Форма № Н-9.02

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ
імені адмірала Макарова
ХЕРСОНСЬКИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ

Кафедра інформаційних технологій та фізико-математичних дисциплін

"Допущений до захисту"
Завідувач кафедри
д.т.н., доцент Гучек П.Й.

"22" грудня 2023 р.

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи

на здобуття ступеня вищої освіти "магістр"

на тему «Дослідження технологій збирання та систематизації даних з Web-сайтів»

Виконав: студент VI курсу, групи 6157м спеціальності
122 - "Комп'ютерні науки", ОПП - "Інформаційні управляючі системи та технології"
(шифр і назва спеціальності та освітньо-професійної програми)

Кухаренко О.О.
(прізвище та ініціали)

Керівник Гайдаєнко О.В.

Рецензент Новіков Ю.М.

22.12 - 2023

Форма № Н-9.01

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ

імені адмірала Макарова
ХЕРСОНСЬКИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
Кафедра інформаційних технологій та фізико-математичних дисциплін

Освітній ступень магістр
Галузь знань 12 - "Інформаційні технології"
(шифр і назва)
Спеціальність 122 - " Комп'ютерні науки"
(шифр і назва)
Освітньо-професійна програма "Інформаційні управляючі системи та технології"

ЗАТВЕРДЖУЮ
Гарант освітньої програми
Гучек П.Й.
"23" жовтня 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ
на здобуття ступеня вищої освіти «магістр»

Студенту Кухаренко Олександр Олександрович
(прізвище, ім'я, по батькові)

1. Тема дипломної роботи: Дослідження технологій збирання та систематизації даних з Web - сайтів

Керівник роботи Гайдаєнко Оксана Володимирівна, к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Тема затверджена розпорядженням по інституту від "10" жовтня 2023 року № 19

2. Термін подання студентом роботи 11.12.2023 р.

3. Вихідні дані до роботи:

ДСТУ щодо обробки інформації, літературні джерела, технічна документація та розробка інформаційної системи збирання систематизації. Аналіз технологій збирання даних. Аналіз вимог до побудови архітектури інформаційної системи.

Розробка проектних рішень.

4. Зміст та структура розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):

4.1 Титульний аркуш, завдання на магістерську роботу, анотація (українською, російською, англійською), зміст, перелік умовних означень, символів, одиниць та термінів (при необхідності).

4.2 Вступ.

4.3 Дослідження технологій проектування інформаційних систем та аналіз їх особливостей.

4.4 Формування вимог до інформаційної системи, розробка її концепції і постановка задачі.

4.5 Розробка проектних рішень по інформаційній системі

4.6 Реалізація проекту інформаційної системи (технічна документація по проекту).

4.7 Розділ з економіки

4.8 Розділ з охорони праці та безпеки в надзвичайних ситуаціях (охорони оточуючого середовища).

4.9 Висновки.

4.10 Список використаної літератури.

4.11 Додатки (технічне завдання, загальний опис ІС, інструкція користувача, текст програми).

5. Перелік презентаційних матеріалів

5.1 Схема організаційної структури підприємства.
5.2 Архітектура автоматизованої ІС.
5.3 Організаційно-функціональна структура автоматизованої ІС.
5.4 Інформаційна модель програмного забезпечення автоматизованої ІС.
5.5 Модель переходів станів програмного забезпечення автоматизованої ІС.
5.6 Модель технологічного забезпечення автоматизованої ІС.
5.7 Інтерфейс програми
5.8

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
4.7	Ломоносов Дмитро Анатолійович		
4.8	Щедролосєв Олександр Вікторович		

7. Дата видачі завдання 23 жовтня 2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Строк виконання етапів роботи	Примітка
1.	Вивчення і аналіз предметної галузі	19.10.2023	
2.	Розробка концепції ІС	21.10.2023	
3.	Розробка проектних рішень по ІС	30.10.2023	
4.	Реалізація проекту, розробка робочої документації	02.11.2023	
5.	Вирішення питань з економіки	09.11.2023	
6.	Вирішення питань охорони праці та безпеки в надзвичайних ситуаціях (охорони оточуючого середовища)	16.11.2023	
7.	Виконання графічної частини роботи	23.11.2023	
8.	Оформлення пояснювальної записки	04.12.2023	
9.	Подання роботи на попередній захист	11.12.2023	
10.	Подання роботи рецензенту	17.12.2023	
11.	Подання роботи на державний захист	18.12.2023	

Студент Кухаренко О.О.
Прізвище та ініціали

Керівник роботи Гайдаєнко О.В.
Підпис Прізвище та ініціали

Анотація

Дана дипломна робота присвячена Дослідженню технологій збирання та систематизації Даних з Web-сайтів**.** У даній роботі проведений аналіз сучасних інформаційних технологій, які використовують для розробки інформаційних систем; сформовані вимоги до інформаційної системи; розроблена її концепція; проведене техніко-економічне обґрунтування розробки системи; розроблене технічне завдання; розроблені загальносистемні рішення, рішення з інформаційного, організаційного, технічного та програмного забезпечення; розроблена відповідна документація. Також у роботі виконані розділи з економіки, охорони праці, охорони навколишнього середовища.

Робота виконана на 106 сторінках машинописного тексту, містить 15 рисунків, 22 таблиці, 3 додатки і список використаних джерел з 14 найменувань.

Ключові слова: веб-скрапінг, збирання даних, систематизація даних, обробка веб-даних, парсинг веб-сторінок, аналіз даних, інформаційні системи, автоматизація збору даних, структуризація інформації, веб-технології.

Abstract

This thesis is dedicated to the Research of technologies for collecting and systematizing data from websites. In this work, an analysis of modern information technologies, which are used for the development of information systems, is carried out; formed requirements for the information system; its concept was developed; carried out technical and economic substantiation of system development; developed technical task; developed system-wide solutions, information, organizational, technical and software solutions; appropriate documentation has been developed. The work also includes chapters on economics, labor protection, and environmental protection.

The work is completed on 106 pages of typewritten text, contains 15 figures, 22 tables, 3 appendices and a list of used sources from 14 titles.

Keywords: web scraping, data collection, data systematization, web data processing, web page parsing, data analysis, information systems, automated data collection, information structuring, web technologies.

ЗМІСТ

ВСТУП

Інформаційні технології є найбільш важливою складовою процесу використання інформаційних ресурсів суспільства. До теперішнього часу вони пройшли кілька еволюційних етапів, зміна яких визначалася головним чином рівнем розвитку науково-технічного прогресу, появою більш досконалих засобів обробки інформації. Сучасний науково-технічний прогрес немислимий без широкого впровадження і використання обчислювальної техніки в освіті, виробництві, управлінні та наукових дослідженнях. Сьогодні на основі засобів обчислювальної техніки розробляються і впроваджуються різні автоматизовані інформаційні системи (управління, проектування, технологічної підготовки виробництва). Успіх у впровадженні цих систем, їхня роль в інтенсифікації розвитку народного господарства нашої країни багато в чому залежить від фахівців із системотехніки, які знають методику аналізу і проектування цих систем, можливості обчислювальної техніки і які володіють математичними методами, що використовуються при постановці та вирішенні задач.

Метою даної роботи є підвищення ефективності збору та систематизації даних з веб-сайтів розробка проекту інформаційної системи «ТОВ».

Для досягнення цієї мети треба вирішити наступні задачі:

- дослідити діючу інформаційну систему, яка не є автоматизованою;
- виявити її недоліки;
- сформулювати вимоги до автоматизованої інформаційної системи;
- розробити концепції системи, та обрати оптимальний варіант методом функціонально-вартісного аналізу;
- розробити проектні рішення по системі.
- розробити робочу документацію по підсистемі;
- розрахувати економічну ефективність від впровадження автоматизованої системи.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ТЕХНОЛОГІЙ ЗБИРАННЯ ТА СИСТЕМАТИЗАЦІЇ ДАНИХ З ВЕБ-САЙТІВ, РОЗРОБКА КОНЦЕПЦІЇ І ПОСТАНОВКА ЗАДАЧІ

1.1 Опис предметної області

Інформаційна система по вилученню даних з веб сайтів(магазини, новини, і т.д.) призначена для автоматизації роботи підприємств. Функції, що автоматизуються системою, пов'язані з великою кількістю інформації на сайтах і вилученню лише потрібної.

Інформаційна система (ІС) є середовищем, складовими елементами якого є комп'ютери, об'єднані в комп'ютерні мережі, програмні продукти, бази даних, люди, різного роду технічні і програмні засоби зв'язку і т.д. Основна мета інформаційної системи — організація збереження, обробки і передачі інформації. Інформаційна система являє собою людино-комп'ютерне оточення (систему) для обробки інформації.

ІС, що розробляється, є розподіленою, тобто для її функціонування, крім програм-клієнтів, необхідна наявність принаймні одного сервера програм. Програмне забезпечення сучасних ІС має наступні логічні рівні (див. рисунок 1.1):

- Рівень користувача (інтерфейс і сервіси користувача);
- Прикладний рівень (прикладні сервіси);
- Рівень даних (сервіси доступу до даних і сховище даних).

