fc41e-prebuilt.qemu-project.org date: 04/01/2014;
- CPU: Single core Common KVM (-UP-) cache – 4096 KB speed – 2899 MHz (max).

Graphics:

- Card: Red Hat QXL paravirtual graphic card;
- Display Server:N/A driver: N/A tty size: 237x56 Advanced Data: N/A out of X;
- Network;
- Card – Intel 82540EM Gigabit Ethernet Controller driver: e1000;
- IF – ens18 state: up speed: 1000 Mbps duplex: full mac: 92:c3:4a:de:6c:14.

Drives:

- HDD Total Size: 68.7GB (24.2% used);
- ID-1: /dev/sda model: QEMU_HARDDISK size: 68.7GB.

Partition:

- ID-1: size: 62G used: 15G (25%) fs: ext4 dev: /dev/sda1;
- ID-2: swap-1 size: 1.07GB used: 0.24GB (23%) fs: swap dev: /dev/sda5;
- RAID: No RAID devices: /proc/mdstat, md_mod kernel module present;
- Info: Processes: 126 Uptime: 9 days Memory: 347.7/992.1MB Init: systemd runlevel: 5
- Client: Shell (bash) inxi: 2.2.35

Інформація по серверу. На сервере установлен:

- Пакет: apache2;
- Версия: 2.4.18-2ubuntu3.9;
- PHP: Version 7.0.32-0ubuntu0.16.04.1.

phpinfo:

- System: Linux teplo.od.ua 4.4.0-139-generic #165-Ubuntu SMP Wed Oct 24 10:58:50 UTC 2018 x86_64;
- Server API: Apache 2.0 Handler;
- PHP API: 20151012;
- PHP Extension: 20151012;
- Zend Extension: 320151012;
- Zend Extension Build: API320151012,NTS;
- PHP Extension Build: API20151012,NTS;
- Debug Build: no;
- Thread Safety: disabled;
- Zend Signal Handling: disabled;
- Zend Memory Manager: enabled;
- Zend Multibyte Support: disabled;
- IPv6 Support: enabled;
- DTrace Support: available, disabled;
- Registered PHP Streams: https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar;
- Registered Stream Socket Transports: tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2;
- Registered Stream Filters: zlib.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk, convert.iconv.

Configuration:

- Apache Version: Apache/2.4.18 (Ubuntu);
- Apache API Version: 20120211;
- Server Administrator: d.ilikhin@teplo.od.ua;
- Hostname:Port: teplo.od.ua:0;
- User/Group: www-data(33)/33;
- Max Requests: Per Child: 0 – Keep Alive: on - Max Per Connection: 100;
- Timeouts: Connection: 300 – Keep-Alive: 5;
- Virtual Server: Yes;
- Server Root: /etc/apache2.

Loaded Modules:

Core, mod_so, mod_watchdog, http_core, mod_log_config, mod_logio, mod_version, mod_unixd, mod_access_compat, mod_alias, mod_auth_basic, mod_authn_core, mod_authn_file, mod_authz_core, mod_authz_host, mod_authz_user, mod_autoindex, mod_deflate, mod_dir, mod_env, mod_filter, mod_mime, prefork mod_negotiation, mod_php7, mod_rewrite, mod_setenvif, mod_status.

Mysqli:

- MysqlI Support: enabled;
- Client API library version: mysqlnd 5.0.12-dev.

Session:

- Session Support: enabled;
- Registered save handlers: files user;
- Registered serializer handlers: php_serialize php php_binary wddx.

2 МОДЕЛЮВАННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

2.1 Вибір типу сайту для організації

Залежно від того, з якою метою створюється сайт, потрібно правильно вибрати тип сайту.

1) Інформаційний сайт зручний при наданні потрібних даних для клієнтів. Інфосайт може вирішувати різні завдання. Наприклад, громадська організація може використовувати такий ресурс для розміщення інформації про діяльність об'єднання. Комерційна компанія може вести свій особистий блог і розповідати про свою діяльність. Простий обиватель може створити такий сайт, щоб ділитися своїми думками, порадами з іншими користувачами мережі і, до того ж, заробляти при цьому. Контент на інформаційному сайті повинен бути актуальним. Його потрібно постійно оновлювати, доповнювати і стежити за якістю. Це вимагає часу, сил і грошей.

2) Корпоративний сайт – містить повну інформацію про компанії-власника, послуги / продукцію, події в житті компанії. Відрізняється від сайту-візитки і представницького сайту повнотою наданої інформації, часто містить різні функціональні інструменти для роботи з контентом (пошук і фільтри, календарі подій, фотогалереї, корпоративні блоги, форуми). Може бути інтегрований з внутрішніми інформаційними системами компанії-власника (КІС, CRM, бухгалтерськими системами). Може містити закриті розділи для тих чи інших груп користувачів – співробітників, дилерів, контрагентів тощо.

3) Корпоративний сайт – це висококласний ресурс компанії, який може використовуватися для будь-яких комерційних та інформаційних цілей:

а) інформування ваших партнерів і клієнтів про діяльність компанії;

б) демонстрації портфоліо і асортименту товарів;

в) повноцінного інтернет-просування по різних каналах;

г) активного залучення нових клієнтів;

д) створення унікального інформаційного ресурсу, який стосується вашої сфери.

Комерційним сайтом прийнято називати веб-ресурс, призначений для просування своїх товарів або послуг, а також, залучення клієнтів і бізнес партнерів.

Традиційно, комерційний сайт повинен включати модулі для розміщення:

а) новостей фірми;

б) каталогу продукції або вітрини магазину;

в) контактної інформації;

г) формою зворотного зв'язку.

Крім обов'язкової інформації, в залежності від профілю компанії, на сайті розміщуються додаткові модулі (калькулятор послуг, онлайн консультант і т.п.), блог статей і будь-яку іншу інформацію для досягнення основної мети комерційного сайту, отримання прибутку [1].

З усього вище сказаного, було вирішено створити комерційний сайт. Він зручний і практичний для організації. Сайт буде в собі містити 6 модулів: головна сторінка, новини, співробітництво, тарифи, про підприємство та особистий кабінет користувача. Таким чином, на головній сторінці будуть відображені-ни: інформація про підприємство, контакні дані підприємства, останні новини, підтримка, програми, форуми та семінари пов'язані з діяльно-стю підприємства, google-карта з адресами відділень для користувачів послугами підприємства і пошук інформації по сайту. В категорії новини будуть показані останні новини підприємства і його діяльність. У контактах буде вказано телефони, адреси офісів підприємства для громадян. Форма зворотнього зв'язку для питань також актуальна. Сайт буде працювати для всіх жителів міста, перегляду вартості їх послуг і в подальшому, оплати цих послуг, а також для інформування користувачів про послуги, які надає підприємство.

Адміністратору сайту доступний весь його функціонал, а саме функції:

- додавання новин;
- моніторинг всіх користувачів зареєстрованих на сайті;
- оновлення контенту на статичних сторінках сайту;
- оновлення інформації про плани на закупівлі підприємства;
- оновлення інформації про зміну тарифів на послуги підприємства;
- зміна / додавання даних які бачать кінцеві користувачі в своїх особистих кабінетах;
- зміна / додавання інформації про головних керуючих компанії, про статут підприємства, про структуру підприємства;
- зміна інформаці про фінансову звітність підприємства;
- редагування контактних даних.

Функції які доступні для користувачів сайту які на ньому не зареєстровані:

- перегляд всієї інформації про підприємство, такі як: плани на закупівлю, тарифи на послуги підприємства, інформмації про головних

керуючих компанії, про статут підприємства, про структуру підприємства, перегляд головної сторінки і контактної інформації;

– можливість перейти на сторінку реєстрації, де надалі пройти всі пункти і отримати свій аккаунт на сайті.

Функції які доступні для користувачів сайту які на ньому зареєстровані:

– особистий кабінет, в якому користувач може додати потрібну йому кількість особових рахунків, за номером особового рахунку і фамілії на яку оформлений даний особовий рахунок;
– після додавання особового рахунку, користувач буде бачити всю інформацію про нього: адреса, якщо у цього особового рахунку є якісь пільги, то буде виведена інформація про пільги, далі по-описаних вище буде доступна таблиця з даними його особового рахунку, де помісячно буде виводиться: кількість прописаних осіб, площа, оплата за цей місяць, чи була оплачена сума яка показується в попередніх поле, матеріальна допомога, компенсація, субсидії і борг;
– нижче знаходяться кнопки, які можуть вивести таблицю Опис вище за будь-попередній рік, починаючи з 2006.

Все це зображено на рис. 2.1, за допомогою UML-діаграми прецендентів [2].

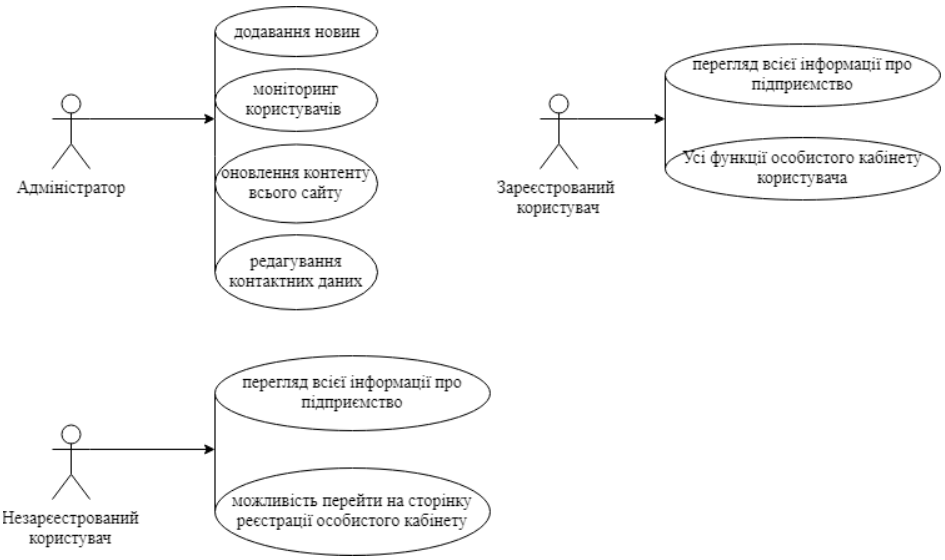


Рисунок 2.1 – UML діаграма прецедентів

Таблиці ядра бази даних Joomla 3 містить 68 таблиць (рис.2.2). База даних підтримує основні функції сайту на стороні front-end і back-end сайту. У таблицях бази зберігається динамічний контент сайту, який часто змінюється і оновлюється[3]. Адреса бази даних, як правило, localhost, а може і бути конкретна адреса, типу, логін.ідентифікатор.mysql.

Адміністратор сайту і група дозволених користувачів мають доступ до таблиць ядра бази даних через додаток phpMyAdmin, встановлене на сервері хостингу. Для входу в phpMyAdmin використовується ім'я користувача БД і пароль доступу, які задаються при створенні бази даних.

Приложение	Подчиненное приложение	Таблица базы
Материалы	Менеджер материалов	#_content -Материалы #_content_rating – Материалыврейтинге #_contentitem_tag_map
	Менеджер разделов	#_categories
	Менеджер Главной страницы	#_content_frontpage
Расширения	Компоненты	#_extensions
	Менеджер шаблонов	Не существует таблицы в базые данных, в которой перечисляются доступные шаблоны. Список формируется по содержимому каталога templates.
	Менеджер языков	#_languages
Компоненты	Баннеры	#_banners -Все баннеры сайта. #_banner_clients – Размещенные баннеры. #_banner_tracks – Каналы банеров.
	Ленты новостей	#_newsfeeds
	Интелектуальный поиск	Таблицы #_finder_
	Поиск	#_core_log_searches
	Сообщения	#_messages #_messages_cfg
	Общие настройки	Общие настройки сайта хранятся в файле configuration.php. Для общих настроек нет таблицы в базе данных. ДанныеБДв configuration.php: var $host = ''; var $user = ''; var $password = '';
	Журнал регистрации и статистика	#_core_log_searches
Меню		#_menu - Всепунктыменюсайта (back-end + front-end). #_menu_types – Менюback-end. #_associations – Соответствие пунктов меню языковой локализации.
UCM	Unified Content Model Единаямодельсодержимого	#_ucm_base #_ucm_content #_ucm_history

Рисунок 2.2 – Таблиці ядра Joomla 3.

Структура БД Joomla зображена на рис. 2.3.

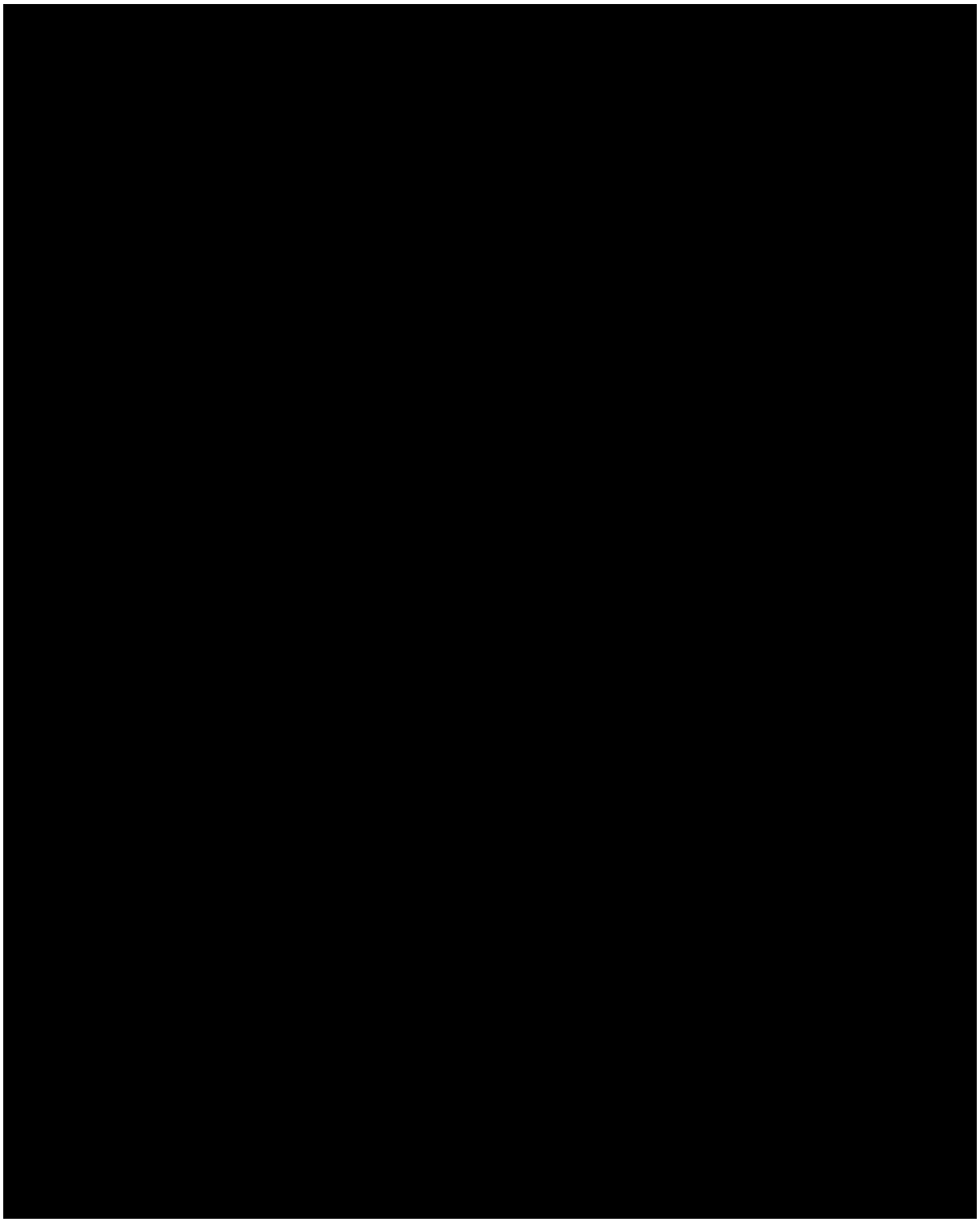


Рисунок 2.3 – Структура БД Joomla 3.

Діаграма класів, які використовуються виключно для організації роботи особистого кабінету користувічів зображена на рис. 2.4.

Таблиця users зберігає в собі всі дані про всіх користувачів інформаційного порталу (ІП) такі як ім'я, прізвище, логін користувача на ІП, email, пароль (у вигляді хєша), стан блокування користувача на ІП, чи був відправлений email з підтвердженням користувачеві, дата реєстрації користувача,

дата останнього візиту користувача на ІП, підтвердив він свій email, поле параметри тільки для адмінів (туди зберігаються настройки аккаунта кожного з адмінів), дата останнього скидання користувачем пароля.

Таблиця usergroups описує різні типи користувачів на інформаційному порталі: адміністратор, редактор новин, редактор тендерів, редактор тарифів, зареєстровані користувачі.

Таблиця user_usergroup_map зберігає в собі id користувача з таблиці users і id його групи з таблиці usergroups.

Таблиця session зберігає в собі всі активні сесії користувачів та ВП.

Таблиця user_notes зберігає в собі всякі замітки, інформація про користувача, яку додав один з адміністраторів сайту.

Таблиця user_keys зберігає в собі ключі cookie.

Таблиця user_profiles зберігає в собі додаткову інформацію яку користувач міг ввести в необов'язкові поля, після реєстрації, такі поля як: адреса, місто, поштовий код, телефон, веб сайт.

Таблиця user_acc зберігає в собі: код користувача, прізвище користувача, і id аккаунта до якого прив'язаний рахунок цього користувача, це таблиця потрібна для того щоб при вході в особистий кабінет отримувати з неї дані які картки повинні бути створені в даному особистому кабінеті.