Опишемо функції кожного рівня:

- Рівень користувача: відображає дані і дозволяє користувачу редагувати їх. Існує два основних типи інтерфейсу: віконний (реалізується засобами підсистеми інтерфейсу користувача операційної системи) і на основі Web. Web-інтерфейси засновані на HTML, DHTML, XHTML чи XML, у результаті чого вони можуть відображатися будь-яким Web-оглядачем на будь-якій платформі.
- Прикладний рівень: тут реалізовані бізнес-правила й обмеження на дані (перевірка коректності даних). І хоча його сервіси використовуються

презентаційним рівнем, він не прив'язаний до якого-небудь клієнта – сервіси прикладного рівня доступні будь-якому клієнту. Обмеження на дані гарантують точність і цілісність інформації, що зберігається. Бізнес-правила звичайно реалізуються окремим модулем на централізованому сервері, що дає можливість доступу до нього відразу декільком клієнтам.

- Рівень даних: прикладний рівень не має інформації про те, як і де зберігаються дані, які він обробляє. У цьому питанні він покладається на сервіси доступу до даних, що виконують всю роботу з отримання і передачі даних. Сервіси доступу до даних також реалізуються у виді ізольованих модулів, які знають про місце збереження інформації. Кожен модуль доступу до даних, як правило, відповідає і за цілісність сховища (наприклад, реляційної бази даних). Для багаторівневих програм у якості сховища інформації підходять сервера баз даних (MS SQL Server, Oracle, Sybase, MySQL, PostgreSQL і ін.), необхідні для обслуговування даних у таблицях і високопродуктивної виборки інформації.

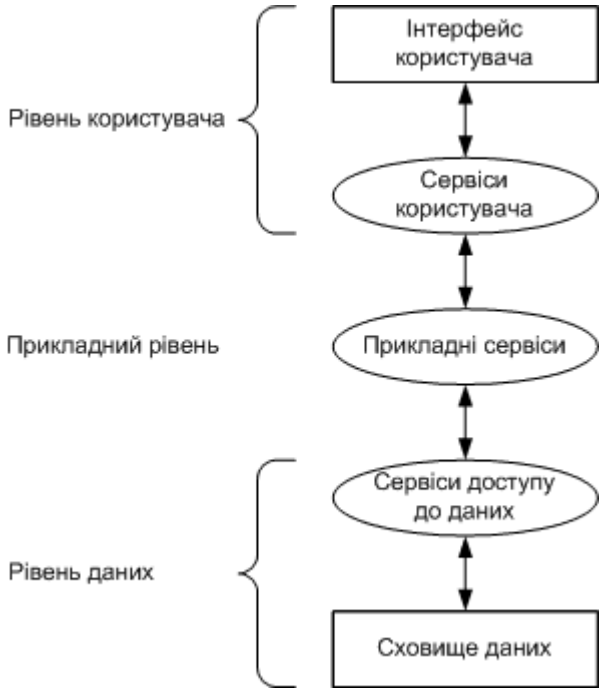

Рисунок 1.1 – Логічна архітектура програмного забезпечення інформаційної системи

Реалізація функцій інформаційної системи неможлива без знання орієнтованих на неї інформаційних технологій. Нижче представлений опис технологій, які використовуються на кожному рівні інформаційної системи, що розробляється.

1.1.1 Сучасні технології збирання та систематизації даних

Збір та систематизація даних з веб-сайтів може бути корисним з точки зору багатьох аспектів:

1. Аналіз конкурентів. Вивчення даних з веб-сайтів конкурентів може допомогти визначити їх стратегії, цінову політику, акції та інші ключові аспекти бізнесу.

2. Маркетингові дослідження. Визначення тенденцій у попиті, змінах у споживчих уподобаннях та реакції на рекламні кампанії дозволяє покращити стратегію маркетингу.

3. Цінова політика. Аналіз цінових пропозицій конкурентів або динаміки цін на ринку допомагає оптимізувати цінову політику власного бізнесу.

4. Створення контенту. Збір інформації з різних джерел дозволяє створювати актуальний та цікавий контент для аудиторії.

5. Прийняття рішень. Дані з веб-сайтів можуть бути використані для підтримки рішень в бізнесі, включаючи розробку нових продуктів, стратегічне планування та розвиток бренду.

6. Моніторинг репутації. Відстеження згадок про компанію або бренд у відгуках, відгуках на соціальних мережах чи новинах допомагає управляти репутацією.

7. Дослідження ринку. Збір даних про споживчі уподобання, поведінку покупців та інші параметри ринку допомагає адаптувати продукт чи послугу до потреб споживачів.

8. Фінансове планування. Дані про фінансовий стан компаній чи ринкові індикатори дозволяють робити прогнози та планувати фінансові стратегії.

В цілому, збір та систематизація даних з веб-сайтів дозволяє компаніям отримувати конкретну, актуальну та корисну інформацію для прийняття найкращих рішень та ефективного управління.

Збір і систематизація даних з веб-сайтів може здійснюватися різними технологіями, залежно від конкретних вимог та завдань. Загальні технології, які можна використовувати для цих цілей:

1. Веб-скрапінг:

- Beautiful Soup - бібліотека для парсингу HTML та XML даних.

- Scrapy - фреймворк для створення веб-скраперів.

Beautiful Soup - це бібліотека для парсингу HTML і XML даних. Вона робить роботу з різними видами даних, отриманих з веб-сайтів, більш простою та зручною. Beautiful Soup дозволяє ефективно витягувати дані з HTML-розмітки, а також проводити навігацію та знаходження потрібних елементів.

Scrapy - це фреймворк для створення та управління веб-скраперами. Він надає потужні засоби для створення складних і великих скраперів, які можуть збирати дані з багатьох сторінок одночасно та взаємодіяти з різними веб-сайтами.

Ці інструменти часто використовуються в сфері веб-розробки для отримання даних з Інтернету, тестування та аналізу вмісту веб-сайтів.

2. API:

- RESTful API - дозволяє взаємодіяти з веб-серверами за допомогою HTTP-запитів.

- GraphQL - мова запитів та середовище виконання для API.

RESTful API (Representational State Transfer) - це архітектурний стиль, що визначає принципи для побудови веб-сервісів. RESTful API використовує HTTP-протокол для взаємодії з ресурсами (наприклад, даними або послугами) за допомогою стандартних HTTP-методів (GET, POST, PUT, DELETE). Запити та відповіді передаються у форматі JSON або XML.

GraphQL - це мова запитів та середовище виконання для API. Вона дозволяє клієнтам запитувати лише ті дані, які їм потрібні, і отримувати їх у

структурованому форматі. GraphQL використовує один єдиний ендпоінт для всіх запитів, а клієнт визначає структуру отримуваних даних.

Обираючи між RESTful API та GraphQL, розробники враховують потреби свого додатка та специфіку взаємодії з сервером. REST залишається дуже популярним та широко використовується, тоді як GraphQL набуває популярності завдяки своїм можливостям управління даними.

3. Опрацювання JavaScript:

- Selenium - автоматизація веб-браузера для взаємодії з динамічним контентом.

- Puppeteer - набір інструментів для контролю браузера в Node.js середовищі.

Selenium - це набір інструментів для автоматизації веб-браузерів. Він підтримує різні мови програмування (Python, Java, C#, і т.д.) і дозволяє розробникам автоматизовувати дії, які раніше виконувалися вручну. Selenium часто використовується для тестування веб-сайтів, автоматизації скрапінгу або навігації веб-сайтами.

Puppeteer - це набір інструментів для контролю браузера в середовищі Node.js. Він надає інтерфейс для взаємодії з Chromium-браузером або Google Chrome з можливістю виконання різних завдань, таких як генерація скріншотів, створення PDF-документів, автоматизація дій на веб-сайтах.

Якщо потрібно автоматизувати взаємодію з браузерами для тестування, скрапінгу даних або інших завдань, то і Selenium, і Puppeteer можуть бути відмінними виборами в залежності від конкретних вимог та вподобань розробника.

4. Бази даних

- SQL - використання стандартної мови запитів SQL для взаємодії з реляційними базами даних.

- NoSQL - для роботи з нереляційними базами даних, наприклад, MongoDB, Redis, Elasticsearch.

SQL (Structured Query Language) - це стандартна мова запитів для взаємодії з реляційними базами даних. Вона використовується для вставки, оновлення, вилучення та керування даними в таблицях реляційних баз даних. SQL включає в себе такі команди, як SELECT, INSERT, UPDATE, DELETE, а також команди для створення та управління базою даних, таблицями, індексами та іншими об'єктами.

NoSQL - це підхід до роботи з базами даних, який використовує моделі даних, відмінні від традиційної реляційної моделі. NoSQL бази даних можуть бути документ-орієнтованими, ключ-значеннями, колоночними або графовими, і вони спроектовані для зберігання та обробки різноманітних типів даних. Популярні NoSQL бази даних включають MongoDB, Redis, Cassandra, Elasticsearch.

Обираючи між SQL та NoSQL, розробники враховують вимоги їх конкретного додатка, обсяг та характер даних, а також потреби в продуктивності та масштабованості. SQL і NoSQL бази даних часто використовуються у різних сценаріях розробки.

5. Інструменти ETL (Extract, Transform, Load)

- Apache Nifi - потужний інструмент для автоматизації ETL процесів.
- Talend - відкритий інтегрований пакет для ETL та обробки даних.

Apache NiFi - це потужний інструмент для автоматизації ETL процесів та обробки потоків даних. Він надає візуальний інтерфейс для конфігурації та моніторингу потоків даних, дозволяючи легко перетягувати та викидати компоненти для налаштування обробки даних. NiFi підтримує різноманітні джерела даних і призначений для ефективної роботи з великим обсягом даних.

Характеристики:

- Візуальний дизайнер потоків даних.
- Розширені можливості маршрутизації та фільтрації даних.
- Підтримка безпеки та моніторингу.

Talend - це відкритий інтегрований пакет для ETL та обробки даних. Talend надає розгалужену платформу для проектування, розробки, тестування

та управління процесами обробки даних. Забезпечуючи широкий спектр компонентів та підтримуючи різноманітні джерела та призначення даних, Talend дозволяє розробникам легко створювати складні ETL потоки.

Характеристики:

- Графічний інтерфейс для візуального проектування ETL потоків.
- Підтримка різних джерел та систем баз даних.
- Можливості масштабованості та управління версіями.

Обираючи між Apache NiFi та Talend, розробники можуть враховувати конкретні потреби їх проектів, предметну область, вимоги до масштабованості та інші фактори. Обидва інструменти володіють потужними можливостями для автоматизації та оптимізації ETL процесів.

6. Мови програмування

- Python - зручна для веб-скрапінгу, використання різних бібліотек (Beautiful Soup, Requests, Scrapy).
- JavaScript/Node.js - для роботи з Puppeteer, використання пакетів для HTTP-запитів.

Python є відмінним вибором для веб-скрапінгу завдяки своїй простоті та великій кількості бібліотек, спеціально розроблених для цієї задачі. Деякі з найпопулярніших бібліотек для веб-скрапінгу в Python включають Beautiful Soup для парсингу HTML/XML, Requests для виконання HTTP-запитів та Scrapy для створення потужних веб-скраперів.

JavaScript/Node.js

JavaScript, особливо у вигляді Node.js, також є популярним вибором для веб-скрапінгу. У Node.js можна використовувати Puppeteer для автоматизації веб-браузера та взаємодії з динамічним контентом. Крім того, існують пакети для здійснення HTTP-запитів, такі як Axios або вбудований модуль `http`.

Обираючи мову для веб-скрапінгу, розробники враховують особливості своїх проектів, зручність роботи з конкретними бібліотеками та їхнім функціоналом. Як Python, так і JavaScript/Node.js є потужними інструментами

для цієї задачі, і обирання мови залежить від конкретних потреб та вподобань розробника.

7. Фреймворки для обробки даних

- Pandas - відмінний інструмент для обробки та аналізу даних у Python.
- Apache Spark - розподілене обчислення для великих обсягів даних.

Pandas - це бібліотека для обробки та аналізу даних у мові програмування Python. Вона надає структури даних, такі як DataFrame та Series, для ефективного представлення та операцій над даними. Pandas використовується для роботи з невеликими та середніми обсягами даних, де важлива зручність та ефективність в роботі з табличними даними.

Характеристики:

- Робота з табличними даними, подібними до SQL-таблиць.
- Зручний синтаксис для виконання операцій над даними.
- Велика кількість вбудованих функцій для обробки та аналізу даних.

Apache Spark - це розподілене обчислення та обробка великих обсягів даних. Spark надає API для роботи з даними у великих кластерах та дозволяє виконувати розподілені операції над даними. Spark може ефективно обробляти великі обсяги даних, розподілені через кластер комп'ютерів.

Характеристики:

- Розподілене обчислення та обробка даних.
- Підтримка мов програмування Scala, Java, Python та R.
- Висока продуктивність при роботі з великими обсягами даних.

Обираючи між Pandas та Apache Spark, розробники враховують обсяги даних, які вони обробляють, та потреби щодо розподіленого обчислення. Pandas є зручним для роботи з невеликими та середніми обсягами даних на одному комп'ютері, тоді як Apache Spark найбільш ефективний у великих обсягах даних та розподілених обчисленнях на кластерах комп'ютерів.

8. Автоматизація

- Airflow - платформа для програмування, моніторингу та автоматизації роботи процесів збору даних.

Платформа для програмування Apache Airflow дозволяє вам визначати робочі процеси (workflow) у вигляді напівструктурованого коду на мові Python. Ви можете визначати, які завдання повинні виконуватися, та як вони повинні бути пов'язані між собою.

Моніторинг - платформа забезпечує інтерфейс для моніторингу виконання робочих процесів, де ви можете переглядати статуси завдань, логи виконання та іншу інформацію про процеси.

Автоматизація- Airflow дозволяє вам автоматизувати виконання робочих процесів за заданим графіком чи подіями. Ви можете планувати виконання завдань, визначати їх залежності та автоматизувати рутинні операції.

Розширюваність - Airflow має модульну структуру та API для розширення його функціональності. Ви можете використовувати різні плагіни та розширення для підтримки різноманітних джерел даних та інших сервісів.

При використанні Apache Airflow для збору та обробки даних, розробники можуть створювати складні робочі процеси, керувати їх виконанням та автоматизувати задачі, пов'язані із збором та обробкою інформації.

При виборі конкретних технологій важливо враховувати природу веб-сайтів, їх обмеження та правила використання. Також важливо дотримуватися етичних стандартів та правил використання даних.

Web-оглядач і Web-інтерфейс

Web-оглядач чи Web-броузер – це програма, що формує стандартний запит для Web-сервера відповідно до правил протоколу HTTP (Hyper Text Transfer Protocol), одержує документи від Web-сервера, інтерпретує і відображає їх на екрані користувача.

Фактично Web-оглядач є єдиною програмою, яка необхідна користувачу для роботи з інформаційною системою; він надає відповідний Web-інтерфейс.

Web-інтерфейс широко розповсюджений і пропонує готові засоби відображення. Web-оглядачі, що вільно розповсюджуються, мають прості інтерфейси, що збільшує ефективність розгортання програм.

Найпоширеніші Web-оглядачі: MS Internet Explorer, Safari, Mozila FireFox, Opera.

HTML

Найпоширеніший і простий спосіб створення Web-інтерфейсу – використання HTML (HyperText Markup Language, Мова розмітки гіпертексту). Web-інтерфейс складається з набору HTML-документів, статичних чи генерованих динамічно. HTML являє собою набір команд, що називаються тегами, які розташовуються між фрагментами тексту й усередині них. Команди HTML дозволяють структурувати документ, виділяючи в ньому частини тексту (заголовки різних рівнів, абзаци, перерахування і т.д.), які логічно розрізняються. У результаті кожен Web-оглядач може форматувати текст документа таким чином, щоб найкращим способом відобразити його на конкретному дисплеї. Для надання документам більшої виразності текст звичайно форматується з використанням збільшених розмірів шрифту заголовків, застосуванням напівжирного і курсивного накреслень для важливих термінів, виділенням пунктів списків і т.д. Мова HTML дозволяє також включати в документи ілюстративну графіку, що може бути відображена програмами перегляду, що основані на використанні графічного інтерфейсу користувача

Однією з найважливіших властивостей HTML є можливість включення в документ гіпертекстових посилань. Ці посилання дозволяють користувачу завантажити новий документ на свій комп'ютер, просто клацнувши вказівником миші в тім місці екрана, де розташоване посилання. Будь-який документ може містити посилання на інші документи. Областю документа, що використовується як посилання, може служити слово, група слів, графічне зображення чи навіть заданий фрагмент зображення.

Іншою не менш важливою властивістю HTML є форми. HTML-форми дають розробнику Web-інтерфейсу можливість змішування традиційних елементів, властивих друкованим сторінкам, з інтерфейсом користувача сучасної віконної системи. Механізм форм дозволяє розміщати серед форматованого тексту з ілюстраціями кнопки, меню, що розкриваються, текстові поля й інше. Ці елементи інтерфейсу дають користувачам, що переглядають сторінки, можливість передачі відгуку, перетворюючи їх із простих глядачів в учасників, що мають право голосу. Головне обмеження HTML-форм пов'язано з тим, що Web-оглядачі змушені здійснювати нове з'єднання з сервером при кожнім отриманні HTML-сторінки чи передачі даних форми.

HTML сам по собі не дозволяє створювати інтерактивні документи, тобто такий Web-інтерфейс є статичним. Існує ще одна технологія (DHTML), що дозволяє Web-сторінкам бути більш динамічними. DHTML (Dynamic HTML, динамічний HTML) дозволяє змінювати вид сторінки Web і її стиль після того, як документ завантажився.

Ключовим поняттям DHTML є шари. У звичайному HTML-документі, всі елементи так чи інакше впливають один на одного при формуванні його зовнішнього вигляду. Тобто абзац тексту "зміщує" наступні елементи на деяку відстань униз, картинка, вирівняна вправо, "змушує" текст обтікати її і так далі. Всі елементи разом утворять так званий HTML-потік чи потік документа (document flow). Шари у свою чергу – це частини сторінки, що не входять в основний HTML-потік. Для кожного з них створюється свій потік, що ніяк не впливає на основний. При цьому вміст шарів відображається, як правило, над (зверху) основним змістом сторінки. Шари можна пересувати і робити невидимими. Таким чином, за допомогою шарів можна створювати різні інтерактивні ефекти (наприклад, деревоподібне меню як у провіднику Windows).