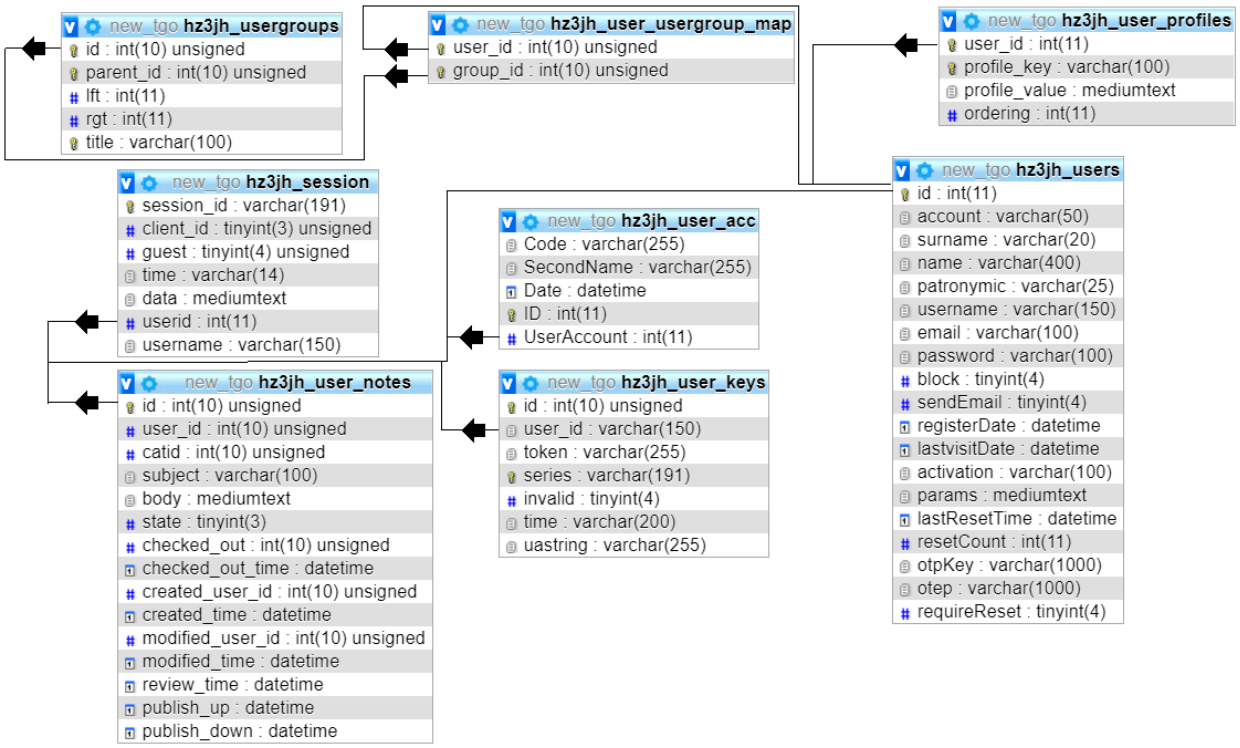


Рисунок 2.4 – Діаграма класів

Діаграма класів, які будуть використовуватися виключно для особистого кабінету користувачів зображена на рис. 2.5.

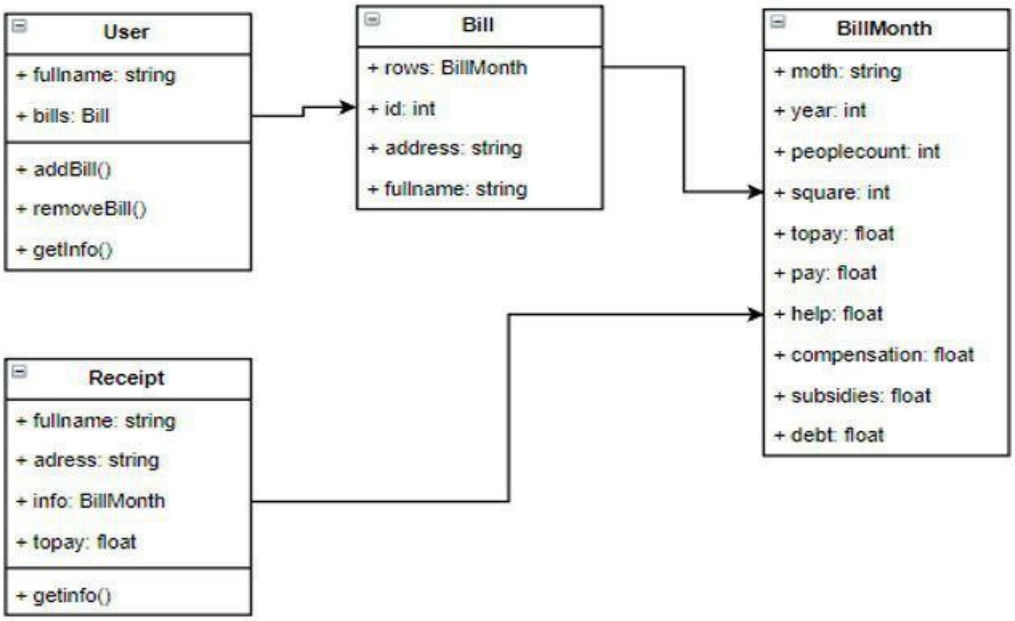


Рисунок 2.5 – Діаграма класів

3 ВИБІР ПРОГРАММНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Скриптова мова програмування (PHP)

PHP (англ. PHP: Hypertext Preprocessor – PHP: гіпертекстовий препроцесор), попередня назва: Personal Home Page Tools – скриптова мова програмування, була створена для генерації HTML-сторінок на стороні веб-сервера. PHP є однією з найпоширеніших мов, що використовуються у сфері веб-розробок (разом із Java, .NET, Perl, Python, Ruby). PHP підтримується переважною більшістю хостинг-провайдерів. PHP – проект відкритого програмного забезпечення.

PHP інтерпретується веб-сервером у HTML-код, який передається на сторону клієнта. На відміну від скриптової мови JavaScript, користувач не бачить PHP-коду, бо браузер отримує готовий html-код. Це є перевага з точки зору безпеки, але погіршує інтерактивність сторінок. Але ніхто не забороняє використовувати PHP для генерування JavaScript-кодів, які виконуються вже на стороні клієнта.

PHP – мова, код якої можна вбудовувати безпосередньо html-код сторінок, які у свою чергу, будуть коректно оброблені PHP-інтерпретатором. Обробник PHP просто починає виконувати код після відкриваючого тегу (<?php) і продовжує виконання до того моменту, поки не зустріне закриваючий тег (?>).

Велика різноманітність функцій PHP дає можливість уникати написання багаторядкових функцій, призначених для користувача, як це відбувається в C або Pascal.

Наявність інтерфейсів до багатьох баз даних в PHP вбудовані бібліотеки для роботи з MySQL, PostgreSQL, mSQL, Oracle, dbm, Hyperware, Informix, InterBase, Sybase, через стандарт відкритого інтерфейсу зв'язку з базами даних (Open Database Connectivity Standard – ODBC) можна підключатися до всіх баз даних, до яких існує драйвер.

3.1.1 Традиційність PHP

Мова PHP здаватиметься знайомою програмістам, що працюють в різних областях. Багато конструкцій мови запозичені з C, Perl. Код PHP дуже схожий на той, який зустрічається в типових програмах на C або Pascal. Це помітно знижує початкові зусилля при вивченні PHP. PHP – мова, що поєднує переваги Perl та C і спеціально спрямована на роботу в Інтернеті, мова з універсальним і зрозумілим синтаксисом. І хоча PHP є досить

молодою мовою, вона здобула таку популярність серед web-програмістів, що в наш час є найпопулярнішою мовою для створення веб-застосунків (скриптів).

3.1.2 Наявність сирцевого коду та безкоштовність програмного забезпечення

Стратегія Open Source, і розповсюдження початкових текстів програм в масах, безсумнівно справили благотворний вплив на багато проектів, в першу чергу – Linux хоч і успіх проекту Apache сильно підкріпив позиції прихильників Open Source. Сказане відноситься і до історії створення PHP, оскільки підтримка користувачів зі всього світу виявилася дуже важливим чинником в розвитку проекту PHP.

Ухвалення стратегії Open Source і безкоштовне розповсюдження початкових текстів PHP надало неоціненну послугу користувачам. Окрім цього, користувачі PHP в усьому світі є свого роду колективною службою підтримки, і в популярних електронних конференціях можна знайти відповіді, навіть на найскладніші питання.

3.1.3 Ефективність PHP

Ефективність є дуже важливим чинником у програмуванні для середовищ розрахованих на багато користувачів, до яких належить і web. Важливою перевагою PHP є те, що ця мова належить до інтерпретованих. Це дозволяє обробляти сценарії з достатньо високою швидкістю. За деякими оцінками, більшість PHP-сценаріїв (особливо не дуже великих розмірів) обробляються швидше за аналогічні їм програми, написані на Perl. Проте хоч би що робили розробники PHP, виконавчі файли, отримані за допомогою компіляції, працюватимуть значно швидше – в десятки, а іноді і в сотні разів. Але продуктивність PHP достатня для створення цілком серйозних веб-застосунків.

3.1.4 Типи даних в PHP

До базових типів належать булеві дані, цілі та дійсні числа із плаваючою комою, а також рядки. Булеві дані виражають істинність значення. Цілі числа можуть бути подані у вісімковому, десятковому та шістнадцятковому вигляді. Розмір цілого числа може змінюватись залежно від платформи, зазвичай, розрядність становить 32 біти. PHP не підтримує беззнакові цілі числа. Дійсні числа із плаваючою комою можуть бути подані в десятковій

або експоненційній формі. Для кожної змінної можна надати власний тип даних. Для цього існує декілька видів:

Рядки ділять на два класи: рядки, що підлягають аналізу, та ті, що не підлягають. Перший клас досліджується інтерпретатором на наявність посилань на інші змінні, і за умови їхньої наявності робиться підстановка значень у відповідне місце. Крім того, клас дозволяє проводити маніпуляції з керівними символами. Символ рядка може мати лише одне з 256 значень, але є можливість працювати з багатобайтовими символами. Доступ до символів рядка можливий з використанням синтаксису, схожого на доступ до елементів масивів.

PHP надає широкий спектр функцій для пошуку та заміни тексту в рядках. Для цього використовують як традиційний підхід, так і спеціальний, що базується на використанні регулярних виразів. При цьому в мові реалізована підтримка двох видів регулярних виразів: Perl-сумісні та POSIX-сумісні, що розрізняються за синтаксисом та особливостями роботи.

3.1.5 Змішані типи даних в PHP

До змішаних типів належать масиви, хеші та об'єкти.

Масиви в сенсі мови є наборами змінних, що згруповані в єдину змінну. Вимога однотипності наповнення масивів не ставиться. Технічно, масиви – це впорядковані карти, що відображають ключові значення на позиції змінних даних. Вмістом значення, на яке вказує ключ може бути будь-чим, що можна подати у вигляді змінної. Не існує жодних обмежень, крім обсягу пам'яті, що накладаються на кількість ключів масиву.

Особливістю мови є відмова від рівномірного розподілу ключів масивів. Реалізовано і модель багатовимірних масивів, причому без явного обмеження глибини вкладеності. Корисною властивістю PHP є можливість асоціації масивів із функцією зворотного виклику. Ці функції дозволяють проводити дії над одним чи кількома масивами в пакетному режимі. PHP наділений великою кількістю функцій роботи з масивами. В PHP можна по-різному оголошувати масиви:

Також, існують два спеціальні типи даних – ресурс та NULL.

Ресурс — спеціальна змінна, що містить посилання на зовнішній ресурс. Ресурси створюються та використовуються в спеціалізованих функціях. Оскільки тип містить спеціальні вказівники на відкриті файли, під'єднання (англ. include) та інше, то будь-які дії над значенням ресурсу не мають сенсу.

Область видимості змінної – середовище, в якій вона визначена. Розрізняють локальні та глобальні змінні. За замовчуванням, всі змінні мають локальний характер дії. Виділяють особливий тип змінних – статичні, що дозволяє повторне звернення до змінної в певному сегменті коду, причому змінна буде зберігати попередньо отримане значення. Існує також поняття суперглобальних змінних, які є місцем збереження даних оточення або даних, отриманих ззовні. Підтримується концепція динамічних змінних та функцій.

Константи в PHP – ідентифікатори простих значень. Можливе визначення константи, причому після її оголошення стає неможливою зміна її значення чи анулювання. Константи можуть мати лише скалярні значення. Підтримується можливість отримання значення константи за динамічним ім'ям. Область видимості констант буде глобальною для сценарію та всіх під'єднаних компонентів. Також в ядрі мови визначено чимало системних констант.

3.1.6 Конструкції мови PHP

Оператори в сенсі мови дозволяють виконувати відповідну дію над одним чи кількома операндами.

Оператори бувають трьох типів – унарні, бінарні та тернарні. Оператори, як і в інших мовах характеризуються не лише дією, а й асоціативністю та пріоритетністю. Особливістю булевих операцій порівняння – розрізнення двох класів – з врахуванням типу і без врахування типу, при якому відбувається приведення до відповідного типу. Округлення відбуваються завжди в меншу сторону. В мові реалізовані особливі класи операторів – виконання, управління помилками та перевірки приналежності до класу.

Функції в сенсі мови є контейнерами коду, причому можливе включення інших функцій та класів. На цьому і базується можливість умовного визначення функції. В цьому випадку висувається вимога попередньої декларації викликаної функції, що не обов'язкове в інших випадках. Можливості перевизначення чи деактивації функції не існує. Результат, який повертає функція може мати будь-який тип.

В мові реалізована функціональність посилань. Можливо створити необмежену кількість псевдонімів, що посилаються на єдиний сегмент даних. При вивільненні будь-якого з псевдонімів, сегмент даних залишається в пам'яті до моменту завершення сценарію або вивільнення усіх посилань.

Що стосується функцій в PHP, то замість прийнятого в багатьох мовах принципу перевантаження функцій, що дозволяє змінити хід виконання певної функції в залежності від типу та кількості переданих параметрів, використовується метод динамічних аргументів. Це дає змогу не визначати кількість параметрів для функцій при їх оголошенні, а працювати із тими аргументами, які були отримані на момент виклику функції. У тілі функції можливо отримати кількість переданих їй аргументів і проводити відповідні маніпуляції. При оголошенні функції звичайним чином, можливе задання значень аргументів за замовчуванням. Функції можуть повертати лише одне значення, проте це обмеження можна оминути, використавши не лише масиви, а й посилання. Передача аргументів за посиланням неможлива під час виконання та оголошення функції.

Після виконання сценаріїв, простір пам'яті, займаної ними очищується збирачем сміття. Проте, за потреби можливе власноруч кероване виконання очищення пам'яті від надлишкових сегментів даних під час виконання скриптів, але використання функцій очищення пам'яті є невиправданим, хоча така можливість існує.

Для побудови програмних комплексів можна використовувати модульний підхід, виконуючи розділення різнорідного коду. При потребі, можливе виконання під'єднання необхідних модулів, причому операція виконання може бути і умовною. Під'єднані до скрипта файли можуть повертати значення[4].

3.2 Переваги PHP

Головним чинником мови PHP є практичність. PHP повинен надати програмісту засоби для швидкого та ефективного вирішення поставлених завдань. Практичний характер PHP обумовлений п'ятьма важливими характеристиками:

- традиційністю;
- простотою;
- ефективністю;
- безпекою;
- гнучкістю.

Існує ще одна «характеристика», яка робить PHP особливо привабливим: він розповсюджується безкоштовно. Причому, з відкритими вихідними кодами (Open Source).

3.2.1 Традиційність PHP

Мова PHP здаватиметься знайомим програмістам, що працюють в різних областях. Багато конструкцій мови запозичені з Сі , Perl.

Код PHP дуже схожий на той , який зустрічається в типових програмах на С або Pascal . Це помітно знижує початкові зусилля при вивченні PHP. PHP – мова, що поєднує переваги Perl і Сі і спеціально націлений на роботу в Інтернеті, мова з універсальним (правда, за деякими застереженнями) і ясним синтаксисом.

І хоча PHP є досить молодим мовою, він знайшов таку популярність серед web-програмістів, що на даний момент є мало не найпопулярнішою мовою для створення web-додатків (скриптів).

3.2.2 Простота PHP

Сценарій PHP може складатися з 10 000 рядків або з одного рядка - все залежить від специфіки вашого завдання. Вам не доведеться довантажувати бібліотеки, вказувати спеціальні параметри компіляції або що-небудь в цьому роді. Механізм PHP просто починає виконувати код після першої екрануючої послідовності. Якщо код має правильний синтаксис, він виконується в точності так, як вказав програміст.

PHP – мова, яка може бути вбудований безпосередньо в html – код сторінок, які, в свою чергу будуть коректно оброблятися PHP – інтерпретатором. Ми можемо використовувати PHP для написання CGI – сценаріїв і позбутися від безлічі незручних операторів виведення тексту. Ми можемо залучати PHP для формування HTML – документів, позбувшись від безлічі викликів зовнішніх сценаріїв.

Велика розмаїтість функцій PHP позбавлять вас від написання багаторядкових призначених для користувача функцій на С або Pascal.

3.2.3 Ефективність PHP

Ефективність є виключно важливим чинником при програмуванні для багатокористувацьких середовищ, до числа яких належить і web.

Дуже важлива перевага PHP полягає в його «движку». «Движок» PHP не є ні компілятором, ні інтерпретатором. Він є транслює інтерпретатором. Такий пристрій «движка» PHP дозволяє обробляти сценарії з достатньо високою швидкістю.

За деякими оцінками, більшість PHP – сценаріїв (особливо не дуже великих розмірів) обробляються швидше за аналогічні їм програм, написаних

на Perl. Однак, щоб не робили розробники PHP, відкомпілювалися виконувані файли будуть працювати значно швидше в десятки, а іноді і в сотні разів. Але продуктивність PHP цілком достатня для створення цілком серйозних web -додатків.

3.2.4 Безпека PHP

PHP надає в розпорядження розробників і адміністраторів гнучкі та ефективні засоби безпеки, які умовно діляться на дві категорії: засоби системного рівня та засоби рівня додатки.

1) Засоби безпеки системного рівня:

У PHP реалізовані механізми безпеки, що знаходяться під управлінням адміністраторів; при правильному налаштуванні PHP це забезпечує максимальну свободу дій і безпеку. PHP може працювати в так званому безпечному режимі (safe mode), який обмежує можливості застосування PHP користувачами по ряду важливих показників. Наприклад, можна обмежити максимальний час виконання і використання пам'яті (неконтрольований витрата пам'яті негативно впливає на швидкодію сервера). За аналогією з cgi – bin адміністратор також може встановлювати обмеження на каталоги, в яких користувач може переглядати і виконувати сценарії PHP, а також використовувати сценарії PHP для перегляда конфіденційної інформації на сервері (наприклад, файлу passwd).

2) Засоби безпеки рівня додатки:

У стандартний набір функцій PHP входить ряд надійних механізмів шифрування. PHP також сумісний з багатьма додатками незалежних фірм, що дозволяє легко інтегрувати його з захищеними технологіями електронної комерції (e-commerce). Інша перевага полягає в тому, що вихідний текст сценаріїв PHP не можна переглянути в браузері, оскільки сценарій компілюється до його відправлення за запитом користувача. Реалізація PHP на стороні сервера запобігає викрадення нетривіальних сценаріїв користувачами, знань яких вистачає хоча б для виконання команди View Source.

3.2.5 Гнучкість PHP

Оскільки PHP є вбудовуваним (embedded) мовою, він відрізняється винятковою гнучкістю по відношенню до потреб розробника. Хоча PHP зазвичай рекомендується використовувати в поєднанні з HTML, він з таким же успіхом інтегрується і в javascript, WML, XML та інші мови. Крім того,

добре структуровані програми PHP легко розширюються в міру необхідності (втім, це відноситься до всіх основних мов програмування).