Одне з обмежень DHTML полягає в тім, що сторінки, створені з застосуванням технології DHTML, можна переглядати тільки в четвертій і більш пізніх версіях Web-оглядачів MS Internet Explorer і Netscape Navigator.

DHTML заснована на технологіях CSS, JavaScript і HTML.

CSS

CSS (Cascading Style Sheets, каскадні таблиці стилів) – ще один стандарт, розроблений W3C. Існують CSS Level 1 і CSS Level 2. Далі мова йде про CSS2.

CSS визначив правила опису стилів візуального відображення елементів HTML-документів. Основні особливості CSS:

- CSS — це мова, що дозволяє приєднувати стилі до будь-яких структурованих документів. На сьогодні такими є HTML-документи і XML-документи.
- CSS поширив поняття стилю відображення на друкувальні пристрої, синтезатори мови й інші пристрої відображення документів.

Ціль створення каскадних таблиць стилів полягала в тому, щоб відокремити структуру документа (описану мовою HTML чи XML) від правил його відображення на різних пристроях (що задаються таблицями CSS). За допомогою стилів можна визначити властивості елементів (текст, гіпертекстове посилання, абзац, заголовок, рисунок, таблиця й інші) у HTML-документі.

Існує три способи завдання стилів у HTML-документі: зовнішні таблиці стилів, внутрішні таблиці стилів і таблиці стилів елементів.

JavaScript

Мова JavaScript – інтерпретуєма об’єктно-базована мова високого рівня, що дозволяє писати програми, які виконуються в середовищі Web-оглядача. В даний час JavaScript є найбільш широко розповсюдженою мовою в Web для написання сценаріїв, що виконуються на стороні клієнта.

JavaScript дозволяє вбудовувати команди в HTML-документи. Команди JavaScript можуть виконуватися при завантаженні документа, при заповненні полів форм, при наведенні вказівника миші на елемент документа й інше. Мова JavaScript має досить широкі можливості. Вона не дозволяє працювати на

рівні машинних кодів, однак дозволяє отримати доступ до багатьох можливостей Web-оглядачів, HTML-документів, а іноді й операційної системи, у якій працює Web-оглядач. На відміну від Java чи C, програми на JavaScript не потрібно компілювати, а оглядачу не потрібно завантажувати віртуальну машину для виконання програмного коду.

Одна з причин, згідно якої Web-розробники використовують JavaScript, – можливість виконання на стороні клієнта багатьох функцій, що раніше виконувалися винятково на стороні сервера. Кращим прикладом є перевірка HTML-форм. За допомогою JavaScript елементи форми можна перевірити до того, як користувач передасть інформацію Web-серверу. Це приводить до зменшення обсягу транзакцій HTTP.

JavaScript широко використовується для створення DHTML-документів. JavaScript дозволяє переміщатися по дереву (ієрархії) елементів HTML-документа, змінювати його властивості і багато чого іншого.

Крім JavaScript існує ще одна мова для вирішення аналогічних задач – VBScript. Однак вона, на відміну від JavaScript, не отримала такого ж широкого поширення.

1.1.2 Технології прикладного рівня

Web-сервер

Web-сервер (HTTP-сервер) – програма, що сприймає запити клієнтів (наприклад, Web-оглядачів), які приходять, як правило, на стандартний порт TCP/IP (порт 80), і виконує в найпростішому випадку пересилання HTML-документів клієнту. Web-сервер є програмою, яка дозволяє службі Internet, що називається WWW (World Wide Web, Всесвітня інформаційна мережа), знаходити і використовувати гіпертекстові і графічні документи на різних серверах.

Фактично Web-сервер передає інформацію користувачу, який знаходиться на віддаленому комп'ютері. Користувач переглядає інформацію за допомогою Web-оглядача (MS Internet Explorer, Netscape Navigator і інші) і відсилає нові запити Web-серверу.

Web-сервер як правило має вбудовану систему аутентифікації по логіну та паролю, яку можна використовувати для обмеження доступу до ресурсів сервера.

Найрозповсюджені Web-сервера: Apache Server, MS IIS, Netscape Enterprise Server.

CGI

CGI (Common Gateway Interface) – це набір правил, згідно з якими програми на стороні сервера можуть через Web-сервер пересилати дані клієнтам. CGI є перевіреним і універсальним способом створення розподілених систем у Internet. Головна задача, яку він вирішує, – це забезпечення викликів віддалених процедур між клієнтом і сервером з використанням сокетів TCP/IP.

Крім CGI існують ще кілька подібних технологій – ISAPI/NSAPI, Servlet і JSP. Ці чотири технології призначені для вирішення приблизно тих самих задач і концептуально дуже схожі. Всі вони дозволяють розширити можливості Web-серверів шляхом розробки програм, зареєстрованих на цьому сервері.

Servlet і JSP

Технологія Servlet була розроблена компанією Sun Microsystems, і булла призначена для того щоб серверна частина отримуючи запит від клієнта у якості відповіді повернула сгенеровану HTML сторінку яка і відображає дані. Servlet – це спеціальний Java класс, який реалізує необхідні API для отримання запросів від WEB клієнта та після обробки надання результату. Сам по собі Servlet не чого робити не вміє він є частиною системи яка знаходиться у контернері(Java web сервер) який вміє отримати запит і передати його необхідному приложеню. Технологія JSP (JSPx) є „надстройкою” над сервлетами і призначина для більш наглядного конструювання інтерфейсу. JSP є наслідником Servlet і являє собою комбінацію Java коду та HTML, на зразок PHP.

Java web сервери

Існує декілька Java web серверів зокрема такі, JBoss, Tomcat, Glassfish та ін. Майже усі вони виконують одні і тіж самі задачі:

- компіляція JSP сторінок;
- звязок з базою даних
- обмін даними між програмою та клієнтом;
- обмін даними між програмою та БД;

1.1.3 Технології рівня даних

Мова SQL. SQL – це скорочена назва структурованої мови запитів (Structured Query Language). SQL – це мова програмування, яка використовується для виконання запитів і модифікації даних, а також для управління СУБД (системами управління базами даних). SQL працює тільки з базами даних одного певного типу, які називаються реляційними.

Стандарти SQL затверджені ANSI і ISO. Хоча в кожній СУБД використовується власний діалект SQL, більшість цих діалектів задовольняють стандарту ANSI SQL. Таким чином, для сумісності програмного забезпечення з різними СУБД, необхідно строго дотримуватися стандарту ANSI SQL.

Мова SQL містить оператори, які підрозділяються на дві категорії: мова опису даних (data definition language, DDL) і мова маніпулювання даними (data manipulation language, DML). DDL використовується для створення і управління такими об'єктами, як бази даних, таблиці і представлення. Оператори DDL містять команди CREATE, ALTER і DROP для кожного з об'єктів. DML використовується для управління даними в об'єктах баз даних за допомогою таких операторів, як SELECT, INSERT, UPDATE і DELETE.

Сервер баз даних

Сервер баз даних – це окрема програма, яка виконуванується як окремий процес (сервіс). Сервер передає обрану з бази інформацію по каналу клієнту. Саме сервер працює з даними, піклується про їхнє розміщення на диску.

Найрозповсюджені сервера баз даних: MS SQL Server, Oracle, Sybase, MySQL, PostgreSQL.

1.1.4 Динаміка роботи інформаційної системи

Маючи загальне уявлення про технології, які використовуються для розробки системи, можна описати динаміку роботи системи:

- Користувач запускає Web-оглядач на віддаленому комп'ютері-клієнті і вводить адресу Web-сервера інформаційної системи.
- Web-оглядач надсилає запит Web-серверу.
- Web-серверу надсилає запит Java-серверу.
- Java-сервер передає запит відповідному приложенню(системі).
- Система отримав запит і передає його відповідному сервлету(JSP).
- Servlet(JSP) обробив запит звертається (якщо необхідно) до SessionBean
- SessionBean звертається до Java-сервера з проханням надати звязок з необхідним сервером БД та проводить аутентифікацію користувача по логіну і паролю та вибирає потрібну базу.
- Отримав звязок SessionBean проводиь необхідний запит і отримав результати повертає їх контролеру (Servlet або JSP)
- Контролер отримав результат генерює необхідну HTML сторінку та повертає її і передає її Web-серверу.
- Web-сервер додає HTTP-заголовок до отриманого документа і відправляє його Web-оглядачу.
- Web-оглядач приймає HTML -документ і відображає його на екрані користувача.
- Користувач отримує Web-інтерфейс і необхідну інформацію. Він може надіслати новий запит Web-серверу.

Таким чином, вимальовується схема взаємодії технологій (рисунок 1.2)

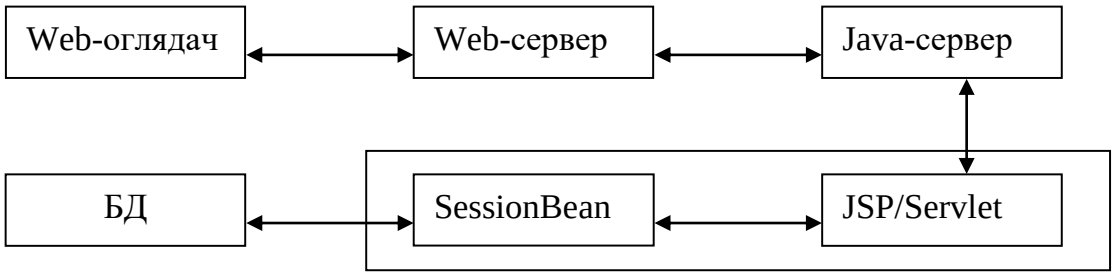

Рисунок 1.2 – Схема взаємодії технологій інформаційної системи

1.2 Формування вимог до інформаційної системи

1.2.1 Характеристика об'єкта автоматизації

Об’єктом автоматизації є ПП „ТОВ”, пов’язане з розробкою програмного забезпечення. Підприємство має потребу у створенні єдиної бази даних окремих веб-магазинів та інформацію про існуючі продукти в них.

1.2.2 Цілі, критерії й обмеження створення інформаційної системи

Метою розробки є підвищення ефективності роботи інформаційної системи, збільшення продуктивності шляхом автоматизації функцій, які вона виконує.

Основні критерії створення інформаційної системи:

- Соціальний критерій. Інформаційна система розрахована на користувачів з різною комп'ютерною підготовкою. Необхідний зручний і інтуїтивно зрозумілий інтерфейс для роботи з нею.

- Економічний критерій. Автоматизована інформаційна система створюється для підприємства безкоштовно. Вартість інформаційної системи буде складатися з витрат на придбання необхідного програмного, технічного й іншого видів забезпечення.

- Критерій мобільності. Користувач повинен мати можливість працювати із системою як з комп'ютера локальної мережі, так і з будь-якого комп'ютера, що має доступ до Internet.

- Критерій безпеки. Система повинна бути безпечною з погляду збереження й обробки даних. Втрата даних може привести до серйозних збоїв у роботі системи. Необхідний захист системи від несанкціонованого втручання.

1.2.3 Функції та задачі інформаційної системи

Задачею інформаційної системи, що створюється, є автоматизація наступних функцій програми:

- Облік інформації необхідної для заповнення БД
- Облік завданнь та інформацію про їхній статус.
- Формування звітів.
- Усі інші вимоги не є головними і можуть змінюватися під час розробки або експлуатації

Для створення інформаційної системи необхідно розробити єдину базу даних для збереження інформації про існуючі товари у веб-магазинах. Також повинно бути передбачене програмне забезпечення для адміністрування системи.

1.2.4 Висновки і пропозиції

Таким чином, необхідно розробити автоматизовану інформаційну систему, що дозволить підвищити ефективність та якість роботи програми. Інформаційна система повинна включати базу даних, програмне забезпечення адміністратора, а також відповідне організаційне, технічне та інші види забезпечення.

У перспективі, база даних інформаційної системи може використовуватися різними підрозділами підприємства. Начальник відділу або менеджер може одержувати коректну і своєчасну інформацію про стан кожного завдання та інформацію про співробітників.

1.3 Розробка концепції інформаційної системи

1.3.1 Опис результатів вивчення об'єкта автоматизації

Для створення системи потрібна розробка бази даних інтерфейсу користувача та адміністратора.

1.3.2 Аналіз вимог до інформаційної системи

Крім функціональних вимог до інформаційної системи (корректне вилучення інформації), необхідно врахувати вимоги до системи в цілому:

- Простий і зручний інтерфейс користувача.
- Доступ до системи через Internet з будь-якого комп'ютера глобальної мережі.
- Висока масштабуємість системи.
- Висока продуктивність.
- Надійність.
- Конфіденційність інформації.

Аналізуючи зазначені вище вимоги, можна зробити висновок, що інформаційна система повинна мати/підтримувати:

- Web-інтерфейс.
- Сучасні стандарти в області інформаційних технологій (протоколи передачі даних, стандарти збереження даних і т.д.).
- Сучасні серверні платформи і програмне забезпечення.
- Сучасні рішення в області інформаційних технологій по розробці розподілених систем.
- Ефективні методи доступу до даних.
- Забезпечити простоту модернізації, можливість повторного використання програмного забезпечення (коду).

1.3.3 Вибір логічної архітектури програмного забезпечення інформаційної системи

Логічна архітектура програмного забезпечення інформаційної системи являє собою концептуальний опис структури програмного забезпечення системи. Кожна система має три рівні: рівень користувача, прикладний і рівень даних. Відрізняються ж архітектури тільки організацією програмного забезпечення (програмного коду), індивідуальної для конкретної системи.

Багато сучасних інформаційних систем побудовані на базі дворівневої архітектури «клієнт/сервер». При цьому клієнтське програмне забезпечення

відповідає за обробку і відображення даних. Самі ж дані централізовано зберігаються на серверах, до яких підключаються клієнти. Часто час використання системи обмежується тривалістю такого з'єднання.

Клієнт-серверні системи добре працюють тільки в контрольованих середовищах, коли число користувачів передбачуване (це необхідно для виділення ресурсів). Однак ця архітектура стає неефективною, при великому числі користувачів або якщо їхнє число невідомо. Крім цього, масштабуємість таких систем невисока, тому що кількість безпосередніх з'єднань із сервером обмежена. Невеликі і можливості повторного застосування, оскільки користувачі прив'язані до визначеного формату бази даних. А тому що клієнтське програмне забезпечення містить блок обробки інформації, загальний обсяг програмного забезпечення досить великий (Такий тип клієнтських програм іноді називають «товстим» клієнтом). У цьому випадку при зміні алгоритмів обробки даних приходиться встановлювати нове програмне забезпечення на кожен клієнтський комп'ютер.

Можна отримати невелике поліпшення шляхом переміщення прикладних алгоритмів і блоків обробки даних на сервери даних (наприклад, за допомогою збережених процедур сервера баз даних). Таку архітектуру іноді називають 2,5-рівневою. Масштабуємість подібних систем трохи краща, але все рівно мала для виконання вимог інформаційної системи. Крім того, можливість повторного використання програмного забезпечення (коду) залишається на невисокому рівні.

Масштабуємість і ступінь повторного використання можна помітно поліпшити, додавши в архітектуру системи третій рівень. У такій багаторівневій архітектурі (її також називають N-рівневою) усі рівні – рівень користувача, прикладний і рівень даних – логічно розділені (рисунок 1.3). Багаторівнева архітектура задовольняє вимогам до інформаційної системи, що розробляється.

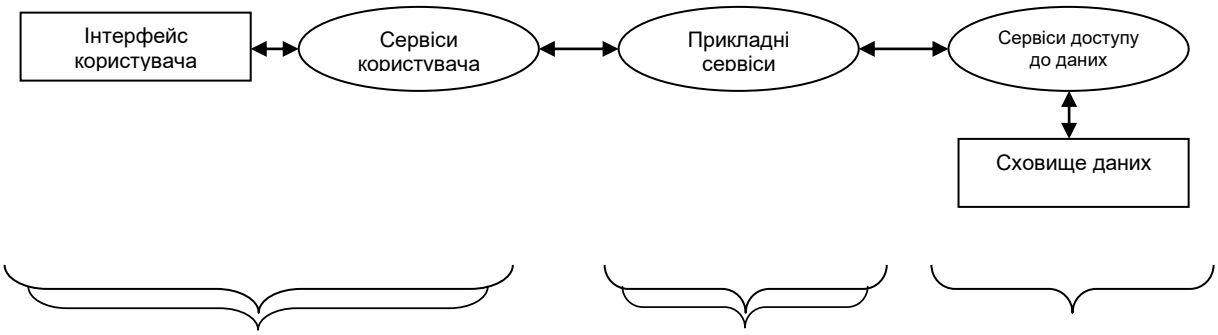

Рисунок 1.3 - Логічна архітектура програмного забезпечення інформаційної системи

Опишемо функції кожного рівня:

- Рівень користувача: відображає дані і дозволяє користувачу редагувати їх. Існує два основних типи інтерфейсу: віконний (реалізується засобами підсистеми інтерфейсу користувача операційної системи) і на основі Web. Web-інтерфейси засновані на HTML, DHTML, XHTML чи XML, у результаті чого вони можуть відображатися будь-яким Web-оглядачем на будь-якій платформі.

- Прикладний рівень: тут реалізовані правила й обмеження на дані (перевірка коректності даних). І хоча його сервіси використовуються презентаційним рівнем, він не прив'язаний до якого-небудь клієнта – сервіси прикладного рівня доступні будь-якому клієнту. Бізнес-правила виражаються у формі прикладних алгоритмів, корпоративних правил і т.д. Обмеження на дані гарантують точність і цілісність інформації, що зберігається. Бізнес-правила звичайно реалізуються окремим модулем на централізованому сервері, що дає можливість доступу до нього відразу декільком клієнтам.