Немає проблем і з залежністю від браузерів, оскільки перед відправкою клієнту сценарії PHP повністю компілюються на стороні сервера. По суті, сценарії PHP можуть передаватися будь-яких пристроїв з браузерами, включаючи стільникові телефони, електронні записні книжки, пейджери і портативні комп'ютери, не кажучи вже про традиційні ПК. Програмісти, що займаються допоміжними утилітами, можуть запускати PHP в режимі командного рядка.

Оскільки PHP не містить коду, орієнтованого на конкретний web-сервер, користувачі не обмежуються певними серверами (можливо, незнайомими для них). Apache, Microsoft IIS, Netscape Enterprise Server, Stronghold і Zeus - PHP працює на всіх перерахованих серверах. Оскільки ці сервери працюють на різних платформах, PHP в цілому є платформенно – незалежним мовою й існує на таких платформах, як UNIX, Solaris, FreeBSD та Windows 95/98/NT/2000/XP/2003.

Нарешті, засоби PHP дозволяють програмісту працювати із зовнішніми компонентами, такими як Enterprise Java Beans або COM -об'єкти Win32. Завдяки цим новим можливостям PHP займає гідне місце серед сучасних технологій і забезпечує масштабування проектів до необхідних меж.

3.2.6 Безкоштовне розповсюдження PHP

Стратегія Open Source, і розповсюдження початкових текстів програм в масах, зробило безсумнівно благотворний вплив на багато проектів, в перш чергу – Linux, хоча й успіх проекту Apache сильно підкріпив позиції прихильників Open Source. Сказане відноситься і до історії створення PHP, оскільки підтримка користувачів зі всього світу виявилася дуже важливим чинником в розвитку проекту PHP.

Прийняття стратегії Open Source і безкоштовне розповсюдження початкових текстів PHP надало неоціниму послугу користувачам. До того ж, чуйне співтовариство користувачів PHP є свого роду «колективної службою підтримки», і в популярних електронних конференціях можна знайти відповіді навіть на найскладніші питання [4].

3.3 Динамічна об'єткно-орієнтована мова програмування (JS)

Javascript – це мова програмування, за допомогою якого веб-сторінок надається інтерактивність. З його допомогою створюються додатки, які

включаються в HTML-код (наприклад, анкети або форми реєстрації, які заповнюються користувачем). Часто Javascript плутають з мовою програмування Java, проте спільного між ними дуже мало. До того ж, деякі порівнюють Javascript з мовами Python, Self, Ruby. Однак це особлива мова, який існує сам по собі[5].

Використання: За допомогою Javascript можна змінювати сторінку, змінювати стилі елементів, видаляти або додавати теги. З його допомогою можна дізнатися про будь-яких маніпуляціях користувача на сторінці (прокрутка сторінки, натискання будь-якої клавіші, кліки мишкою, збільшення або зменшення робочої області екрана ...) Через нього можна до будь-якого елементу HTML-коду отримати доступ і робити з цим елементом безліч маніпуляцій. Можна завантажувати дані без перезавантаження сторінки, виводити повідомлення, зчитувати або встановлювати cookie і виконувати безліч інших дій.

Вся унікальність даного мови програмування полягає в тому, що він підтримується практично всіма браузерами і повністю інтегрується з ними, а все що можна зробити з його допомогою – робиться дуже просто. Жодна інша технологія не вміщає в собі всі ці переваги разом. Наприклад, є такі, що не кросбраузерну (тобто підтримуються не всіма браузерами). Це – VBScript, ActiveX, XUL. А є такі, які з браузером не інтегровані в потрібному ступені, це – Java, Flash, Silverlight. На сьогоднішній день дана технологія активно розвивається, розробляється мову програмування Javascript [6].

jQuery - бібліотека JavaScript, що фокусується на взаємодії JavaScript і HTML. Бібліотека jQuery допомагає легко отримувати доступ до будь-якого елементу DOM, звертатися до атрибутів і вмісту елементів DOM, маніпулювати ними. Також бібліотека jQuery надає зручний API для роботи з AJAX. Зараз розробка jQuery ведеться командою jQuery.

Функції jQuery:

– звертатися до будь-якого елементу DOM (об'єктній моделі документа) і не тільки звертатися, але і маніпулювати ними;
– працювати з подіями;
– легко здійснювати різні візуальні ефекти;
– працювати з AJAX (дуже корисна технологія, що дозволяє спілкуватися з сервером без перезавантаження сторінки, але поки ми її чіпати не будемо);
– має величезну кількість JavaScript плагінів, призначених для створення елементів призначених для користувача інтерфейсів.

Філософія jQuery: так само, як і CSS відокремлює візуалізацію від структури HTML, JQuery відокремлює поведінку від структури HTML. Наприклад, замість прямої вказівки на обробник події натискання кнопки, управління передається JQuery, яка ідентифікує кнопки і потім перетворює його в обробник події кліка. Такий поділ поведінки і структури також називається принципом ненав'язливого JavaScript.

Бібліотека jQuery містить функціональність, корисну для максимально широкого кола завдань. Проте, розробниками бібліотеки не ставилося завдання суміщення в jQuery функцій, які підійшли б усюди, оскільки це призвело б до великого коду, велика частина якого не затребувана. Тому була реалізована архітектура компактного універсального ядра бібліотеки і плагінів. Це дозволяє зібрати для ресурсу саме ту JavaScript-функціональність, яка на ньому була б затребувана.[4,7]

3.4 Використання CSS-фреймворку для створення інтерфейсу

CSS-фреймворк – фреймворк, створений для спрощення роботи верстальника, швидкості розробки і виключення максимально можливого числа помилок верстки (проблеми сумісності різних версій браузерів і т. д.). Як і бібліотеки скриптових мов програмування, CSS-бібліотеки, зазвичай мають вигляд зовнішнього css-файлу, «підключаються» до проекту (додаються в заголовок веб-сторінки).

Переваги:

- дозволяє не досвідченому в тонкощах верстки програмісту або дизайнерові правильно створити HTML-макет;
- верстка на базі шарів, а не таблиць;
- більш швидка розробка;
- кросбраузерність4
- можливість використання генераторів коду і візуальних редакторів;
- однаковість коду при роботі в команді дозволяє знизити число розбіжностей при розробці.

Недоліки:

- бібліотеки бувають сильно «роздуті» – може бути багато зайвого коду, який не буде використовуватися в проекті;
- дизайн буде залежати від css-бібліотеки4
- через необхідність додавання безлічі класів до одного елементу порушується принцип, заради якого і був створений CSS: поділ описів структури і зовнішнього вигляду.

На своєму веб-сайті я використовував новий та досить потужний фрейворк Bootstrap на якому була побудована популярна соціальна мережа – Twitter.

Bootstrap (також відомий як Twitter Bootstrap) – вільний набір інструментів для створення сайтів і веб-додатків. Включає в себе HTML- і CSS-шаблони оформлення для типографіки, веб-форм, кнопок, міток, блоків навігації та інших компонентів веб-інтерфейсу, включаючи JavaScript-розширення. Bootstrap використовує сучасні напрацювання в області CSS і HTML, тому необхідно бути уважним при підтримці старих браузерів.

Історія: Цей фреймворк почав розроблятися як внутрішня бібліотека компанії Twitter під назвою Twitter Blueprint. Після кількох місяців розробки він був відкритий під назвою Bootstrap 19 серпня 2011 року.
Основними нововведеннями другої версії, що з'явилася 31 січня 2012 року, стали 12-колоночная сітка і підтримка адаптивності. Третя версія випущена 19 серпня 2013 року. У ній адаптивність отримала подальший розвиток, був здійснений перехід до концепції mobile first, оптимізації перш за все під мобільні пристрої. Дизайн за замовчуванням став плоским. Робота над четвертою версією розпочато 29 жовтня 2014 року. Альфа версія вийшла 19 серпня 2015 року[8].

До недоліків можна віднести бідну колірну гамму стандартного набору іконок. До переваг - хорошу реалізацію grid-сітки для масштабування вебсторінки, створення адаптивного дизайну.

Основні інструменти Bootstrap:

- сітки – заздалегідь задані розміри колонок, які можна відразу ж використовувати, наприклад ширина колонки 140 px відноситься до класу .span2 (.col-md-2 в третій версії фреймворка), який можна використовувати в CSS-описі документа;
- шаблони – фіксований або гумовий шаблон документа;
- типографіка – опису шрифтів, визначення деяких класів для шрифтів, таких як код, цитати і т. п.;
- медіа – представляє деякий спосіб упорядкування зображень та відео
- таблиці – кошти оформлення таблиць, аж до додавання функціональності сортування;
- форми – класи для оформлення форм і деяких подій, що відбуваються з ними;

- навігація – класи оформлення для табів, вкладок, сторінковий, меню і панелі інструментів;
- алерт – оформлення діалогових вікон, підказок і спливаючих вікон.

3.5 Особливості СКБД MySQL

Система управління базами даних (СКБД) MySQL – розробка шведської компанії MySQL AB. СКБД MySQL є програмним забезпе-ням з відкритим вихідним кодом, поширюваним за ліцензією GNU (GPL) і комерційної ліцензії для ситуацій, які не підпадають під дію ліцензії GPL. MySQL підтримує реляційну модель даних, тобто є реляційною СКБД.

3.5.1 Особливості MySQL

Починаючи з версії 5.0, СКБД MySQL практично повністю задовольняє стандарту структурованого мови запитів SQL і, отже, сумісна з іншими базами даних. Вичерпну інформацію про діалекті MySQL SQL можна отримати в [9, 10]

Розглянемо основні переваги СКБД MySQL:

- висока якість – MySQL характеризується стійкою роботою;
- поряд з Oracle, MySQL вважається однією з найшвидших СКБД в світі;
- відкритий код доступний для перегляду і модернізації, що дозволяє постійно покращувати програмний продукт;
- СКБД MySQL, розроблена з використанням мов C / C ++, протестована на багатьох платформах, серед яких Windows, Linux, FreeBSD, Mac OS X, OS / 2, Solaris і ін. ;
- MySQL підтримує API (Application Programming Interface, програмний інтерфейс програми) для C, C ++, Eiffel, Java, Perl, PHP, Python, Ruby і Tcl. MySQL можна успішно застосовувати як для побудови Web-сторінок з використанням Perl, PHP і Java, так і для роботи прикладної програми, створеної з використанням Delphi, Builder C ++ або платформи .NET;
- СКБД MySQL надає широкий вибір типів таблиць, в тому числі і сторонніх розробників, що дозволяє реалізувати оптимальну для розв'язуваної задачі продуктивність і функціональність;
- локалізація в MySQL виконана коректно. У користувача, як правило, не виникає проблем при обробці російського вмісту БД

На офіційному сайті MySQL http://www.mysql.com можна знайти доступні для скачування версії цього продукту.

Ще в версії MySQL 4.1 з'явилися такі важливі нововведення, як повна підтримка вкладених запитів і підтримка транзакцій. А у версії MySQL 5.0 стали доступними такі важливі механізми:

- збережені процедури і функції, які об'єднують в собі цілі послідовності запитів;
- тригери, тобто збережені процедури, прив'язані до події зміни таблиці;
- уявлення - вибірки даних, які можна уявити як повноцінні реально існуючі таблиці бази даних;
- курсори, що дозволяють циклі переглянути кожен рядок результуючої таблиці запитів;
- інформаційна схема, тобто стерпний набір уявлень системної таблиці, в якій зберігається різноманітна внутрішня інформація;
- обробники помилок;
- безліч нових функцій;
- система управління база дані.

3.5.2 Запуск MySQL

Робота з MySQL починається з запуску сервера MySQL (основної програми, яка управляє базами даних). Сервер MySQL має ім'я mysqld-nt.exe (в Windows) і mysqld.exe в UNIX-подібних ОС.

Якщо MySQL використовується в складі пакету «Денвер» (http://web.dklab.ru), то запуск MySQL-сервера буде здійснено автоматично при запуску «Денвера».

Адміністрування СКБД MySQL

3.5.3 Базові програми MySQL

У стандартному дистрибутиві MySQL поставляються клієнтські програми (утиліти), які взаємодіють з MySQL-сервером: mysql (консольний клієнт для доступу до MySQL-сервера, що дозволяє виконувати SQL-запити), mysqladmin (утиліта для виконання адміністративних функцій, таких як створення або видалення баз даних , отримання інформації про процеси, стан сервера і т. п.), mysqldump (утиліта для виведення вмісту бази даних MySQL у вигляді текстового файлу з SQL-операторами), mysqlimport (виконує пере-

несення інформації з текстового файлу в таблиці баз даних) і mysqlshow (відображає Щоб отримати інформацію про існуючі базах даних, табли-цях, полях і індексах).

До утиліт, які можуть функціонувати без підключення до сервера MySQL, відносяться: myisampack (стискає таблиці типу MylSAM, зменшуючи їх в розмірі і роблячи доступними тільки для читання), mysqlcheck (утиліта, яка використовується для опису, перевірки, оптимізації та відновлення таблиць), mysqlbinlog (дана утиліта використовується для читання вмісту журналу двійковій реєстрації при відновленні даних в нештатних си-ситуаціях) і perror (утиліта, яка виводить розшифровку кодів системних помилок і помилок MySQL).

3.5.4 Утиліти

Консольний клієнт mysql часто називають "термінальним монітором" або просто «монітором». У Windows для запуску mysql необхідно перейти в каталог C: \ mysql5 \ bin (звичайно, якщо MySQL встановлений в каталог C: \ mysql5) і набрати в командному рядку ім'я утиліти. Для з'єднання з сервером бази даних в параметрах утиліти mysql необхідно вказати ім'я користувача і його пароль. Наприклад, підключитися до сервера від імені користувача root можна, набравши команду (пароль root за замовчуванням порожній):

```
C:\mysql5\bin>mysql  -u root.
```

В результаті виводиться запрошення за яким можна набирати команди. Команди SQL-інструкції, за ред-ким винятком (exit, quit, use), повинні закінчуватися крапкою з комою.

Параметри в утиліти MySQL можуть мати дві форми: повну, починаючи-ющуюся з двох дефісів (наприклад, --user), і коротку, яка починається з одного дефіса (наприклад, -u). Можна застосовувати обидва варіанти, але для ряду параметрів є тільки повна форма.

Коли користувач вводить команду, вона відправляється на сервер для ви-конання, і, якщо немає помилок в синтаксисі, на екран виводиться результат у вигляді результуючої таблиці, а на новому рядку запрошення mysql>, після якого можна вводити нові команди. Для введення ключових слів можна використовувати будь-який регістр символів.

Якщо команда не поміщається на одному рядку, можливий перехід на другий рядок, після натискання клавіші <Enter> – запит відправляється

сервера тільки після того, як консольний клієнт mysql зустріне символ точки з зап'ятої. Запрошення командного рядка після введення першого рядка цього запроса змінюється з mysql> на –>:

```
mysql> SELECT VERSION ()
    -> CURRENT_DATE.
```

Таким чином програма mysql показує, що завершеного вираження вона поки що не отримала і очікує його повного введення. Точно так же утиліта mysql поводиться, коли очікує завершення рядка, укладеної в подвійні або одинарні лапки: запрошення командного рядка змінюється з mysql> на ">. Якщо рядок укладена в одинарні лапки, запрошення змінюється на '>. Для того щоб скасувати поточний запит , слід ввести послідовник-ність \ c.

Вже введені раніше команди не обов'язково вводити знову, для цього достатньо їх викликати клавішами «стрілка вгору» і «стрілка вниз», очистить рядок запиту можна за допомогою клавіші <Esc>. Повний список комбінацій клавіш, які застосовуються в редакторі утиліти mysgl, можно побачити, наприклад, в [9].

Установка з'єднання з віддаленим хостом проводиться так:

```
C: \ mysql5 \ bin> mysql -u root -h 192.168.200.1  -P 3306.
```

Тут 192.168.200.1 – IP-адреса хоста, 3306 – номер порту, до якого приписаний встановлений сервер. Для успішної установки з'єднання на сер-вірі повинні бути прописані IP-адреси, з яких користувач може об-ращаются до MySQL-сервера.

Наведемо деякі внутрішні команди утиліти mysql. Кожна команда має коротку і повну форму:

CONNECT (\ r). Команда використовується в форматі (тут і далі необ-ково елементи синтаксису укладатимуться в квадратні дужки)

```
CONNECT [db_name [host_name]]
```

для підключення до бази даних <db name>, розташованої на хості host_name. Якщо ім'я бази даних і хоста не вказані, замість них виконуються значення з поточного сеансу mysql:

– HELP (\h). Відображає довідкову інформацію про доступ-них командах в поточній версії mysql;

– USE (\u). Команда USE db_name дозволяє вибрати базу даних db_name для подальшої роботи з нею.

Наведемо приклад створення таблиці tbl в уже існуючій базі dbase:

```
mysql> USE dbase;
mysql> CREATE TABLE tbl (number INT, name TEXT).
```

Утиліта mysqladmin в MySQL бази даних являють собою підкаталоги, розташовані в каталозі даних C: \ mysql5 \ data, імена яких збігаються з іменами баз даних. Створення в цьому каталозі нового підкаталогу аналогічно процедурі з будівлі нової бази даних за допомогою оператора CREATE DATABASE.

Утиліта mysqladmin дозволяє виконувати операції адміністрування баз даних з командного рядка, не використовуючи SQL-запити. З її допомогою можна створювати і знищувати бази даних; контролювати сервер. При запуску mysqladmin можна задавати як параметри, так і команди, яких може бути кілька. Наприклад, команда:

```
mysqladmin -u root create db_name
```

створить нову базу даних з ім'ям db_name.

Повний список команд, які підтримуються mysqladmin, наведено в [11].

Утиліта mysqldump Утиліта mysqldump дозволяє отримати дамп вмісту бази даних чи сукупності баз для створення резервної копії або пересилання даних на інший SQL-сервер (не обов'язково MySQL). Дамп буде містити набір команд SQL для створення і / або заповнення таблиць.

Нехай ми маємо готову базу даних db. Створимо її дамп:

```
C: \ mysql5 \ bin> mysqldump -u root db> db.sql
```

Утиліта mysqldump приймає ім'я користувача за допомогою параметра u. Крім цього, після всіх параметрів вказується ім'я бази даних db, для якої здійснюється створення дампа. Так як висновок даних осуществляється в стандартний потік (за яким за замовчуванням закріплений екран моні-тора), його слід перенаправити в файл (в нашому прикладі це db.sql).

Перенаправлення даних здійснюється за допомогою оператора>. Якщо замість оператора> використовувати », то дані не будуть перезаписувати вже існуючий файл, а будуть додані в кінець файлу.

Mysqldump підтримує кілька спеціальних параметрів (повний список параметрів утиліт MySQL можна подивитися в [12]) До них відноситься параметр --databases або в скороченій формі -B.

Даний параметр дозволяє створити дамп відразу декількох баз даних, які можна перерахувати через пробіл після нього.

Так, наступна команда зберігає дампи баз даних db і mysql в файл db_mysql.sql:

C: \ mysql5 \ bin> mysqldump -u root -B db mysql> db_mysql.sql

Якщо необхідно зберегти дамп всіх баз даних MySQL-сервера, слід скористатися параметром --all-databases або в скороченій формі -A:

mysqldump -u root --all-databases> all_databases.sgl.

Отриманий в результаті дамп бази даних являє собою текстовий файл з SQL-інструкціями, виконати які можна за допомогою утиліти mysql.

При розгортанні дампа бази даних зручно скористатися наступною командою:

mysql -u root db_1 <db.sql.

Тут дані з дампа db.sql перенаправляються на стандартний вхід утиліти mysql, яка розміщує таблиці бази даних db в базі db_1.

Журнальні файли. утиліта mysqlbinlog

MySQL підтримує кілька видів журнальних файлів, в яких реєструються різні події, що відбуваються на MySQL-сервері.

У журнал помилок поміщаються повідомлення про всі помилки, що відбуваються під час запуску, роботи або зупинки MySQL-сервера. Загальний журнал запитів дозволяє реєструвати все установленние клієнтом з'єднання і виконані запити. Бінарний журнал реєстрації реєструє всі оператори, які призводять до зміни даних. У Журнал повільних запитів заносяться всі запити, виконання яких вимагало більше часу, ніж зазначено в системній темної змінної long_query_time (в секундах)

Файли бінарного журналу, які генерує сервер, записуються в двійковому форматі. Для роботи з такими файлами використовується утиліта mysqlbinlog. Синтаксис виклику утиліти mysqlbinlog виглядає наступним чином:

mysqlbinlog [parameters] filename.

Тут parameters – параметри утиліти, а filename -ім'я файлу бінарного журналу. Наприклад, для того щоб відобразити вміст бінарного журналу binlog.003, можна скористатися командою:

mysqlbinlog binlog.003

Утиліти mysqlimport, mysqlshow і mysqlcheck для пакетного завантаження даних з текстового файлу в таблицю в MySQL може бути використана утиліта mysqlimport:

mysqlimport [параметри] db_name tb1.txt tb2.txt ... .

Тут db_name – ім'я бази даних, з якою буде працювати утиліта. Далі через пробіл вказуються імена текстових файлів, інформація з яких буде завантажена в таблиці бази даних. При цьому імена таблиць повинні збігатися з іменами цих файлів.

Для правильного розподілу вмісту текстових файлів по колонках таблиць поля записів в них повинні бути розділені символом табуляції.

Утиліта mysqlshow надає кошти для швидкого перегляду наявних на сервері баз даних, їх таблиць, стовпців таблиць і індексів:

mysqlshow [параметри] [db_name [tb1_name [col_name]]]

Після імені утиліти перераховуються параметри, потім вказується ім'я бази даних db_name, таблиця tb1_name і стовпець col_name. Якщо не вказана база даних, відображаються всі бази даних, перегляд яких дозволений активного користувача. Якщо не вказані таблиці, відображаються всі таблиці бази db_name. Якщо не вказані стовпці, то відображаються всі стовпці таблиці tb1_name.

В іменах баз даних, таблиць і стовпців можна використовувати шаблони: символ «*» або «%» відповідає будь-якому числу символів, символ «?» Або «_» – одному довільному символу.

Утиліта mysqlcheck перевіряє і відновлює таблиці типу MyISAM. Для утиліти mysqlcheck існує альтернативна утиліта myisamchk. Основна відмінність цих двох функцій полягає в тому, що утиліта mysqlcheck повинна використовуватися при запущеному сервері MySQL, а myisamchk – немає. Існують три основні способи запуску утиліти mysqicheck:

mysqlcheck [parameter] db_name [tables];
mysqlcheck [parameter] --databases db_namel [db_name2 db_name3.];
mysqlcheck [parameter] --all-databases.

Якщо не вказуються жодні таблиці tables або використовується параметр --databases або --all-databases, будуть перевірені всі бази даних.

При виклику утиліти з параметром --check здійснюється перевірка таблиць (значення за замовчуванням), при виклику з параметром --repair – відновлення таблиць, при виклику з параметром --analyze – аналіз таблиць, при виклику з параметром --optimize – оптимізація таблиць.

3.5.5 Інші засоби адміністрування MySQL

Поряд з перерахованими утилітами при управлінні MySQL можуть бути використані спеціальні оператори SQL, такі як CREATE TABLE, ALTER TABLE, OPTIMIZE TABLE, ANALYZE TABLE, DROP TABLE і ін.

Однак крім коштів адміністрування MySQL з командного рядка існують спеціальні програми для управління цією СКБД. Вони володіють зручним призначеним для користувача інтерфейсом і дозволяють виробляти всі перераховані вище операції, включаючи, зрозуміло, і виконання SQL-запитів.

Серед таких програм можна виділити MySQL Administrator, MySQL Query Browser і MySQL Control Center, доступні за адресою http://dev.mysql.com/downloads/, а також phpMyAdmin (http://phpmyadmin.net). Останній додаток являє собою систему управління MySQL-сервером через web-інтерфейс. Програма phpMyAdmin входить до складу па-кета «Денвер», але може використовуватися окремо.

3.5.6 Користувачі MySQL

MySQL є багато користувачів середовищем, тому в ній передбачена реєстрація користувачів, що володіють різними правами (привілеями). Це можуть бути привілеї на перегляд таблиці, на перегляд і додавання нових записів і т. д. Можна дозволити користувачеві звертатися до сервера MySQL тільки з певного хоста і дозволити йому доступ лише до одного стовпцю деякої таблиці. Словом, MySQL має гнучкі можливості надання різних прав своїм користувачам, що забезпечує безпечну роботу з базами даних.

Відзначимо, що під користувачем MySQL не слід розуміти окремої людини, що сидить за комп'ютером. Це може бути група осіб, наділених однаковими правами доступу до сервера MySQL [13].

Інформація про зареєстрованих користувачів зберігається в таблиці user системної бази даних mysql.

В системі MySQL існує користувач root, наділений правами адміністратора, тобто має всі можливі привілеї.

Створити нового користувача можна за допомогою оператора

```
CREATE USER user [IDENTIFIED BY [PASSWORD] 'password'].
```

У наступному прикладі за допомогою утиліти mysql створюється користувач newuser з паролем hi:

```
mysql> CREATE USER newuser IDENTIFIED BY 'hi'.
```

Видалення (перейменування) користувачів проводиться за допомогою оператора DROP USER і RENAME USER відповідно:

```
DROP USER user;
RENAME USER user TO user1.
```

Перераховані операції можуть бути здійснені і шляхом прямого редагування таблиці mysql.user (так позначається приналежність таблиці user до бази mysql).

Для надання (скасування) привілеїв окремим користувачам MySQL підтримує стандартні оператори SQL GRANT і REVOKE відповідно.

Список системних привілеїв в MySQL не обмежується операціями SELECT, INSERT, UPDATE, DELETE і REFERENCES; користувачеві можуть

бути надані права на створення таблиць (CREATE), їх редагування (ALTER), на зміну (створення) процедури, що (ALTER ROUTINE) і т. д.

3.5.7 Типи і структура таблиць

Створення, заповнення та зміна таблиць в MySQL здійснюються за допомогою стандартних SQL операторів CREATE TABLE, ALTER TABLE, INSERT, UPDATE, DELETE. Визначено також оператор REPLASE, який діє аналогічно INSERT, але таким чином, що якщо значення індексу unique або primary KEY в старому записі таблиці таке ж, як і в новій, то стара запис перед занесенням нової буде видалена. Оператор TRUNCATE TABLE призначений для повного очищення таблиці.

Переглянути список таблиць поточної бази даних дозволяє конструкція SHOW TABLES, а отримати опис стовпців конкретної таблиці можна за допомогою оператора DESCRIBE, який не є стандартним і присутній лише в MySQL SQL.

СКБД MySQL підтримує кілька типів таблиць.

Тип MyISAM призначається за замовчуванням при створенні таблиці. Кожна MyISAM-таблиця зберігається на диску в трьох файлах, імена яких збігаються з назвою таблиці table_name: table_name.frm, table_name.myd і table_name.myi. Перший з них містить структуру таблиці і інформацію про шпальтах і індексах. Другий містить дані таблиці, а третій – її індекси. Таблиці типу MyISAM можна переносити з сервера простим копірова-ням файлів.

Таблиці типу InnoDB можуть досягати обсягу в 1 Тбайт. Таблиці цього типу зберігаються в єдиному табличному просторі. Даний тип таблиць підтримує транзакції, блокування на рівні окремих записів і – єдиний з типів таблиць MySQL – підтримує зовнішні ключі і каскадне видалення (оновлення). Втім, таблиці InnoDB поступаються в швидкості обробки таблиць MyISAM.

Для створення InnoDB-таблиці застосовується ключове слово ENGINE = InnoDB в операторі CREATE TABLE.

Також MySQL підтримує наступні типи таблиць: MERGE (дозволяє згрупувати кілька таблиць MyISAM в одну), MEMORY (таблиця зберігається в оперативній пам'яті), EXAMPLE, BDB, NDB Cluster (для організації кластерів розподілених таблиць), ARCHIVE (для зберігання великих обсягів інформації в стислому вигляді), CSV (формат таблиць Ms Excel), FEDERATED (для зберігання даних на віддаленому комп'ютері), BLACKHOLE.

3.5.8 Типи даних в MySQL

MySQL підтримує кілька типів даних.

Числові дані. До них відносять цілі числа, що не містять дробової частини (наприклад, 124), а також речові числа, що мають як цілу, так і дробову частини (наприклад, 56.45). Числові дані діляться на точкові (bit, boolean, integer і decimal) і наближені (float, real і double precision).

MySQL має п'ять цілих типів: TINYINT, SMALLINT, MEDIUMINT, BIGINT. Різниця між ними полягає в діапазоні величин, які можна зберігати в шпальтах такого типу. Тип TINYINT має діапазон від -27 до 27-1, тип SMALLINT – від -215 до 215-1, тип MEDIUMINT – від -223 до 223- 1, тип BIGINT – від -263 до 263-1.

Цілі типи даних можуть бути оголошені позитивними. Для цього призначено ключове слово UNSIGNED. В цьому випадку елементам даного стовпця можна буде привласнити негативні значення, а допустимий діапазон значень, які може приймати тип, подвоюється.

При оголошенні цілого типу можна задати кількість відводяться під число розрядів (від 1 до 255).

Тип BIT призначений для зберігання бітових полів.

Тип BOOLEAN є синонімом для TINYINT (1). Значення 1 розглядається як істина (true), а 0 – як брехня (false).

Тип DECIMAL, а також його синоніми NUMERIC і DEC призначені для величин підвищеної точності, наприклад, для грошових даних. Необхідна точність задається при оголошенні даних одного з цих типів, наприклад:

DECIMAL (5,2). Тут цифра 5 визначає загальне число символів, що відводяться під число, а цифра 2 задає кількість знаків після коми. При цьому перший параметр може приймати максимальне значення, рівне 64, а другий – максимальне значення, рівне 30.

Для подання дійсних (наближених) типів в СКБД MySQL є типи: FLOAT (діапазон від -3.4E + 38 до 3.4E + 38, точність 1.2E-39), DOUBLE і DOUBLE PRECISION (-1.8E + 308 до 1.8E + 308 , точність 2.2E-308). Числові типи даних з плаваючою точкою також можуть мати параметр UNSIGNED. Як і в цілочисельних типах, цей атрибут запобігає зберігання в зазначеному стовпці негативних величин, але, на відміну від цілочисельних типів, максимальний інтервал для величин стовпчика залишається колишнім.

При формуванні структури таблиці необхідно звертати увагу на розмір, займаний тим або іншим типом даних: якщо значення в полях стовпчика

ніколи не будуть виходити за межі 100, не слід вибирати тип більше TINYINT. Якщо ж передбачається зберігати тільки цілочисельні дані, то застосування атрибута UNSIGNED дозволить збільшити діапазон в два рази.

Строкові дані – послідовність символів, укладених в одинарні або подвійні лапки: 'Hello world', '123', "MySQL". Оскільки в якості стандарту в SQL визначені одинарні лапки, для сумісності з іншими базами даних рекомендується використовувати саме їх. Розрізняють строкові типи CHAR, VARCHAR, BLOB, TEXT, MEDIUMTEXT, MEDI-UMBLOB, LONGTEXT, LONGBLOB, ENUM, SET.

Тип CHAR дозволяє зберігати рядок фіксованої довжини; його доповнює тип VARCHAR, що дозволяє зберігати рядки змінної довжини. Довжина рядка може змінюватися від 0 до 65 535. При створенні таблиці можна змішувати стовпці типу CHAR і VARCHAR. В цьому випадку СКБД MySQL змінить тип стовпців відповідно до правила: якщо таблиці присутній хоча б один стовпець змінної довжини, всі стовпці типу CHAR будуть приведені до типу varchar.

Тип TEXT зазвичай використовується для зберігання великих обсягів тексту (до 216-1 символів), в той час як BLOB (також до 216-1 символів) – для великих двійкових об'єктів, таких як електронні документи, зображення, музичні файли і тобто типи MEDIUMBLOB, MEDIUMTEXT мають максимальний розмір (224-1) символів, а типи LONGTEXT, LONGBLOB – (232 -1) символів.

До особливих типів даних відносяться ENUM і SET. Рядки цих типів приймають значення з заздалегідь визначеного списку допустимих значень. Основна відмінність між ними полягає в тому, що значення типу ENUM має містити точно одне значення із зазначеного безлічі, тоді як стовпці SET можуть містити будь-який (або все) елементи заздалегідь заданої множини. Так, значення для стовпця, оголошеного як ENUM ('y', 'n'), можуть прирозуміти значення або 'y', або 'n'.

Для типу SET, так само як і для типу ENUM, при оголошенні задається список можливих значень, але в комірці таблиці може зберігатися будь-яке значення зі списку, а порожній рядок означає, що жоден з елементів списку не вибрано. Наприклад, значення для стовпця SET ('y', 'n') можуть приймати значення ('y', 'n'), ('y'), ('n') і порожня множина ().

Календарні дані. СКБД MySQL має п'ять календарних типів даних: DATE, DATETIME, TIME, TIMESTAMP і YEAR. Тип DATE призначений для зберігання дати, TIME – для часу доби, а TIMESTAMP для подання та

дати, і часу доби. Тип TIMESTAMP призначений для представлення дати і часу доби у вигляді числа секунд, що пройшли з півночі 1 січня 1970 року. Тип даних YEAR дозволяє зберігати тільки рік. Для значень, що мають тип DATE і DATATIME прийнятий формат YYYY-MM-DD або YY-MM-DD. У типах TIME і DATATIME час наводиться в звичному форматі hh: mm: ss.

Якщо час задається в неприпустимому форматі, то в поле записується нульове значення. Нульове значення присвоюється полях тимчасового типу та за замовчуванням, коли їм не присвоюється ініціює значення.

Тип NULL. Якщо поле може приймати значення NULL, то у визначенні стовпця після типу даних слід вказати ключове слово NULL. Якщо ні при яких обставинах поле не повинно приймати значення NULL (реєстрація неможлива, якщо прізвище користувача невідома), слід вказати ключове слово NOT NULL, наприклад:

```
CREATE TABLE tbl(
id INT NOT NULL,
col_name VARCHAR (255) NOT NULL,).
```

При виборі типу даних слід пам'ятати, що обробка числових даних відбувається швидше строкових. Так як типи даних ENUM і SET мають внутрішнє числове уявлення, їм слід віддавати перевагу перед іншими видами строкових даних, якщо надається така можливість.

Типи фіксованої довжини обробляються швидше типів змінної довжини, так як в останньому випадку при частих удаленіях і модифікаціях таблиці відбувається її фрагментація.

3.5.9 Індекси в MySQL

Визначення індексу по деякому стовпцю означає, що СКБД створює його копію, яка потім постійно підтримується в відсортованому стані. Пошук по такому колонки здійснюється дуже швидко, так як заздалегідь відомо, де необхідно шукати потрібне значення. Індекс може бути визначений по одному або по декількох стовпцях; в одній таблиці може бути кілька індексів.

MySQL підтримує кілька видів індексів: первинний ключ – головний індекс таблиці; звичайний індекс – таких індексів може бути не-скільки; унікальний індекс – значення в індексованих стовпці не повинен-ни повторюватися (унікальних індексів також може бути кілька), повнотекстовий індекс спеціальний вид індексу для стовпців типу TEXT, що

дозволяє виробляти повнотекстовий пошук. Індекси можуть мати як власні імена, так і імена індексованих стовпців. Один індекс може охоплювати один або декілька стовпців.

У таблиці MySQL може бути не більше 32 індексів, а многостолбцовий індекс може включати не більше 16 стовпців.

В якості первинного ключа може виступати не тільки цілочисельний стовпець, а й текстові дані. В цьому випадку при створенні таблиці необхідно додати в круглих дужках число символів, що входять в індекс:

PRYMARY KEY(name(10)).

Можна індексувати від 1 до 1000 символів текстового стовпчика. Первинний ключ може бути оголошений не по однім стовпці:

PRIMARY KEY (id_PC, name (10)).

Цілочисельний первинний ключ може бути забезпечений атрибутом auto_increment, який забезпечує автоматичне створення унікального значення:

id INT (11) NOT NULL PRIMARY KEY auto_increment.

Передача колонки, забезпеченому цим атрибутом, значення null або 0 призводить до автоматичного присвоєння йому максимального значення стовпця плюс 1. Даний механізм є досить зручним і дозволяє не турбуватися про створення унікального значення засобами прикладної програми, яка працює з СКБД MySQL. Значення AUTO_INCREMENT можна змінити і почати відлік не з 1, а, наприклад, від 1000.

Оголошення звичайних індексів проводиться за допомогою ключового слова index або key: ... name TINYTEXT NOT NULL KEY name(name (10)) ... .

Для унікальних індексів вводиться додаткове ключове слово unique, тобто пишеться UNIQUE index і unique key: Створення звичайних індексів, на відміну від первинного ключа і унікального індексу, допустимо для стовпців, забезпечених атрибутом NULL. Для звичайних індексів (на відміну від первинного ключа) неприпустимо їх визначення разом зі стовпцем.

Посилальна цілісність і зовнішні ключі в СКБД MySQL підтримуються, починаючи з версії 3.23.44, але тільки для таблиць типу InnoDB.

Відзначимо, що MySQL підтримує оператори CREATE INDEX і DROP INDEX для створення (видалення) індексів після визначення таблиць.

3.5.10 Транзакції

Під транзакцією розуміють виконання групи SQL-запитів як єдиної операції. Іншими словами або виконуються всі ці запити, або (якщо на якомусь етапі стався збій) скасовуються всі результати їх роботи. Транзакції виконуються незалежно один від одного це означає, що до завершення транзакції блокується доступ до об'єктів бази даних, що піддаються обробці.

Транзакції підтримуються в MySQL для таблиць типу BDB і InnoDB.

Транзакції не можуть бути вкладеними, тому що всякий оператор, який розпочинає нову транзакцію, завершує попередню.