- Рівень даних: прикладний рівень не має інформації про те, як і де зберігаються дані, які він обробляє. У цьому питанні він покладається на сервіси доступу до даних, що виконують всю роботу з отримання і передачі даних. Сервіси доступу до даних також реалізуються у виді ізольованих модулів, які знають про місце збереження інформації. Кожен модуль доступу

до даних, як правило, відповідає і за цілісність сховища (наприклад, реляційної бази даних). Для багаторівневих програм у якості сховища інформації підходять сервера баз даних (MS SQL Server, Oracle, Sybase, MySQL, PostgreSQL і ін.), необхідні для обслуговування даних у таблицях і високопродуктивної виборки інформації.

1.3.4 Вибір фізичної архітектури програмного забезпечення інформаційної системи

Виходячи з логічної архітектури програмного забезпечення, можна розробити його фізичну архітектуру. Фізична архітектура являє собою опис структури програмного забезпечення системи з погляду наявних у системі фізичних компонентів: серверів і клієнтських комп'ютерів.

Відповідно до описаної логічної архітектури, можна одержати дві фізичні архітектури, які так чи інакше задовольняють вимогам до системи. Перша фізична архітектура є триланковою (рисунок 1.4), друга – чотириланковою (рисунок 1.5).

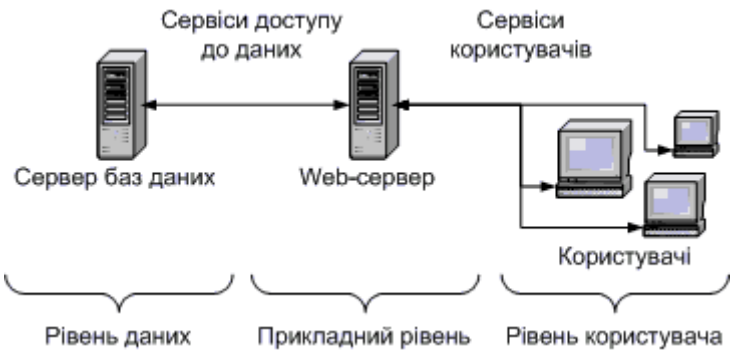

Рисунок 1.4 – Триланкова фізична архітектура програмного забезпечення інформаційної системи

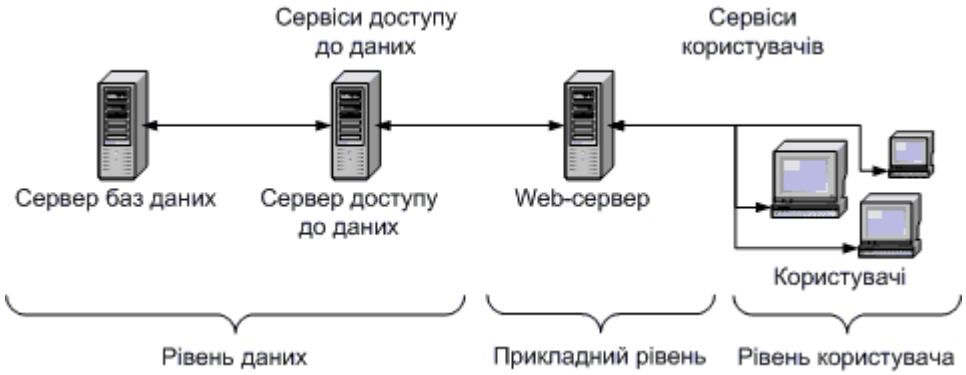

Рисунок 1.5 – Чотириланкова фізична архітектура програмного забезпечення інформаційної системи

Порівняємо отримані фізичні архітектури для інформаційної системи.

Переваги триланкової архітектури над чотириланковою:

- Краща продуктивність при невеликому числі користувачів.
- Порівняльна простота адміністрування системи.

Переваги чотириланкової архітектури над триланковою:

- Краща продуктивність при великому числі користувачів.
- Краща масштабуємість.
- Краща надійність.
- Кращі можливості по модернізації і повторному використанню.

Але чотириланкова архітектура не є більш перспективною для даного проекту, так як паралельних сеансів доступу до БД не буде. Вона в більшій мері задовольняє вимогам до інформаційної системи.

1.3.5 Вибір технологій для розробки інформаційної системи

Відповідно до обраної фізичної архітектури для інформаційної системи «ТОВ», система є розподіленою і має три ланки: сервер баз даних, Web-сервер і клієнт.

Існують наступні технології для створення розподілених систем:

- DCOM/COM+, CORBA (об’єктно-орієнтовані технології).
- CGI, ISAPI/NSAPI, Servlets і інші.

Всі ці технології тим чи іншим способом організують виклик віддалених процедур (методів) чи його аналог.

Взаємодію між клієнтами і Web-сервером раціонально організувати за допомогою CGI, ISAPI/NSAPI і/або Servlets, тому що ці технології орієнтовані на Web-інтерфейс. Використання DCOM/COM+, CORBA чи RMI у цьому випадку серйозно збільшило б вимоги до програмного забезпечення клієнтських машин, що суперечить вимогам до інформаційної системи (доступ до системи через Internet з будь-якого комп'ютера глобальної мережі). Усі зазначені технології (CGI, ISAPI/NSAPI, Servlets) вирішують однакові задачі і мають приблизно однакові характеристики по продуктивності. Servlets є най гібкішою та одною з най сучасніших технологій , а також реалізується на мові високого рівня Java яка не залежиь від операційної системи тому і обераємо цю технологію.

Для взаємодії серверів та доступа до БД потрібно застосовувати більш розвинуті технології.

Найбільш перспективною й універсальною технологією є CORBA, але вона теж має недоліки. CORBA – технологія досить молода і перебуває в стадії доведень і ліквідації «дитячих хвороб». Вона «сира» просто в силу свого надзвичайно швидкого розвитку. Не встигають розробники виправити помилки в поточній версії, потрібно створювати нову внаслідок появи нових можливостей. Також, гідним образом реалізовані не всі стандартні сервіси CORBA. Навіть найкраща специфікація не врятує положення при відсутності її доступної реалізації. Крім того, за CORBA (на відміну від того ж COM) треба платити.

Основний недолік DCOM/COM+ пов'язаний з необхідністю використовувати Windows-платформу. DCOM/COM+ на практиці довела свою придатність для розробки інформаційних систем. Це безкоштовна технологія і на ній засновано багато інших технологій (наприклад, OLE DB – найефективніший/найпродуктивний доступ до будь-яких даних). Ця технологія цілком задовольняє вимогам до інформаційної системи, перевірена на практиці,

є безкоштовною, тому раціонально використовувати її для взаємодії серверів. Так як ми обрали Servlets то логічно обрати технологію яка реалізується на Java тому наш вибір це EJB, крім цього EJB має багато переваг серед своїх конкурентів.

Як говорилося раніше, на рівні користувача необхідно використовувати Web-інтерфейс (HTML/DHTML, CSS, JavaScript, DOM, Java-аплети). Клієнтське програмне забезпечення – Web-оглядач.

1.3.6 Розробка варіантів концепції інформаційної системи

Виходячи з аналізу вимог до інформаційної системи, її функціональних характеристик, фізичної архітектури і технологій для розробки, можна розробити концепцію інформаційної системи. Основними компонентами системи є:

- серверна платформа і серверне програмне забезпечення;
- клієнтська платформа і клієнтське програмне забезпечення;
- засоби розробки;
- мережне середовище;
- технічні засоби.

Від вибору перерахованих компонентів залежить ефективність інформаційної системи. Розглянемо кожен компонент інформаційної системи.

Серверна платформа і серверне програмне забезпечення

На Web-сервері повинен бути встановлений HTTP-сервер. Найрозповсюдженими HTTP-сервера, що задовольняючі зазначеним вимогам і мають високу продуктивність, є Apache Server. На Web-сервері повинна бути встановлена ОС яка підтримує цей сервер.

Так як використовуються технології організовані на Java то операційна система на сервері доступу до БД може бути будь яка з перелічиних:

- MS Windows
- Linux
- Unix

– Mac OS

Також потрібний Java WEB сервер встановлений на сервері будь який з:

- JBoss
- Sun Apllication Server
- GlassFish

На сервері баз даних повинна бути встановлена серверна система управління базами даних (СУБД).

Система управління базами даних

Необхідна сучасна серверна реляційна СУБД, що має високу продуктивність, здатна працювати з великими обсягами інформації і підтримує стандарт SQL-92. Крім того, СУБД повинна працювати під управлінням Windows. Серверними СУБД, що задовольняють зазначеним вимогам, на сьогодні є:

- MS SQL Server.
- MySQL.

Клієнтська платформа і клієнтське програмне забезпечення

На клієнтських машинах може використовуватися будь-яка операційна система з можливістю встановлення віддаленого мережного з'єднання і графічним інтерфейсом. Операційні системи, що рекомендуються – MS Windows, Linux, тощо.

Програмне забезпечення, з яким буде працювати користувач, являє собою Web-оглядач з підтримкою HTML 4.0, JavaScript 1.0, CSS 2 (наприклад, MS Internet Explorer, FireFox, Opera ...) а також з підтримкою створення “правил” для визначення потрібної інформації.