За замовчуванням MySQL працює в режимі автоматичного завершення транзакцій, це означає, що після виконання будь-якого запиту, що модифікує дані, результат відразу записується в базі. Для об'єднання декількох операторів в транзакцію можна відключити цей режим, використовуючи системну змінну AUTOCOMMIT:

```
mysql> SET AUTOCOMMIT = 0.
```

Після цього для завершення транзакції (і збереження змін) слід використовувати оператор COMMIT, а для скасування всіх дій - оператор ROLLBACK, наприклад:

```
mysql> SET AUTOCOMMIT = 0;
mysql> INSERT INTO table VALUES (val1, val2);
mysql> SELECT * FROM table;
mysql> ROLLBACK;
mysql> INSERT INTO table VALUES (val3, val4);
mysql> SELECT * FROM table;
mysql> COMMIT;
mysql> SET AUTOCOMMIT = 1.
```

Об'єднати послідовність запитів в транзакцію можна також з використанням стандартного оператора SET TRANSACTION:

```
mysql> SET TRANSACTION
```

```
mysql> INSERT INTO table VALUES (value1, value2);
mysql> SELECT * FROM table;
mysql> COMMIT.
```

Нарешті, в MySQL (для таблиць InnoDB) існує ще одна можливість визначити транзакцію: оператори SAVEPOINT і ROLLBACK TO SAVEPOINT. Перший з них встановлює іменовану точку початку транзакції; другий дозволяє в разі потреби зробити відкат до стану, в якому перебувала база в момент установки іменованої точки. Якщо виконується оператор COMMIT або ROLLBACK без вказівки імені точки повернення, все іменовані точки віддаляються. наприклад:

```
mysql> SAVEPOINT mypoint;
mysql> INSERT INTO table VALUES (value1, value2);
mysql> SELECT * FROM table;
mysql> ROLLBACK TO SAVEPOINT mypoint.
```

3.5.11 Функції і оператори в MySQL

СКБД MySQL підтримує більшість стандартних логічних і бітових операторів, операторів порівняння і, звичайно ж, арифметичні операції, серед яких цілочисельне ділення «div» і взяття залишку від ділення «%». Якщо при перемножуванні цілих чисел результат виходить за межі типу BIGINT, то повертається 0. Розподіл на 0 в MySQL не приводить до помилки: повертається значення NULL.

Як і інші СКБД, MySQL володіє великим числом вбудованих функцій, серед яких математичні, функції дати і часу, функції роботи з рядками, функції шифрування даних і багато інших. Повний список цих функцій можна побачити в [10]. Слід зазначити, що синтаксис MySQL не допускає використання пробілу між ім'ям функції і круглими дужками, які є обов'язковими навіть в разі відсутності аргументів.

Створення простого сценарію на PHP для роботи з MySQL використовуються різні мови програмування, в числі яких Perl, Java, C ++, Python і ін. Наведемо взаємодія СКБД MySQL (бажану багатьма хост-провайдерами) і мови PHP як найбільш популярного засобу створення Інтернет-додатків.

Нагадаємо, що Інтернет працює за принципом «клієнт-сервер». Клієнт (зазвичай браузер) надсилає запит серверу, сервер обробляє запит і посилає клієнтові відповідь.

Запитуваний браузером URL може вказувати на статичну html-сторінку (в цьому випадку вміст сторінки відобразиться на екрані браузера) або на скрипт, тобто спеціальну програму, яка виконується на сервері і генерує вміст сторінки, що посилається назад в браузер. Скрипту може бути передана деяка інформація, наприклад, дані html-форм.

Простий скрипт на мові PHP виглядає як html-документ зі вставками коду, що виконується на сервері.

Для написання і налагодження скриптів на локальному комп'ютері необхідно встановити web-сервер Apache, а також супутнє ПО. Вельми зручно використовувати систему «Денвер», яка включає в себе Apache, PHP, MySQL, Perl і інші засоби, що застосовуються при розробці web-додатків. Базовий пакет «Денвера» можна завантажити за адресою http://web.dklab.ru

Після запуску «Денвера» на комп'ютері створюється віртуальний диск (зазвичай Z), що містить каталоги etc, home, tmp, usr. В каталозі home слід створити нову папку, чиє ім'я збігається з ім'ям майбутнього віртуального хоста, наприклад myhost.ru (або myhost). У створеному каталозі необхідно помістити папку www, в якій і будуть розташовуватися скрипти (сценарії). Для того щоб створити новий віртуальний хост необхідно перезапустити «Денвер». Тепер звернутися до сценарію myscript.php, розташованому в папці Z: \ home \ myhost.ru \ www можна з рядка браузера наступним чином: http://myhost.ru/myscript.php

Якщо скрипт має ім'я index.php, то він викликається за замовчуванням: http://myhost.ru/

Нехай на сервері вже існує база даних DB, що містить таблицю tbl. Таблиця tbl містить два стовпці: id_name (первинний ключ з атрибутом auto_increment) і name.

Створимо спочатку файл connect.php, який можна буде включати в любие PHP-сценарії для з'єднання з базою DB.

Файл connect.php має такий вигляд:

```
                                        hostname
$ User                                     user
$ Pass                              pass
```

```
$ DBname = 'DB                          DBname
$ Db_connection = mysql_connect
($ hostname, $ user, $ pass);
                                              mysql_select_db
```

Змінні $ hostname і $ user зберігають ім'я і пароль одного з користувачів, зареєстрованих в MySQL. Як ім'я хоста може вказуватися localhost. У змінній $ db_connection зберігається посилання на створене з'єднання. Більшість функцій PHP для роботи з MySQL приймають її як необов'язкового аргументу.

Виконання SQL-запитів до бази даних виконується в php-сценарії наступним чином:

$ Query = 'SQL-запит';
$ Query_result = mysql_query ($ query);

У змінній $ query_result зберігається посилання на результат виконання запиту, це означає, що окажчик на дані, повернуті MySQL.

Створимо сценарій index.php, який виводить html-форму, заносить в базу даних введену в неї інформацію і виводить вміст бази в браузер.

Файл index.php має такий вигляд:

```
<HTML>


If
 (isset ($ _ POST [ 'submit'])) {

include_once ( "connect.php");


            -

$ Query = "INSERT INTO tbl VALUES (NULL, '$ name')";

$ Query_result = mysql_query (Squery);

Header ( "Location:
{$
_SERVER [ 'SCRIPT_NAME']}? ". Time ());
```

```
exit ();

$ Query_result = mysql_query ( 'SELECT * FROM tbl');
while
($ row = mysql_fetch_array ($ query_result)) {
echo'name: '. $ row [1].' <br> \ n '; }}
Else


                                        ?>
<Form action = "index.php" method = "post">
            <Input type = "text" name = "name" size = "30"
/>
<br>

</ Form>
<? Php                                                  ?> </ Html>
```

Розглянемо деякі деталі цього скрипта. Дані, передані web-серверу за допомогою html-форми методом POST, доступні в php-сценарії як елементи масиву $ _POST, наприклад $ _POST ['submit'].

Запит INSERT INTO tbl VALUES (NULL, '$name") служить для вставки нового запису в таблицю. Застосування NULL в даному випадку спричинить за собою присвоєння атрибуту id_name чергового порядкового номера.

Самопереадресація (вказівка браузеру перейти заново до адресою поточного скрипта) після процедури вставки даних необхідна для того, щоб при натисканні кнопки «оновити» в браузері тільки що додані дані не додавалися в базу повторно.

Функція mysql_fetch_array () повертає по одному рядку результат вибірки даних у вигляді масиву. цикл

while ($ row = mysql_fetch_array ($ query_result))

виконується до тих пір, поки є ще дані. Елемент масиву $ row [1] є другим (містить значення name), тому що нумерація елементів масиву в PHP починається з 0. Детальну інформацію про програмуванні на PHP можна почерпнути в [14].

3.6 Опис СКБД Firebird

Firebird (FirebirdSQL) – кроссплатформенная система управління базами даних (СКБД), що працює на Mac OS X, Linux, Microsoft Windows і різноманітних Unix платформах.

Firebird використовується в різних промислових системах (складські та господарські, фінансовий і державний сектори) з 2001 р Це комерційно незалежний проект C і C ++ програмістів, технічних радників.

Серед недоліків: відсутність кеша результатів запитів, повнотекстових індексів, значне падіння продуктивності при зростанні внутрішньої фрагментації бази. Над вирішенням цих проблем невпинно працює співтовариство.

Основні характеристики: відповідність вимогам ACID: Firebird зроблений спеціально, щоб задовольняти вимогам «атомарности, цілісності, ізоляції та надійності» транзакцій («Atomicity, Consistency, Isolation and Durability».

Основна особливість Firebird – версійна архітектура, що дозволяє сервера обробляти різні версії однієї і тієї ж записи в будь-який час таким чином, що кожна транзакція бачить свою версію даних, не заважаючи сусіднім («читають транзакції не блокують пишучих, а пишуть – не блокують читають »). Це дозволяє використовувати одночасно OLTP і OLAP запити.

Збережені процедури, використовуючи мову PSQL (процедурний SQL) Firebird, можна створювати складні збережені процедури для обробки даних повністю на стороні сервера. Для генерації звітів особливо зручні збережені процедури з можливістю вибірки, що повертають дані у вигляді набору записів. Такі процедури можна використовувати в запитах точно так же, як і звичайні таблиці.

Події: Збережені процедури і тригери можуть генерувати події, на які може підписатися клієнт. Після успішного завершення транзакції (COMMIT) він буде сповіщений про події, що відбулися і їх кількості.

Генератори: Ідея генераторів (послідовностей) уможливлює просту реалізацію Автоінкрементний полів, і не тільки їх. Генератори є 64-бітними збереженими в базі даних лічильниками, що працюють незалежно від транзакцій. Вони можуть бути використані для різних цілей, таких як генерація первинних ключів, управління тривалими запитами в сусідніх транзакціях, і т. д.

Бази даних тільки для читання: дозволяють поширювати бази даних, приміром, на CD-ROM. Особливо спрощує поширення даних їхнє використання в комбінації з вбудовуваної версією сервера Firebird (Firebird Embedded).

Повний контроль за транзакціями: Одне клієнтське додаток може виконувати безліч одночасних транзакцій. У різних транзакції можуть бути

використані різні рівні ізоляції. Протокол двофазного підтвердження транзакцій забезпечує гарантовану стійкість при роботі з декількома базами даних. Також доступні оптимістичне блокування даних і точки збереження транзакцій.

Резервне копіювання на льоту. Для резервного копіювання немає потреби зупиняти сервер. Процес резервного копіювання зберігає стан бази даних на момент свого старту, не заважаючи при цьому роботі з базою. Крім того, існує можливість виробляти инкрементальное резервне копіювання БД.

Тригери, для кожної таблиці можливе призначення декількох тригерів, що спрацьовують до або після вставки, оновлення або видалення записів. Для тригерів використовується мова PSQL, дозволяючи вносити початкові значення, перевіряти цілісність даних, викликати виключення і т. д. В Firebird 1.5 з'явилися «універсальні» тригери, що дозволяють в одному тригері обробляти вставки, оновлення та видалення записів таблиці.

Зовнішні функції бібліотеки з UDF (User Defined Function) можуть бути написані на будь-якій мові і легко підключені до сервера у вигляді DLL / SO, дозволяючи розширювати можливості сервера «зсередини».

Декларативне опис посилальної цілісності: Забезпечує несуперечність і цілісність багаторівневих відносин «master-detail» між таблицями.

Набори символів Firebird підтримує безліч міжнародних наборів символів (включаючи Unicode) з безліччю варіантів сортування.

Відповідність стандарту СКЛ: Firebird повністю підтримує SQL-92 Entry Level 1 і реалізує більшу частину стандарту SQL-99 с деякими дуже корисними доповненнями. Це включає вираження DML / DDL, синтаксис об'єднань FULL / LEFT / RIGHT [OUTER] JOIN, вираження UNION, DISTINCT, підзапити (IN, EXISTS), вбудовані функції (AVG, SUM, MIN, MAX, COALESCE, CASE, ...), обмеження цілісності (PRIMARY KEY, UNIQUE, FOREIGN KEY), і все загальні типи даних SQL.

Firebird також реалізує обмеження перевірки (check constraints) на рівні доменів і полів, відображення (views), виключення, ролі і управління правами доступу. Для більш докладної інформації див. Firebird Reference Guide і Release Notes.

Історія Firebird заснована на вихідному коді InterBase 6.0, який був випущений як Open Source компанією Borland в серпні 2000 року. Історія Interbase починається в 1984 році, таким чином, продукт є спадкоємцем майже 30-річного досвіду роботи з реляційними базами даних.

2 червня 2015 року Минкомсвязь Росії видає «Протокол експертної оцінки проектів з імпортозаміщення інфраструктурного програмного забезпечення», в якому вітчизняне програмне засіб «Ред база даних», засноване на програмному коді FireBird, займає 5-е місце в розділі «Системи управління базами даних» протоколу експертної оценкі. На даний момент цей форк СКБД широко застосовується в ряді федеральних міністерств, служб, бюджетних установ і відомств різного рівня. Для кінцевого користувача вибір даного варіанта FireBird ґрунтується часто на тому, що, на відміну від «батьківського» проекту, він сертифікований ФСТЕК Росії і відповідає вітчизняним вимогам захисту інформації, проходить жорстке тестування і при цьому сумісний на рівні резервних копій з основною гілкою розробки.

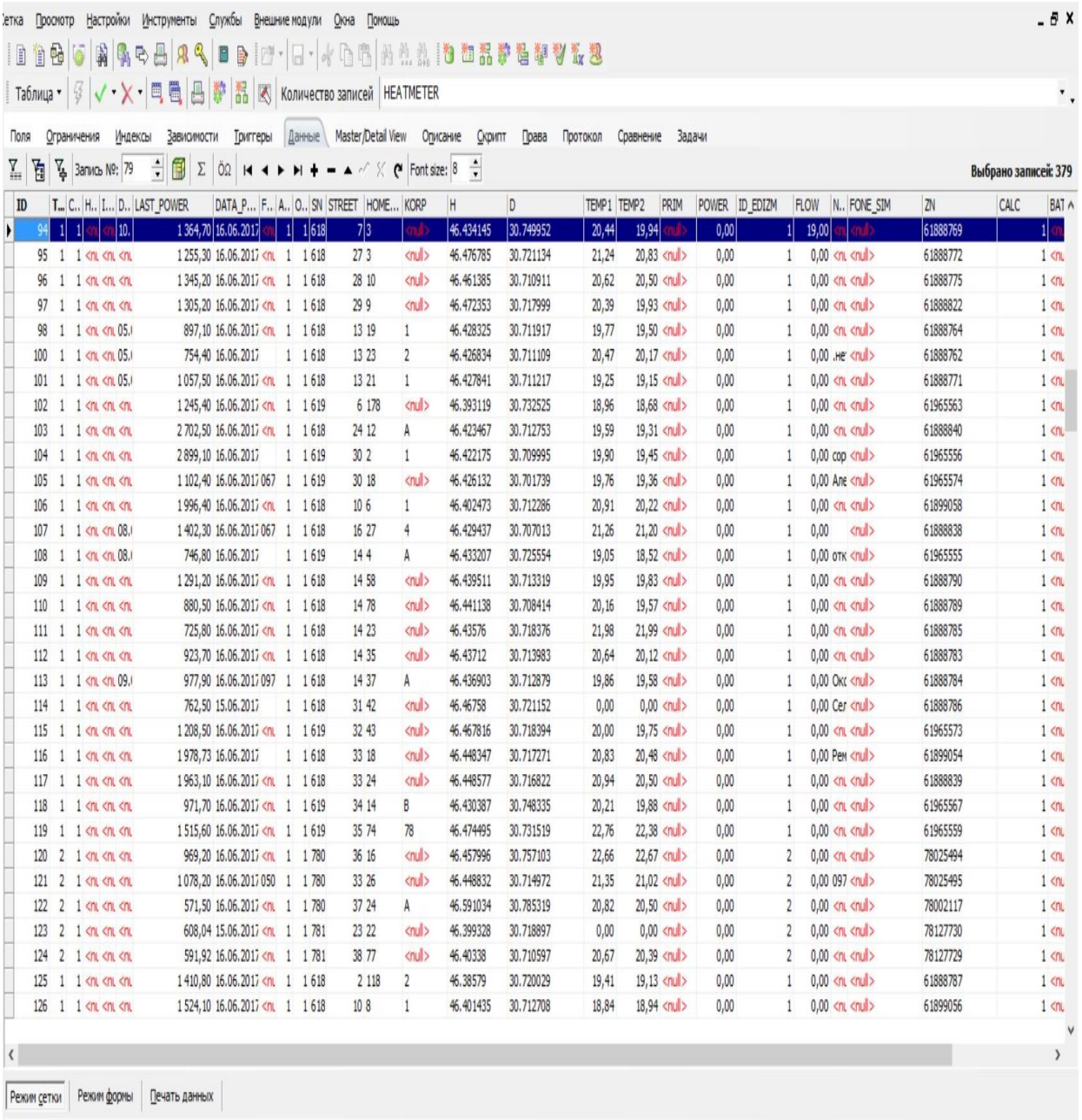
Для роботи з Firebird я використовував програму IBExpert. IBExpert – GUI-оболонка, призначена для розробки і адміністрування баз даних Inter-Base та Firebird, а також для вибору і зміни даних, що зберігаються в базах.

Як основні переваги IBExpert розробники вказують:

- підтримка InterBase версій 4.x, 5.x, 6.x, 7.x, 2007 і 2009; Firebird 1.x, 2.x, 3.x; Yaffil 1.x;
- робота одночасно з декількома базами даних;
- окремі редактори для всіх об'єктів БД з синтаксичної підсвічуванням;
- потужний SQL-редактор з історією запитів і можливістю їх фонового виконання;
- автозавершення коду SQL (назва таблиць, полів, і т. п.);
- відладчик збережених процедур і тригерів;
- пошук в метаданих;
- повне і часткове вилучення даних і метаданих;
- аналізатор залежностей об'єктів баз даних;
- звіти по метаданих;
- менеджери користувачів і призначених для користувача привілеїв;
- експорт даних в різні формати.

IBExpert володіє безліччю полегшують роботу компонентів: візуальний редактор для всіх об'єктів бази даних, редактор SQL і виконавець скриптів, відладчик для збережених процедур і тригерів, будівник області, інструмент для імпорту даних з різних джерел, власний скриптова мова, а також дизайнер баз даних [15].

На рис. 3.1 зображена робота з програмою IBExpert, з таблицею всіх лічильників які встановленні в місті Одеса.



ID	T...	C..	H..	I...	D..	LAST_POWER	DATA_P...	F..	A..	O..	SN	STREET	HOME...	KORP	H	D	TEMP1	TEMP2	PRIM	POWER	ID_EDIZM	FLOW	N..	FONE_SIM	ZN	CALC	BAT
94	1	1	<nu	<nu	10.	1 364,70	16.06.2017	<nu	1	1	618	7	3	<null>	46.434145	30.749952	20,44	19,94	<null>	0,00	1	19,00	<nu	<null>	61888769	1	<nu
95	1	1	<nu	<nu	<nu	1 255,30	16.06.2017	<nu	1	1	618	27	3	<null>	46.476785	30.721134	21,24	20,83	<null>	0,00	1	0,00	<nu	<null>	61888772	1	<nu
96	1	1	<nu	<nu	<nu	1 345,20	16.06.2017	<nu	1	1	618	28	10	<null>	46.461385	30.710911	20,62	20,50	<null>	0,00	1	0,00	<nu	<null>	61888775	1	<nu
97	1	1	<nu	<nu	<nu	1 305,20	16.06.2017	<nu	1	1	618	29	9	<null>	46.472353	30.717999	20,39	19,93	<null>	0,00	1	0,00	<nu	<null>	61888822	1	<nu
98	1	1	<nu	<nu	05.(	897,10	16.06.2017	<nu	1	1	618	13	19	1	46.428325	30.711917	19,77	19,50	<null>	0,00	1	0,00	<nu	<null>	61888764	1	<nu
100	1	1	<nu	<nu	05.(	754,40	16.06.2017		1	1	618	13	23	2	46.426834	30.711109	20,47	20,17	<null>	0,00	1	0,00	.не	<null>	61888762	1	<nu
101	1	1	<nu	<nu	05.(	1 057,50	16.06.2017	<nu	1	1	618	13	21	1	46.427841	30.711217	19,25	19,15	<null>	0,00	1	0,00	<nu	<null>	61888771	1	<nu
102	1	1	<nu	<nu	<nu	1 245,40	16.06.2017	<nu	1	1	619	6	178	<null>	46.393119	30.732525	18,96	18,68	<null>	0,00	1	0,00	<nu	<null>	61965563	1	<nu
103	1	1	<nu	<nu	<nu	2 702,50	16.06.2017	<nu	1	1	618	24	12	A	46.423467	30.712753	19,59	19,31	<null>	0,00	1	0,00	<nu	<null>	61888840	1	<nu
104	1	1	<nu	<nu	<nu	2 899,10	16.06.2017		1	1	619	30	2	1	46.422175	30.709995	19,90	19,45	<null>	0,00	1	0,00	cop	<null>	61965556	1	<nu
105	1	1	<nu	<nu	<nu	1 102,40	16.06.2017	067	1	1	619	30	18	<null>	46.426132	30.701739	19,76	19,36	<null>	0,00	1	0,00	Але	<null>	61965574	1	<nu
106	1	1	<nu	<nu	<nu	1 996,40	16.06.2017	<nu	1	1	618	10	6	1	46.402473	30.712286	20,91	20,22	<null>	0,00	1	0,00	<nu	<null>	61899058	1	<nu
107	1	1	<nu	<nu	08.(	1 402,30	16.06.2017	067	1	1	618	16	27	4	46.429437	30.707013	21,26	21,20	<null>	0,00	1	0,00		<null>	61888838	1	<nu
108	1	1	<nu	<nu	08.(	746,80	16.06.2017		1	1	619	14	4	A	46.433207	30.725554	19,05	18,52	<null>	0,00	1	0,00	отк	<null>	61965555	1	<nu
109	1	1	<nu	<nu	<nu	1 291,20	16.06.2017	<nu	1	1	618	14	58	<null>	46.439511	30.713319	19,95	19,83	<null>	0,00	1	0,00	<nu	<null>	61888790	1	<nu
110	1	1	<nu	<nu	<nu	880,50	16.06.2017	<nu	1	1	618	14	78	<null>	46.441138	30.708414	20,16	19,57	<null>	0,00	1	0,00	<nu	<null>	61888789	1	<nu
111	1	1	<nu	<nu	<nu	725,80	16.06.2017	<nu	1	1	618	14	23	<null>	46.43576	30.718376	21,98	21,99	<null>	0,00	1	0,00	<nu	<null>	61888785	1	<nu
112	1	1	<nu	<nu	<nu	923,70	16.06.2017	<nu	1	1	618	14	35	<null>	46.43712	30.713983	20,64	20,12	<null>	0,00	1	0,00	<nu	<null>	61888783	1	<nu
113	1	1	<nu	<nu	09.(	977,90	16.06.2017	097	1	1	618	14	37	A	46.436903	30.712879	19,86	19,58	<null>	0,00	1	0,00	Окс	<null>	61888784	1	<nu
114	1	1	<nu	<nu	<nu	762,50	15.06.2017		1	1	618	31	42	<null>	46.46758	30.721152	0,00	0,00	<null>	0,00	1	0,00	Сел	<null>	61888786	1	<nu
115	1	1	<nu	<nu	<nu	1 208,50	16.06.2017	<nu	1	1	619	32	43	<null>	46.467816	30.718394	20,00	19,75	<null>	0,00	1	0,00	<nu	<null>	61965573	1	<nu
116	1	1	<nu	<nu	<nu	1 978,73	16.06.2017		1	1	618	33	18	<null>	46.448347	30.717271	20,83	20,48	<null>	0,00	1	0,00	Рем	<null>	61899054	1	<nu
117	1	1	<nu	<nu	<nu	1 963,10	16.06.2017	<nu	1	1	618	33	24	<null>	46.448577	30.716822	20,94	20,50	<null>	0,00	1	0,00	<nu	<null>	61888839	1	<nu
118	1	1	<nu	<nu	<nu	971,70	16.06.2017	<nu	1	1	619	34	14	B	46.430387	30.748335	20,21	19,88	<null>	0,00	1	0,00	<nu	<null>	61965567	1	<nu
119	1	1	<nu	<nu	<nu	1 515,60	16.06.2017	<nu	1	1	619	35	74	78	46.474495	30.731519	22,76	22,38	<null>	0,00	1	0,00	<nu	<null>	61965559	1	<nu
120	2	1	<nu	<nu	<nu	969,20	16.06.2017	<nu	1	1	780	36	16	<null>	46.457996	30.757103	22,66	22,67	<null>	0,00	2	0,00	<nu	<null>	78025494	1	<nu
121	2	1	<nu	<nu	<nu	1 078,20	16.06.2017	050	1	1	780	33	26	<null>	46.448832	30.714972	21,35	21,02	<null>	0,00	2	0,00	097	<null>	78025495	1	<nu
122	2	1	<nu	<nu	<nu	571,50	16.06.2017	<nu	1	1	780	37	24	A	46.591034	30.785319	20,82	20,50	<null>	0,00	2	0,00	<nu	<null>	78002117	1	<nu
123	2	1	<nu	<nu	<nu	608,04	15.06.2017	<nu	1	1	781	23	22	<null>	46.399328	30.718897	0,00	0,00	<null>	0,00	2	0,00	<nu	<null>	78127730	1	<nu
124	2	1	<nu	<nu	<nu	591,92	16.06.2017	<nu	1	1	781	38	77	<null>	46.40338	30.710597	20,67	20,39	<null>	0,00	2	0,00	<nu	<null>	78127729	1	<nu
125	1	1	<nu	<nu	<nu	1 410,80	16.06.2017	<nu	1	1	618	2	118	2	46.38579	30.720029	19,41	19,13	<null>	0,00	1	0,00	<nu	<null>	61888787	1	<nu
126	1	1	<nu	<nu	<nu	1 524,10	16.06.2017	<nu	1	1	618	10	8	1	46.401435	30.712708	18,84	18,94	<null>	0,00	1	0,00	<nu	<null>	61899056	1	<nu

Рисунок 3.1 – Таблиця лічильників

На рис. 3.2 робота з таблицею всіх вулиць на яких стоять лічильники міста Одеса.

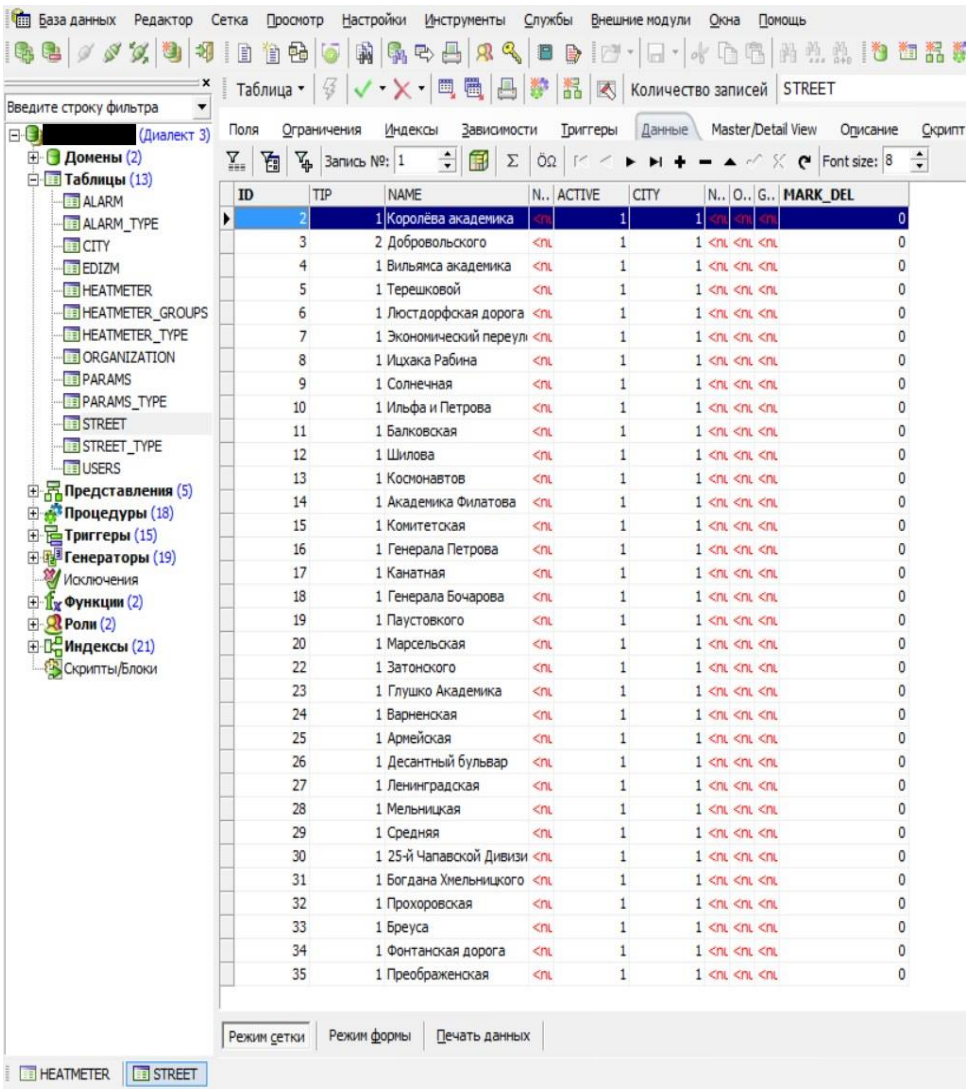


Рисунок 3.2 – Таблиця вулиць

4 РОЗРОБКА ІНФОРМАЦІЙНОГО ПОРТАЛУ ДЛЯ КП «ТМО»

Для початку демонстрація дизайну сайту, головна сторінка на рис. 4.1:

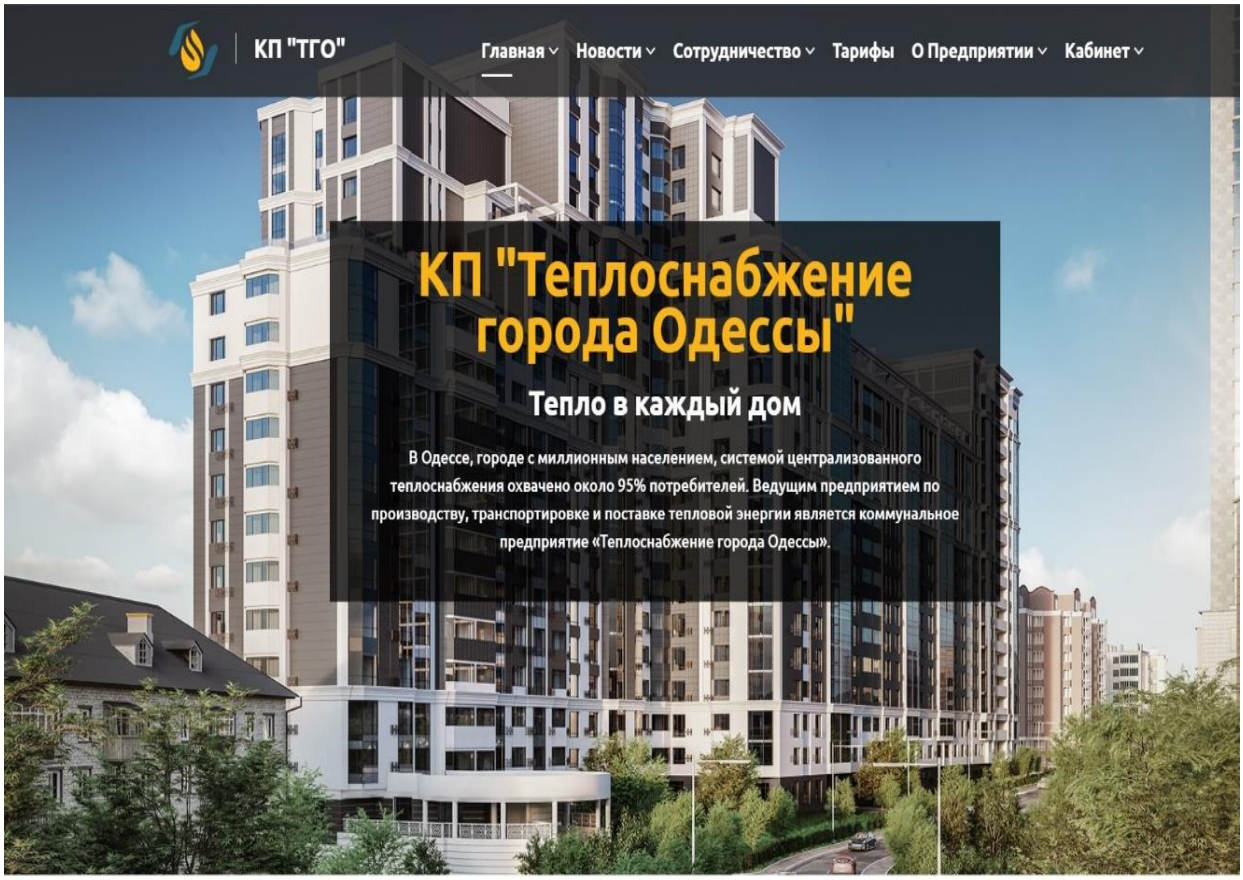


Рисунок 4.1 – Головна сторінка інформаційного порталу

У шапці сайту, на всіх сторінках, користувач бачитиме закріплене меню з випадають підкатегоріями (рис. 4.2):

На головній сторінці, в самому початку текст описує підприємство КП «ТМО», відразу після цього йде інформація з цілодобовим телефоном диспетчерської служби підприємства та телефон єдиного інформаційного центру підприємства, після цього йде коротка секція новин, в яку по-падають останні чотири додані на сайт новини.

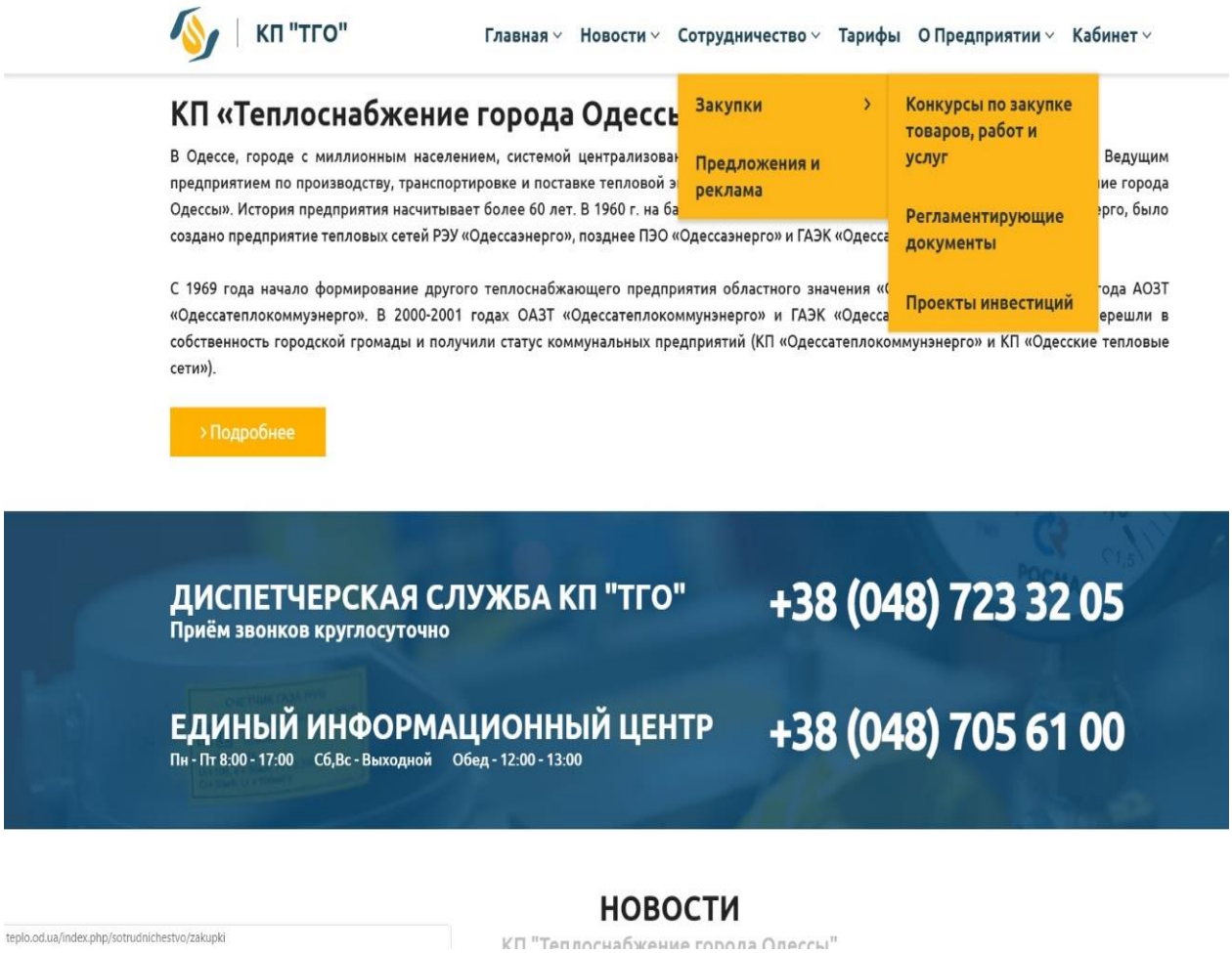


Рисунок 4.2 – Меню інформаційного порталу

Далі йде блок з написом «Згідно з постановою НКРЕКП № 464 від 30.03.2017 ...» праворуч від напису є текстове поле з написом «Ваш особовий рахунок», під текстовим полем кнопка «Отримати дані» (рис. 4.3), користувачеві пропонують ввести його особовий рахунок для ознайомлення з документами які я збираю на ходу по його особовому рахунку і повертаю користувачеві у вигляді PDF-документа з функцією друку прямо з браузера, в цьому документі багато полів, таких як опалювальна площа приміщення користувача, його адреса і споживання теплової енергі за останній повний опалювальний рік.