Мережне середовище

Підприємство має локальну комп'ютерну мережу Ethernet, яку можна використовувати як мережне середовище інформаційної системи. Можливо, потрібно буде внести зміни в конфігурацію існуючої комп'ютерної мережі. Основна вимога до мережі – швидкість передачі даних не менше 10 Мбіт/с.

Технічні засоби

Технічні засоби серверів повинні задовольняти вимогам операційної системи і СУБД до обладнання. Конфігурація кожного сервера може бути наступною:

- Мікропроцесор класу Intel Core 2 Duo 2600 МГц і вище.
- Оперативна пам'ять RAM 4 Гб і більше.
- Жорсткий диск з 150 Гб вільного місця і більше
- Мережна карта Ethernet (не менше 10 Мбіт/с).

Технічні засоби клієнтських машин повинні відповідати вимогам операційної системи, що використовується. Конфігурація комп'ютера користувача може бути наступною:

- Мікропроцесор класу Intel Dual Core 2500 МГц і вище.
- Оперативна пам'ять RAM 2 Гб і більше.
- Жорсткий диск з 20 Гб вільного місця і більше.
- Мережна карта Ethernet.

Засоби розробки

У якості засоба розробки є обєктно орінтована мова програмування Java 1.5 це обгрунтовується по-перше незалежністю від платформи а по-друге безкоштовністю.

Варіанти концепції інформаційної системи

Таким чином, розглянувши компоненти системи і вибравши найбільш оптимальні, можна визначити кілька варіантів концепції інформаційної системи «WoW» (див. таблицю 1.1).

Таблиця 1.1 – Варіанти концепції інформаційної системи

№	ПЗ Web-сервера (ОС, HTTP-сервер)	ПЗ сервера доступу до даних (ОС)	ПЗ сервера баз даних (ОС, СУБД)
1	MS Windows 2019 Server, MS IIS 5.1	MS Windows 2019 Server, GlassFish 3	MS Windows 2019 Server,MS SQL Server 2014
2	Linux OpenSUSE 15.5, Apache Server	MS Windows 2019 Server, GlassFish	MS Windows 2019 Server, MS SQL Server
3	MS Windows 2019 Server, MS IIS 10.0	Linux, Jboss	MS Windows 2019 Server, MySQL
4	Linux CentOS, Apache Server	Linux, Jboss	Linux, MySQL EnterpriseEdition

Крім того, кожен варіант концепції системи має відповідне технічне (описане вище) і організаційне забезпечення. Причому, і технічне, і організаційне забезпечення будуть однаковими для усіх варіантів, оскільки обране інформаційне забезпечення має приблизно однакові вимоги до апаратних ресурсів і обслуговування.

При виборі варіантів концепції інформаційної системи, були враховані наступні особливості:

- На Web-сервері і сервері доступу до даних повинна працювати операційна система з повною підтримкою Java, тобто Windows, Linux, Macintosh, так як з фінансової точки зору Linux є безкоштовною ОС то цей вибір є найкращим. Сервер баз даних може працювати під управлінням як Windows, так і UNIX. Якщо сервер баз даних не буде виділеним (наприклад, сполучений із сервером доступу до даних), то необхідне використання Linux.
- Краще на всіх серверах встановлювати однакову операційну систему, тому що їх легше буде підтримувати (складно буде знайти фахівця, який буде добре знати кілька серверних операційних систем).
- СУБД MySQL ідеально підходить так як плотформо незалежний до того ж безкоштовний.

- Використання ОС Windows нераціонально, оскільки її могутні можливості не будуть використані повною мірою, а витрати на її придбання досить великі.

1.3.7 Вимоги, що впливають на якість інформаційної системи

Якість інформаційної системи і якість програмного забезпечення не піддаються точному визначенню і виміру. На даний момент, не існує ні метрології якості інформаційних систем, ні реальних методів оцінки якості програмного забезпечення, ні методів оцінки якості процесу проектування, розробки і впровадження.

Проте, задача поставлена і вимагає вирішення. Складемо список вимог (критеріїв), що можуть впливати на якість інформаційної системи (частина з них взята зі стандарту ISO-9126):

- Здатність підвищити ефективність працівників.
- Функціональність.
  - Відповідність призначенню.
  - Точність.
  - Повнота інформації.
  - Здатність взаємодіяти із середовищем.
  - Відповідність нормам.
  - Безпека (захист від взлому даних і інших злочинних дій).
- Надійність.
  - Зрілість ("обкатаність").
  - Стійкість до відмов.
  - Стійкість до помилок користувача.
  - Здатність відновлюватися після збоїв.
- Масштабуємість.
- Придатність до використання.
  - Зрозумілість.
  - Вивчаємість.
  - Зручність і простота в роботі.

– Ефективність.

- Продуктивність і час відгуку.
- Споживання ресурсів.

– Супроводжуємість.

- Аналізуємість (діагностика причин помилок і зіставлення з первісним кодом).
- Придатність до змін.
- Стабільність.
- Тестуємість.

Потрібно відзначити, що виконання всіх зазначених вимог ще не гарантує якість інформаційної системи. Також, невідповідність системи якій-небудь вимозі не говорить про те, що система неякісна. Як уже говорилося вище, не існує формальних методів оцінки якості інформаційної системи.

1.4 Техніко-економічне обґрунтування розробки інформаційної системи на основі функціонально-вартісного аналізу

1.4.1 Постановка задачі для функціонально-вартісного аналізу

Для вибору найбільш ефективного варіанта концепції інформаційної системи управління завданнями можна використати метод функціонально-вартісного аналізу.

В основі методу лежить функціональний підхід, згідно з яким об'єктом аналізу є не сам виріб (система, програмне забезпечення), а функції, що він виконує.

Функціонально-вартісний аналіз складається з двох основних етапів:

- функціональний аналіз;
- вартісний аналіз.

Обрані варіантами концепції системи представлені в таблиці 1.2.

Таблиця 1.2 Концепція системи

№	ПЗ Web-сервера (ОС, HTTP-сервер)	ПЗ сервера доступу до даних (ОС)	ПЗ сервера баз даних (ОС, СУБД)
1	MS Windows 2019 Server, MS IIS 5.1	MS Windows 2019 Server, GlassFish 3	MS Windows 2019 Server,MS SQL Server 2014
2	Linux OpenSUSE 15.5, Apache Server	MS Windows 2019 Server, GlassFish	MS Windows 2019 Server, MS SQL Server
3	MS Windows 2019 Server, MS IIS 10.0	Linux, Jboss	MS Windows 2019 Server, MySQL
4	Linux CentOS, Apache Server	Linux, Jboss	Linux, MySQL EnterpriseEdition

Конфігурація кожного сервера може бути наступною:

- Мікропроцесор класу Intel Core 5 Duo 2600 МГц і вище.
- Оперативна пам'ять RAM 4 Гб і більше.
- Жорсткий диск з 150 Гб вільного місця і більше
- Мережна карта Ethernet (не менше 10 Мбіт/с).

Конфігурація комп'ютера користувача може бути наступною:

- Мікропроцесор класу Intel Dual Core 2500 МГц і вище.
- Оперативна пам'ять RAM 2 Гб і більше.
- Жорсткий диск з 20 Гб вільного місця і більше.
- Мережна карта Ethernet.

Як мережне середовище можна використовувати кабелі типу «вита пара» категорії 3, 5 і вище. Активне мережне обладнання повинне підтримувати як мінімум швидкість 10 Мбіт/сек.

Організаційне забезпечення передбачає обслуговування системи протягом строку окупності. Розрахунок строку окупності приведений в організаційно-економічному розділі даної роботи.

1.4.2 Функціональний аналіз

Якщо оцінювати варіанти реалізації функцій з погляду надання необхідної функціональності (для реалізації системи), то всі вони надають необхідну функціональність вовною мірою. Таким чином, говорити про недоліки чи переваги окремих варіантів зараз передчасно. Система, реалізована по кожному з цих варіантів, буде працювати успішно. Для того, щоб оцінити різні варіанти необхідно визначити основні параметри системи. Основними параметрами системи відповідно до вимог замовника є:

- Функціональність (відповідність призначенню, точність, повнота інформації, безпека).
- Ефективність (продуктивність і час відгуку, споживання ресурсів).
- Надійність (стійкість до відмов, стійкість до помилок користувача, здатність відновлюватися після збоїв).
- Масштабуємість.
- Придатність до використання (зрозумілість, вивчаємість, зручність і простота в роботі).
- Супроводжуємість (аналізуємість (діагностика причин помилок і зіставлення з первісним кодом), придатність до змін, стабільність, тестуємість).

Жоден з цих параметрів не має одиниці виміру; не існує також стандартної методики визначення їхніх чисельних значень. Таким чином, необхідно одержати експертні оцінки по кожнім варіанті системи і відповідним параметрам. Експертами в даному випадку можуть бути люди які є фахівцями в області інформаційних технологій, фахівцями з інформаційної системи, автоматизації функцій якої присвячена дана робота, і потенційними користувачами інформаційної системи, що розробляється. Усереднені експертні оцінки (10 бальна шкала) по кожному варіанту системи і відповідним параметрам представлені в таблиці 1.3.