Результат роботи буде показаний на скріншотах нижче, для отримання PDF-документа я використовував бібліотеку FPDF.

Так само нижче присутня інформація про майбутні події і проектах які допомагають людям добитися більшої енергоефективності в різних сферах життя (рис. 4.4).

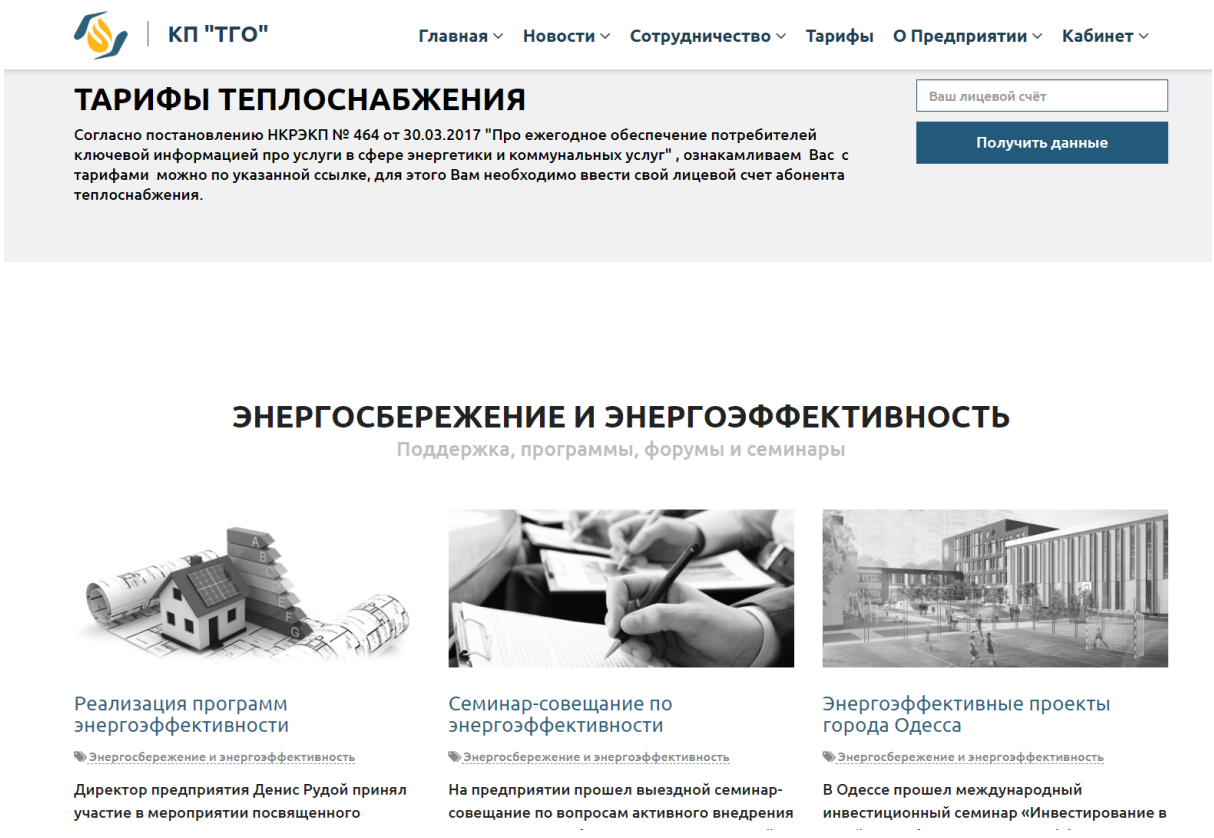


Рисунок 4.3 – Головна сторінка інформаційного порталу (2 частина)

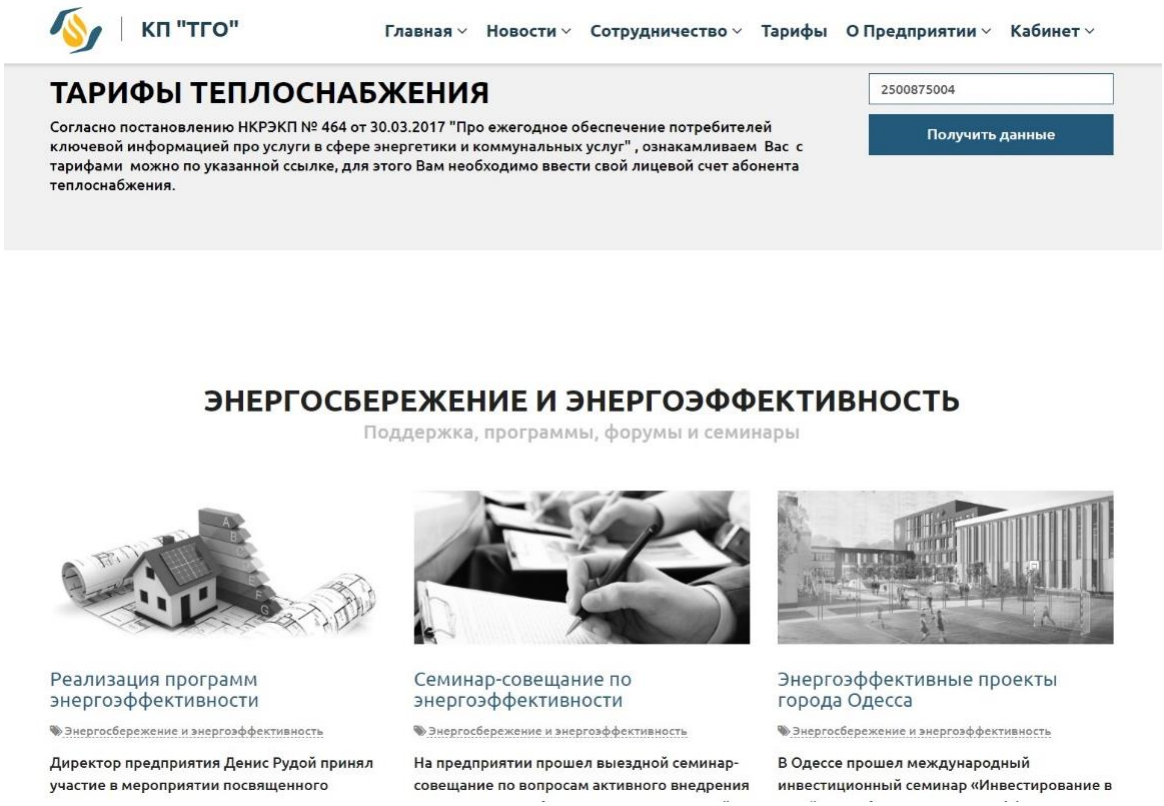


Рисунок 4.4 – Головна сторінка інформаційного порталу (3 частина)

Ввели особовий рахунок і натискаємо на кнопку «Отримати дані» і ми автоматически переходимо в нове відкрите вікно з PDF-документом документом рис. 4.5 та 4.6:

Адреса споживача	О/р (особистий рахунок)	Виконавець послуг
Люстдорфская дорога, д.123 корп. 2, кв.4	2500875004	КП «Теплопостачання міста Одеси»

Шановний споживачу!

Висловлюємо Вам подяку, що протягом минулого року користувалися послугами нашого підприємства, та вважаємо своїм обов'язком згідно з постановою НКРЕКП від 30.03.2017 № 464 довести до Вашого відома інформацію, а саме:

Інформація щодо споживання споживачем послуг з централізованного опалення та постачання гарячої води у 2017 році

	Фізичні одиниці		Грошові одиниці
Централізоване опалення	5.314541 Гкал	кВт * год	4698.89 грн
у тому числі на 1 м кв. опалювальної площі	.086135 Гкал/м кв	кВт * год/м кв.	X
Централізоване постачання гарячої води	2 м³		78.1 грн

По Вашому домогосподарству(опалювальна площа__ м кв.) за 2017 рік обсяг споживання послуг склав:

	Населений пункт	Централізоване опалення (у розрахунку на 1 м кв. опалювальної площі)						Централізоване постачання гарячої води
		для 1-2 поверхових будинків		для 3-4 поверхових будинків		для 5 і більше поверхових будинків		
		фізичні одиниці		фізичні одиниці		фізичні одиниці		фізичні одиниці
		Гкал/м кв.	кВт * год/м кв.	Гкал/м кв.	кВт * год/м кв.	Гкал/м кв.	кВт * год/м кв.	м куб.
Середнє по виконавцю послуг	м. Одеса	0.01917		0.01618		0.01256		

При цьому середнє річне споживання у 2017 році за приладами обліку склало:

Структури діючих одностаткових тарифів з урахуванням податку на додану вартість складають:

Елементи тарифу	Централізоване опалення				Централізоване постачання гарячої води, грн/куб м			
	грн/Гкал	доля %	год/м кв.	доля %	з рушникосушником		без рушникосушника	
Разом, у т. ч.:	**1 229,20**	100,00%	**30,76**	100,00%	75,12	100,00%	**69,48**	100,00%
паливна складова	866,60	70,5%	21,69	70,5%	43,74	58,2%	43,74	62,9%
решта без паливної складової	362,6	29,5%	9,07	29,5%	31,38	41,8%	25,74	37,1%
Частка витрат у структурі тарифів, яка має бути спрямована на фінансування інвестиційної програми								
Фінансування ІП								

Загальна протяжність теплових мереж(в однотрубному вимірі)*,км	1557,53	У тому числі протяжність мереж, що експлуатуються понад 25 років*,км	856,6	Рівень необхідних інвестицій для заміни трубопроводів, що експуатуються понад 25 років*, млн грн.	632,5	Термін реалізації заходів по заміні трубопроводів, що експуатуються понад 25 років за умови фінансування цих заходів на рівні передбаченому діючимтарифом*, років	---

*інформація заповнюється у разі здійснення виконавцем послуг ліцензійної діяльності з транспортування теплової енергії.

Вартість послуги з централізованного опалення в порівнянні з іншими країнами з податками, грн/Гкал

Болгарія	Польща	Корея	США	Австрія	Румунія	Фінляндія	Словенія	Латвія	Франція
1288,6	1292,2	1469,2	1667,9	1681,2	1751	1782,3	1812,4	1848,6	2201,4

Під час порівняння використанні статистичні данні офіційного курсу НБУ станом на 20 лютого 2017року, що становив 2878,3 за 100 євро. Данні отриманні з офіційного сайту НКРЕКП. **Динаміка фактичної тривалості опалювального періоду(ОП) та температури зовнішнього повітря(м. Одеса)**

Найменування показників	Січень 2017	Лютий 2017	Березень 2017	Квітень 2017	Жовтень 2017	Листопад 2017	Грудень 2017	Разом
фактично за 20__ рік								
Кількість діб ОП	31	28	31	0	7	30	31	158
Середня температура зовнішнього повітря за ОП, С	-3,3	+0,3	+6,7	0	+8,4	+7,0	+5,4	+3,5
Споживання теплової енергії, Гкал	349641	273274	201055	0	14177	174130	226740	1239017

Рисунок 4.5 – Отриманий PDF-документ (1 частина)

Вартість послуги з централізованного опалення в порівнянні з іншими країнами з податками, грн/Гкал

Болгарія	Польща	Корея	США	Австрія	Румунія	Фінляндія	Словенія	Латвія	Франція
1288,6	1292,2	1469,2	1667,9	1681,2	1751	1782,3	1812,4	1848,6	2201,4

Під час порівняння використанні статистичні данні офіційного курсу НБУ станом на 20 лютого 2017року, що становив 2878,3 за 100 євро. Данні отриманні з офіційного сайту НКРЕКП. **Динаміка фактичної тривалості опалювального періоду(ОП) та температури зовнішнього повітря(м. Одеса)**

Найменування показників	Січень 2017	Лютий 2017	Березень 2017	Квітень 2017	Жовтень 2017	Листопад 2017	Грудень 2017	Разом
фактично за 20__ рік								
Кількість діб ОП	31	28	31	0	7	30	31	158
Середня температура зовнішнього повітря за ОП, С	-3,3	+0,3	+6,7	0	+8,4	+7,0	+5,4	+3,5
Споживання теплової енергії, Гкал	349641	273274	201055	0	14177	174130	226740	1239017

Стан оснащення будинковими приладами обліку теплової енергії по виконавцю послуг складає %.
Стан оснащення будинковими приладами обліку теплової енергії по Україні складає %.

Наявність будинкового приладу обліку теплової енергії (так/ні)	Так	Наявність квартирного приладу обліку теплової енергії (так/ні)	Ні	Наявність квартирного приладу обліку гарячої води (так/ні)	Так
	X	термін наступної повірки		термін наступної повірки	31-JAN-19

З планами державного контролю та результатами перевірок ліцензійної діяльності можна ознайомитись на офіційному веб-сайті НКРЕКП www.nerc.gov.ua.

Ключові нормативно-правові акти:

постанова КМУ від 21.07.2005.No630 «Про затвердження правил надання послуг з централізованого опалення, постачання холодної та гарячої води і водовідведення»(у редакції постанови КМУ від 30.10.2015 No1037);

постанова КМУ від 30.11.2016 No 865 «Про особливості нарахування плати за надану послугу з централізованого опалення у разі відсутності у квартирі (будинку садибного типу) та на вводах у багатоквартирний будинок засобів обліку теплової енергії в опалювальний сезон 2016/2017року)»;

постанова КМУ від 19.10.2016 No744 «Про зменшення фінансового навантаження на споживачів щодо оплати послуги з централізованого опалення шляхом створення умов для отримання розстрочки на оплту послуги з централізованого опалення».

З повагою Комунальне підприємство «Теплопостачання міста Одеси».

Рисунок 4.6 – Отриманий PDF-документ (2 частина)

Так само на головній сторінці в нижній частині дві секції:

1) це карта районних центрів для наших клієнтів, куди можна звернути з будь якими питаннями;
2) найнижча частина сторінки, яка розбита на 4 блоки, перший блок – Контактна інформація, з телефонами які були перераховані вище, і адресу головного відділення міста Одеси; другий блок – «Компанія», посилання на сторінки Про компанію, Керівництво, Розрахункові рахунки і Новости; третій блок з посиланнями для абонентів Тарифи на послуги, Особистий кабінет, Інформація; четвертий блок є пошуком по сайту інформації яка необхідна кінцевому користувачеві;

Все це показано на скріншоті нижче рис. 4.7.

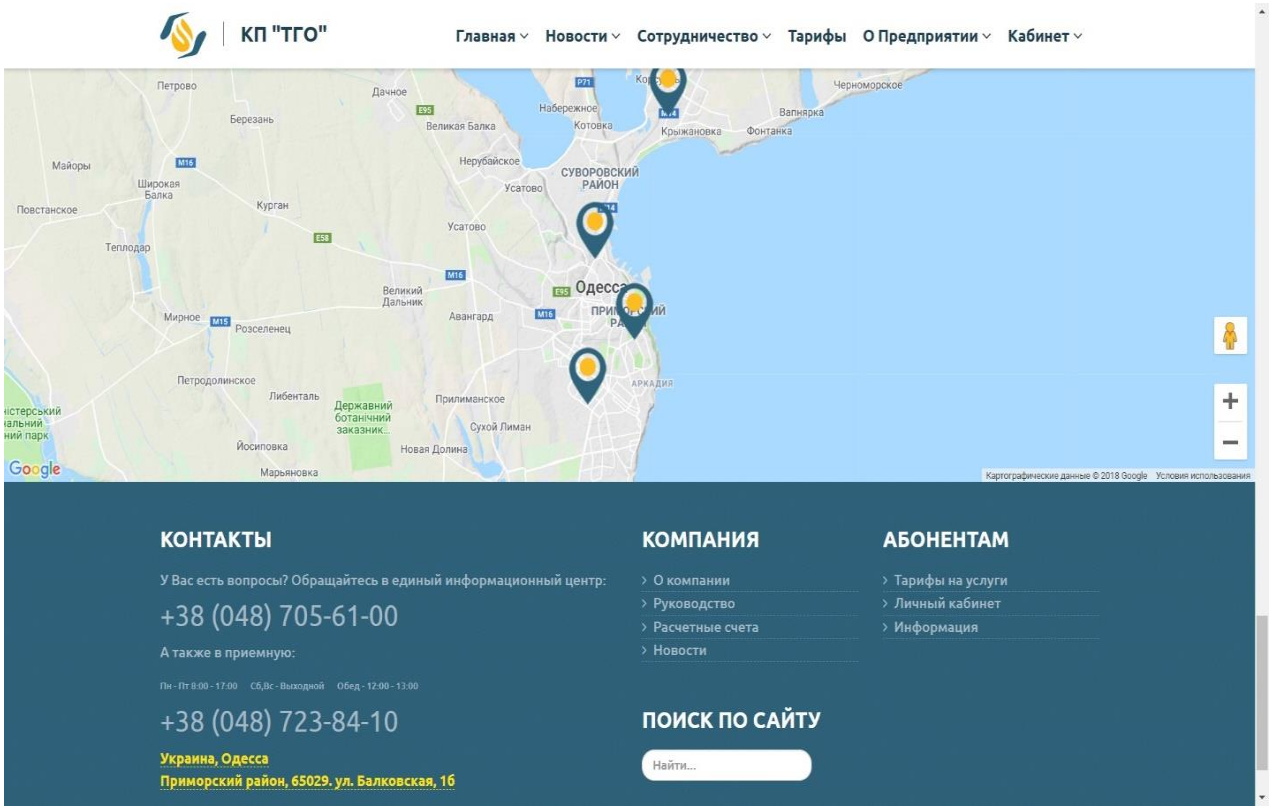


Рисунок 4.7 – Головна сторінка інформаційного порталу (4 частина)

ВИСНОВКИ

Дослідження предметної області, безумовно, є важливим для кожної сфери діяльності, особливо в тому випадку, якщо предметну область потрібно проаналізувати та виявити недоліки для їх усунення.

КП «ТМО» як один з найкрупніших постачальників теплової енергії по місту Одеса, виконує великий обсяг рутинної роботи. В основному це контроль температури та відстеження її зміни протягом опалювального періоду.

В ході магістерської роботи був створений інформаційний портал з безліччю баз даних в які вносяться показання всіх лічильників, які в автоматизованому режимі передають дані до БД, інформаційний портал отримує ці дані з БД і виводить їх на сайт, тим самим спрощуючи та прискоруючі процес, який раніше вівся тільки у відділеннях прийому користувачів послуг теплопостачання міста Одеси і займав у багато разів довший період часу, ніж зараз.

В роботі були розроблені наступні елелменти:

- комплекс баз данних для особистого кабінету користувачів сайту MySQL;
- комплекс баз данних для сбору та зберігання інформації по всім користувачам послуг підприємства(Oracle DataBase);
- Веб-сайт КП «ТМО» для зручності і швидкості роботи з Базою даних.

ПЕРЕЛІК ПОСИЛАНЬ

1. Правильная структура сайта. [Электронный ресурс]: – Режим доступа: http://yrokiwp.ru/seo/vnutrennjaja-optimizacija-sajta/struktura-sajta-primer-shema/.
2. Ларман К. Применение UML 2.0 и шаблонов проектирования = Applying UML and Patterns : An Introduction to Object-Oriented Analysis and Design and Iterative Development. – 3-е изд. – М.: Вильямс, 2006. – 736 с.
3. Создание таблицы в режиме конструктора [Электронный ресурс]. – Режим доступа: http://www.officepack.ru/access/11/, свободный. – Название с экрана.
4. Ленгсторф Д. PHP и jQuery для профессионалов = Pro PHP and jQuery. – М.: «Вильямс», 2010. – 362 с.
5. Босова И.Н., Разработка образовательных программ по информатике на вариативной основе и индивидуальных учебных планов. Информационные технологии в образовании – 2009 : Сборник научных трудов участников IX научно-практической конференции-выставки 29 — 30 октября 2009 г.. – Ростов-на-Дону: Ростиздат, 2009. – 525 с.
6. Рейсиг. Д., Инструменты отладки и тестирования JavaScript. Профессиональные приёмы программирования – Pro JavaScript™ Techniques / Перевод Н. Вильчинский. – СПб.: БХВ-Петербург, 2008. – 629 с.
7. Фримен А. jQuery для профессионалов = Pro jQuery. – М.: «Вильямс», 2012. – 960 с.
8. Шафер С. HTML, XHTML и CSS. Библия пользователя, 5-е издание = HTML, XHTML, and CSS Bible, 5th Edition. – М.: «Диалектика», 2010. – 656 с.
9. Кузнецов М. В., Симдянов И. В. MySQL5.– СПб.: БХВ-Петербург, 2006. 1024 с.
10. Компания MySQL AB. MySQL. Справочник по языку – М.: Издательский дом "Вильямс", 2005. – 432 с.
11. Ульман Л. MySQL– М.: ДМК Пресс; СПб.: Питер, 2004. – 352 с.
12. Васвани. В. А. MySQL: использование, администрирование MySQL Database Usage & Administration. – М.: «Питер», 2011. – 368 с.
13. Рыбанов А. А., Средства автоматизированного проектирования баз данных [Электронный ресурс] / А. А. Рыбанов. –Режим доступа:

http://window.edu.ru/window_catalog/redir?id=
47119&file=rybanov_bd.pdf, свободный. – Название с экрана.

14. Котеров Д. В., Костарев А. Ф. PHP5. – СПб.: БХВ-Петербург, 2005. – 1120 с.
15. Ткаченко В. А. , Системы управления базами данных и экспертные системы [Электронный ресурс] / В.А. Ткаченко. – Режим доступа: http://www.lessons-tva.info/edu/e-inf2/m2t4.html, свободный. – Название с экрана.

ДОДАТОК А
КОД ПРОГРАМИ

```
<?php
/**
* @package   Warp Theme Framework
* @author    YOOtheme http://www.yootheme.com
* @copyright Copyright (C) YOOtheme GmbH
* @license   YOOtheme Proprietary Use License
(http://www.yootheme.com/license)
*/

// no direct access
defined('_JEXEC') or die;

//JHtml::_('behavior.mootools');
?>


<style>
/* === Remove input autofocus webkit === */
*:focus {outline: none;}
body{
        background: #062a8429 ;
}
/* === Form Typography === */
body {font-family: Ubuntu, sans-serif;}
.contact_form h2, .contact_form label {font-family: Ubuntu, sans-serif;}
.form_hint, .required_notification {font-size: 11px;}
.new_class{
        text-align: -webkit-center;
        margin-top: -75px;
}
/* === List Styles === */
.contact_form ul {
    width: fit-content;
    list-style-type:none;
        list-style-position:outside;
        margin:0px;
        padding:0px;
}
.contact_form li{
        padding: 8px;
        position:relative;
}
.contact_form li:first-child{
        padding: 25px;
        position:relative;
}
.contact_form li:first-child{
        border-bottom:1px solid #373b3b;;
}

/* === Form Header === */
.contact_form h2 {
        margin:0;
        display: inline;
        font-weight: 550;
    color: #292e2eeb;
}
.required_notification {
        color:#d45252;
```

```
        margin:5px 0 0 0;
        display:inline;
        float:right;
}

/* === Form Elements === */
.contact_form label {
        width:160px;
        margin-top: 3px;
        display:inline-block;
        padding:3px;
        font-weight: bold !important;
        color: #373b3b;
}
.contact_form input {
        height:20px;
        width:220px;
        padding:5px 8px;
        height: 35px !important;
}
.contact_form textarea {padding:8px; width:300px;}

        /* form element visual styles */
        .contact_form input, .contact_form textarea {


                padding-right:30px;
                -moz-transition: padding .25s;
                -webkit-transition: padding .25s;
                -o-transition: padding .25s;
                transition: padding .25s;
        }
        .contact_form input:focus, .contact_form textarea:focus {
                background: #fff;

                padding-right:70px;
        }

/* === HTML5 validation styles === */
/*.contact_form input:required, .contact_form textarea:required {
        background: #fff url(/lichnii_cab/images/red_asterisk.png) no-repeat
98% center;*/
}
.contact_form input:required:valid, .contact_form textarea:required:valid {
        background: #fff url(/lichnii_cab/images/valid.png) no-repeat 98% cen-
ter;

}
.contact_form input:focus:invalid, .contact_form textarea:focus:invalid {
        background: #fff url(/lichnii_cab/images/invalid.png) no-repeat 98%
center;

}

/* === Form hints === */
.form_hint {
        font-size: 13px;
        background: #d45252;
        border-radius: 3px 3px 3px 3px;
        color: white;
        margin-left:8px;
        padding: 1px 6px;
        z-index: 999; /* hints stay above all other elements */
```

```
        position: absolute; /* allows proper formatting if hint is two lines
*/
        display: none;
}
.form_hint::before {
        /*content: "\25C0";*/
        color:#d45252;
        position: absolute;
        top:1px;
        left:-6px;
}
.contact_form input:focus + .form_hint {display: inline;}
.contact_form input:required:valid + .form_hint {background: #28921f;}
.contact_form input:required:valid + .form_hint::before {color:#28921f;}

/* === Button Style === */
button.submit {
        background-color: #2f617a;
        border: 1px solid #2f617a;
        border-bottom: 1px solid #2f617a;
        color: white;
        font-weight: bold;
        padding: 6px 20px;
        text-align: center;
        text-shadow: 0 -1px 0 #396715;
}
button.submit:hover {
        opacity:.65;
        cursor: pointer;
}

#submit:disabled {
    background: -webkit-linear-gradient(top, #c60000, #8f0505);
    border: 0px;
        cursor: not-allowed;
        border: 1px solid #509111;
        border-bottom: 1px solid #5b992b;
   }

</style>


<script src='https://www.google.com/recaptcha/api.js'></script> <!-- Google
recaptcha -->

<div id="system">
<div class="new_class">
<form class="contact_form" action="#" method="post" name="contact_form">
    <ul>
        <li>

             <!--<span class="required_notification">*
          </span>-->
        </li>
        <li>
            <label for="name">   :</label>
            <input type="text" name="FIO"  placeholder="                    "
required />
            <span class="form_hint">            ,               </span>
        </li>
               <li>
            <label for="login">     :</label>
            <input type="text" name="login" id="login" placehold-
er="ivanov12345" required />
```

```
            <span id="loginMessage" class="form_hint">                </span>
        </li>
        <li>
            <label for="email">Email:</label>
            <input type="email" name="email" id="email" placehold-
er="email@example.com" required />
            <span id="emailMessage" class="form_hint">
     </span>
        </li>
               <li>




        </li>
        <li>
            <label for="password">      :</label>
            <input type="password" name="password" class="password" required
/>
            <span class="form_hint">        6         </span>
        </li>

               <li>
            <label for="password_repeat">                :</label>
            <input type="password" name="password_repeat"
class="password_repeat" required />
            <span class="form_hint">        6         </span>
        </li>
                        <label class="error" ></label>
                        <div class="g-recaptcha" data-
sitekey="6LekyWoUAAAAAGu27H07RvcdtFr7pA7GBzTOjjzu" data-
callback="enableBtn"></div>
        <li>
               <button class="submit" id="submit" type="submit"
id="button1">                  </button>
        </li>
    </ul>
</form>
</div>



<?php
$z=$_SERVER['DOCUMENT_ROOT'];
include("$z"."/user_cab/passw.php");

if(isset($_POST['login'])){

        // Captcha ckeck
        if(isset($_POST['login'])) {
            $url = 'https://www.google.com/recaptcha/api/siteverify';
            $key = '6LekyWoUAAAAAKB4hxZhPaF4sRkVL-F1iastFxmu';
            $query = $url.'?secret='.$key.'&response='.$_POST['g-recaptcha-
response'].'&remoteip='.$_SERVER['REMOTE_ADDR'];

            $data = json_decode(file_get_contents($query));
            if ($data->success === false){
                        echo "<script>
                     alert('             ');
                     </script>";
               exit();
            }
        }
```

```
        $login=$_POST['login'];

        $password=$_POST['password'];

        $password_repeat=$_POST['password_repeat'];

        $fio=$_POST['FIO'];

        $email=$_POST['email'];

        $ip=$_SERVER['REMOTE_ADDR'];


        if(trim($login)==''){
                echo "<script>alert('                    !')</script>";;
        }

        if(trim($fio)==''){
                echo "<script>alert('                !')</script>";;
        }

        if(trim($password)==''){
                echo "<script>alert('                    !')</script>";;
        }

        if(trim($password_repeat)==''){

        }

        if(trim($email)==''){
                echo "<script>alert('                EMAIL !')</script>";;
        }

        if($password!=$password_repeat){
                echo "<script>alert('                    !')</script>";
        }

                                        -
        if(strlen($password)<6){
                                                                        -

        }

        //
        $phpass = new PasswordHash(10, true);

        $password = $phpass->HashPassword($password);


        //
        $conn_id = mysqli_connect("localhost", "new_tgo", "V1rtu@l",
"new_tgo");

        /* check connection */
        if (mysqli_connect_errno()) {
            printf("Connect failed: %s\n", mysqli_connect_error());
            exit();
        }
        mysqli_set_charset($conn_id, 'utf8');


        if(isset($_POST['email']) && $_POST['email'] != ''){
                        $email = $_POST['email'];
```

```
                $email = mysqli_real_escape_string($conn_id,$email);
            if(!filter_var($email, FILTER_VALIDATE_EMAIL)){
            echo 'valid';
            }else{
            $sql = "SELECT id FROM hz3jh_users WHERE email='$email'";
            $result = mysqli_query($conn_id,$sql);
            if(mysqli_num_rows($result) == 1){
                    echo 'valid';
            }else{
                    echo 'valid';
            }
            }
    }



    $q=mysqli_query($conn_id, "SELECT id FROM hz3jh_users WHERE
username='".$login."' OR email='".$email."'");

    if(mysqli_error($conn_id)!=''){
            echo "<script>alert('                        !')</script>";
    }

    if(mysqli_num_rows($q)!=0){

!')</script>";
    }
    else{
            unset($q);


                                                    -
            $email_cnx=explode("@",$email);


            $check-
Sum=base64_encode(substr($login,0,3).$email_cnx[0].md5($login));
            $q=mysqli_query($conn_id, "INSERT INTO hz3jh_users (ac-
count,surname,name,patronymic,username,email,password,block,sendEmail,activat
ion,params,registerDate,lastvisitDate,lastResetTime,resetCount, otpKey, otep,
requireReset)
            VALUES
('','','$fio','','$login','$email','$password','1','0','$checkSum','{}',now()
,now(),now(),'0', '', '', '0')");
            $get_id=mysqli_query($conn_id, "SELECT id FROM hz3jh_users
WHERE email='$email'");
            $get_id_fetch=mysqli_fetch_row($get_id);
            $new_id=$get_id_fetch[0];
            $group_id=2;
            $qu=mysqli_query($conn_id, "INSERT INTO
hz3jh_user_usergroup_map (user_id, group_id) VALUES('$new_id','$group_id')");
            $que=mysqli_query($conn_id, "INSERT INTO hz3jh_user_profiles
(user_id, profile_key, profile_value, ordering)
            VALUES('$new_id','profile.phone','\"\"','1')");


            //$z=mysqli_query($conn_id, "UPDATE hz3jh_users SET
block='0',activation='0' WHERE hz3jh_users.email='$email'");
            $error_messs=mysqli_error ($conn_id);
            if(mysqli_error($conn_id)!=''){
                    echo "<br/>$error_messs<br/>";
                    //echo "<script>alert('
  141!')</script>";
            }
```

```
                else{

                        /**

                                                                -



                        **/

                                                                    -



                        $date=time();
                        $data_mail=date('d.m.Y',$date);
                        unset($q);


        $href_link="https://teplo.od.ua/user_cab/activate.php?checkSum=$checkS
um&email=$email";









                        $href_link









                        $z=$_SERVER['DOCUMENT_ROOT'];
                        require "$z".'/user_cab/PHPMailer-
master/src/PHPMailer.php';

                        $mail = new PHPMailer;

                        //
                        //0 = off (for production use)


                        $mail->SMTPDebug = 0;
```

```
                        $mail->CharSet = 'UTF-8';
                        $mail->isSendmail();
                        $mail->Host = '192.168.10.1'; //gmail: smtp.gmail.com
                        $mail->SMTPAuth = '1';
                        $mail->Username = 'reply@teplo.od.ua';    //
                        $mail->Password = 'noreply';                    //
                        $mail->SMTPSecure = 'none';
                        $mail->Port = 587;
                        $mail->AuthType='PLAIN';
                        $mail->setLanguage('ru');
                        //$mail->AddReplyTo('replyto@email.com', 'Reply to
name');
                        $mail->setFrom('reply@teplo.od.ua', 'Teplo.od.ua');
                        $mail->addAddress("$email");


                        $mail->Subject = '                  ';
                        $mail->Body    = $body;
                        $mail->isHTML(true);
                        $mail->AltBody = $alt_body;

                        //
                        if(!$mail->send()) {
                                                                $mail-
>ErrorInfo;
                        }
                        else {
                           echo "
                                <script>
                                alert('                    ');
                                document.location.href = '/index.php/kabinet'
                                </script>";
                        }
                }
        }
}
?>



</div>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></scri
pt>
<script>
        $(document).ready(function() {
        //$('#login').bind("change keyup input click", function() {
        //              if (this.value.match(/[^0-9a-zA-Z_-]/g)) {
        //              this.value = this.value.replace(/[^0-9a-zA-Z_-]/g, '');
        //              }
        //      });

                /*$(document).ready(function(){
                        var pattern = /^[a-z0-9_-]+@[a-z0-9-]+\.([a-
z]{1,6}\.)?[a-z]{2,6}$/i;
                        var mail = $('#mail');

                        mail.blur(function(){
                                if(mail.val() != ''){
                                        if(mail.val().search(pattern) == 0){
                                                $('#valid').text('        ');
                                                $('#submit').attr('disabled',
false);
```

```
        mail.removeClass('error').addClass('ok');
                                        }else{

                                                $('#submit').attr('disabled',
true);
                                                mail.addClass('ok');
                                        }
                                }else{
                                        $('#valid').text('     e-mail
           !');
                                        mail.addClass('error');
                                        $('#submit').attr('disabled', true);
                                }
                        });
                });*/
        })

                document.getElementById("submit").disabled = true;
        function enableBtn(){
        document.getElementById("submit").disabled = false;
    }

        $('.email').on('keyup', function(){
                var reg = /^([A-Za-z0-9_\-\.])+\@([A-Za-z0-9_\-\.])+\.([A-Za-
z]{2,4})$/;
                var address = $(".email").val();


                if(reg.test(address) == false) {
                        $('.error').attr('display', 'block');

                        $('#submit').attr('disabled', 'disabled !important');
                } else {
                        $('.error').attr('display', 'none');
                        $('#submit').removeAttr('disabled');
                        $('.error').html('');
                }
        });
        $('.login').on('keyup', function(){
                var value_input2 = $(".login").val();
                if(value_input1 != value_input2) {
                        $('.error').attr('display', 'block');

                        $('#submit').attr('disabled', 'disabled !important');
                } else {
                        $('.error').attr('display', 'none');
                        $('#submit').removeAttr('disabled');
                        $('.error').html('');
                }
        });
        $('.FIO').on('keyup', function(){
                var value_input2 = $(".FIO").val();
                if(value_input1 != value_input2) {
                        $('.error').attr('display', 'block');

                        $('#submit').attr('disabled', 'disabled !important');
                } else {
                        $('.error').attr('display', 'none');
                        $('#submit').removeAttr('disabled');
                        $('.error').html('');
                }
        });
```

```
        $('.password_repeat').on('keyup', function(){ //
                        2-
                var value_input1 = $(".password").val(); //
    1-
                var value_input2 = $(".password_repeat").val(); //
            2-


                        $('.error').attr('display', 'block');


                        $('#submit').attr('disabled', 'disabled !important');
//
                } else { //        ,
                        $('.error').attr('display', 'none');
                        $('#submit').removeAttr('disabled');  //


                        document.getElementById("submit").disabled = true;
                        function enableBtn(){
                        document.getElementById("submit").disabled = false;
                        }
                }
        });

        $('.password').on('keyup', function(){ //
                 2-
                var value_input1 = $(".password_repeat").val(); //
            1-
                var value_input2 = $(".password").val(); //
    2-


                        $('.error').attr('display', 'block');




                } e
                        $('.error').attr('display', 'none');



                        document.getElementById("submit").disabled = true;
                        function enableBtn(){
                        document.getElementById("submit").disabled = false;
                        }
                }
        });


$(document).ready(function(){

  var email = '';
  var login = '';
```

```
  $('#email').keyup(function(){
      var value = $(this).val();

      $.ajax({
          type:'POST',
          url:'/user_cab/chekLoginEmail.php',
          data:'email='+value,
          success:function(msg){

            if(msg == 'emailValid'){
              $('#emailMessage').html('     Email                    ');
              $('#emailMessage').css('background-color', 'green');
              email = value;
            }else{

              $('#emailMessage').css('background-color', '#d45252');
            }
          }
        });
    });

  $('#login').keyup(function(){
      var value = $(this).val();

      $.ajax({
          type:'POST',
          url:'/user_cab/chekLoginEmail.php',
          data:'login='+value,
          success:function(msg){

            if(msg == 'loginValid'){
              $('#loginMessage').html('                                ');
              $('#loginMessage').css('background-color', 'green');
              login = value;
            }else{
              $('#loginMessage').html('                      ');
              $('#loginMessage').css('background-color', '#d45252');
            }
          }
        });
    });


});

</script>
```