Таблиця 1.3 – Основні параметри системи

Параметр	Оцінка в балах по варіанту №, B_i			
	1	2	3	4
Функціональність	10	10	10	10
Ефективність	7	7	8	9
Надійність	8	8	8	8
Масштабуємість	9	9	9	10
Супроводжуємість	8	8	6	7

На наступному етапі функціонального аналізу потрібно визначити вагові коефіцієнти для кожного з параметрів. У даному випадку вони також визначаються експертним шляхом. Значення вагових коефіцієнтів для кожного з параметрів приведені в таблиці 1.4. Насправді, сума усіх вагових коефіцієнтів повинна дорівнювати одиниці, але для наочності проведемо розрахунок з ненормалізованими коефіцієнтами, а потім розділимо отриманий результат (коефіцієнт технічного рівня) на суму вагових коефіцієнтів (5.0).

Таблиця 1.4 – Вагові коефіцієнти для параметрів системи

Параметр	Коефіцієнт, φ_i
Функціональність	0.3
Ефективність	0.2
Надійність	0.3
Масштабуємість	0.1
Супроводжуємість	0.1

Після визначення вагових коефіцієнтів φ_i і бальних оцінок B_i для всіх параметрів системи, необхідно визначити коефіцієнти технічного рівня для кожного з параметрів по кожнім варіанту по формулі

Результати розрахунку приведені в таблиці 1.5.

За даними таблиці 1.5 розраховується коефіцієнт технічного рівня K_{TP} для кожного з варіантів по формулі

$K_{TP}=\sum_{i=1}^{N} K_{TP_i}$, де K_{TPi} – показник технічного рівня для і-го параметра.

Також для нормалізації значення коефіцієнта технічного рівня, отримане по попередній формулі значення потрібно розділити на суму вагових коефіцієнтів, що дорівнює 5.0. Результати розрахунку приведені в таблиці 1.6.

Параметр	Варіант	Оцінка параметра в балах, B_i	Ваговий коефіцієнт, φ_i	Показник технічного рівня, K_{TP}
Функціональність	1	10	0.3	3
	2	10		3
	3	10		3
	4	10		3
Ефективність	1	7	0.2	1.4
	2	7		1.4
	3	8		1.6
	4	9		1.9
Надійність	1	8	0.3	2.4
	2	8		2.4
	3	8		2.4
	4	8		2.4
Масштабуємість	1	9	0.1	0.9
	2	9		0.9
	3	9		0.9
	4	10		1
Супроводжуємість	1	8	0.1	0.8
	2	8		0.8
	3	6		0.6
	4	7		0.7

Таблиця 1.6 – Коефіцієнти технічного рівня для варіантів концепції системи

▪ Варіант	Коефіцієнт технічного рівня, K_{TP}
▪ 1	7.5
▪ 2	7.5
▪ 3	8.5
▪ 4	9.0

Найкращим на етапі функціонального аналізу є варіант, який має максимальне значення коефіцієнта технічного рівня $К_{ТР}$, тобто варіант 4.

1.4.3 Вартісний аналіз

Аналіз функцій, що реалізуються системою, повинен бути доповнений вартісним аналізом. Для цього розглядаються зведені витрати $З_j$ по кожному j-му варіанту: $З_j = С_j + Е_н К_j$, де $С_j$ – поточні витрати по j-му варіанту; $К_j$ – капітальні вкладення по j-му варіанту; $Е_н$ – нормативний коефіцієнт економічної ефективності капітальних вкладень, що дорівнює 0,15.

Зведені витрати $З_j$ розраховують у сфері виробництва (у розробника) чи в сфері експлуатації (у користувача). У нашому випадку доцільно розраховувати витрати у сфері виробництва.

Поточні витрати при розробці інформаційної системи $С_j$ – це собівартість створення системи $С_с$. Для усіх варіантів будемо вважати, що собівартість створення системи $С_с$ однакова. Собівартість системи розрахована в організаційно-економічному розділі даної роботи і складає 6276 грн.

Капітальні вкладення $К_j$ – це витрати, що носять одноразовий характер. У даному випадку внесемо сюди вартість покупного ліцензійного програмного забезпечення, технічного й організаційного забезпечення, необхідного для створення системи.

Вартість необхідного програмного забезпечення на сьогоднішній день приведена в таблиці 1.7. Для наочності в таблиці 1.7 приведені не тільки ціни на програмне забезпечення, необхідне для впровадження системи в експлуатацію, але і ціни на засоби розробки, витрати на які входять у собівартість системи.

Оскільки необхідне технічне забезпечення для кожного варіанта вже є на підприємстві, то витрати на технічне забезпечення дорівнюють нулю.

Вартість організаційного забезпечення розрахована в організаційно-економічному розділі даної роботи. Вона включає витрати на налагодження локальної обчислювальної мережі і підготовку кадрів, і складає 150 грн. для кожного варіанта. Такі порівняно маленькі витрати обумовлені тим, що

впровадження інформаційної системи в експлуатацію не приведе до змін в організаційній структурі.

Таблиця 1.7 Вартість необхідного програмного забезпечення

Варіант	Найменування ПЗ	Вартість, тис. грн.	Кіль-кість	Разом, тис. грн.
1	MS Windows 2019 Server English Int	20.997	3	62991
	MS IIS 10.0	0	1	0
	InterBase	18.959	1	18.959
	GlassFish	0	1	0
2	MS Windows 2019 Server English Int	20.997	3	62991
	Apache	0	1	0
	MS SQL Server 2022 Standard Edition English Academic Edition OLP NL	12.316	1	12.316
	JBoss	0	1	0
3	SUSE Linux 20.3	0	3	0
	MySQL 10 EnterpriceEdition	0	1	0
	GlassFish	0	1	0
4	CentOS Linux	0	3	0
	MySQL 10 EnterpriceEdition	0	1	0
	JBoss	0	1	0

Капітальні вкладення по кожному варіанту, згідно до сучасного стану ринку програмного забезпечення, приведені в таблиці 1.8.

Таблиця 1.8 – Капітальні вкладення по варіантах

▪ Варіант	Вартість, тис. грн
▪ 1	170950
▪ 2	110307
▪ 3	0
▪ 4	0

Таким чином, можна розрахувати зведені витрати по кожному варіанту. Результати розрахунку приведені в таблиці 1.9.

Таблиця 1.9 – Зведені витрати по варіантах

Варіант	Витрати, тис. грн
▪ 1	89680
▪ 2	79720
▪ 3	60276
▪ 4	60276

1.4.4 Розрахунок узагальненого показника ефективності варіантів концепції системи

За результатами функціонального і вартісного аналізу можна розрахувати коефіцієнт техніко-економічного рівня по кожному з варіантів по формулі:

$$K_{TEP_j} = \frac{K_{TP_j}}{З_j} \quad .$$

Результати розрахунку приведені в таблиці 1.10.

Варіант	Коефіцієнт, K_{TEPj}
▪ 1	0.00084
▪ 2	0.00094
▪ 3	0.00137
▪ **4**	**0.00145**

Найкращий варіант визначається за максимальним значенням коефіцієнта K_{TEPj}. Таким чином, визначаємо, що найкращий варіант – це варіант 4:

CentOS Linux

Apache

MySQL 10 EnterpriceEdition

JBoss

1.4.5 Очікувані результати й ефективність реалізації обраного варіанта концепції системи

Тому що обраний варіант реалізації задовольняє всім основним вимогам до системи, очікуються ефективні і високі результати при рішенні поставлених задач:

– створення єдиної комп’ютерної мережі підприємства;

– забезпечення надійного контролю цілісності даних;

– підвищення ефективності виконання програм за рахунок використання потужності сервера з застосуванням технології “клієнт-сервер”;

– використання існуючого обладнання у якості клієнтських місць.

1.4.6 Орієнтовний план реалізації обраного варіанта концепції системи

Орієнтований план реалізації обраного варіанта концепції створення системи зображено в таблиці 1.11:

Таблиця 1.11 - Орієнтований план реалізації обраного варіанта концепції системи

Етапи розробки	Терміни виконання
Технічне завдання (визначення й аналіз вимог)	01.09.2023 – 21.09.2023
Ескізне проектування (розробка специфікацій)	22.09.2023 – 15.10.2023
Технічне проектування	16.10.2023–15.11.2023
Робоче проектування (кодування, налагодження, тестування)	16.11.2023 – 30.11.2023
Впровадження	1.12.2023 – 19.12.2023

▪

1.4.7 Умови приймання системи

Приймання створеної системи здійснюється представниками виконавця і замовника відповідно до технічного завдання. Складається двосторонній акт, що стає підставою для розрахунків

Висновки до розділу 1

Проведено дослідження технологій збирання та систематизації даних з веб-сайтів. За допомогою функціонально-вартісного аналізу який складається з двох основних етапів: функціональний аналіз та вартісний аналіз визначено концепцію системи.

РОЗДІЛ 2. РОЗРОБКА ПРОЕКТНИХ РІШЕНЬ ПО СИСТЕМІ

2.1 Загальносистемні проектні рішення

2.1.1 Організаційна і функціональна структура

У процесі свого функціонування система взаємодіє з для виконання ними різних функцій. Схема цієї взаємодії, побудована в нотації варіантів використання системи (use case) показана на рисунку 2.1.

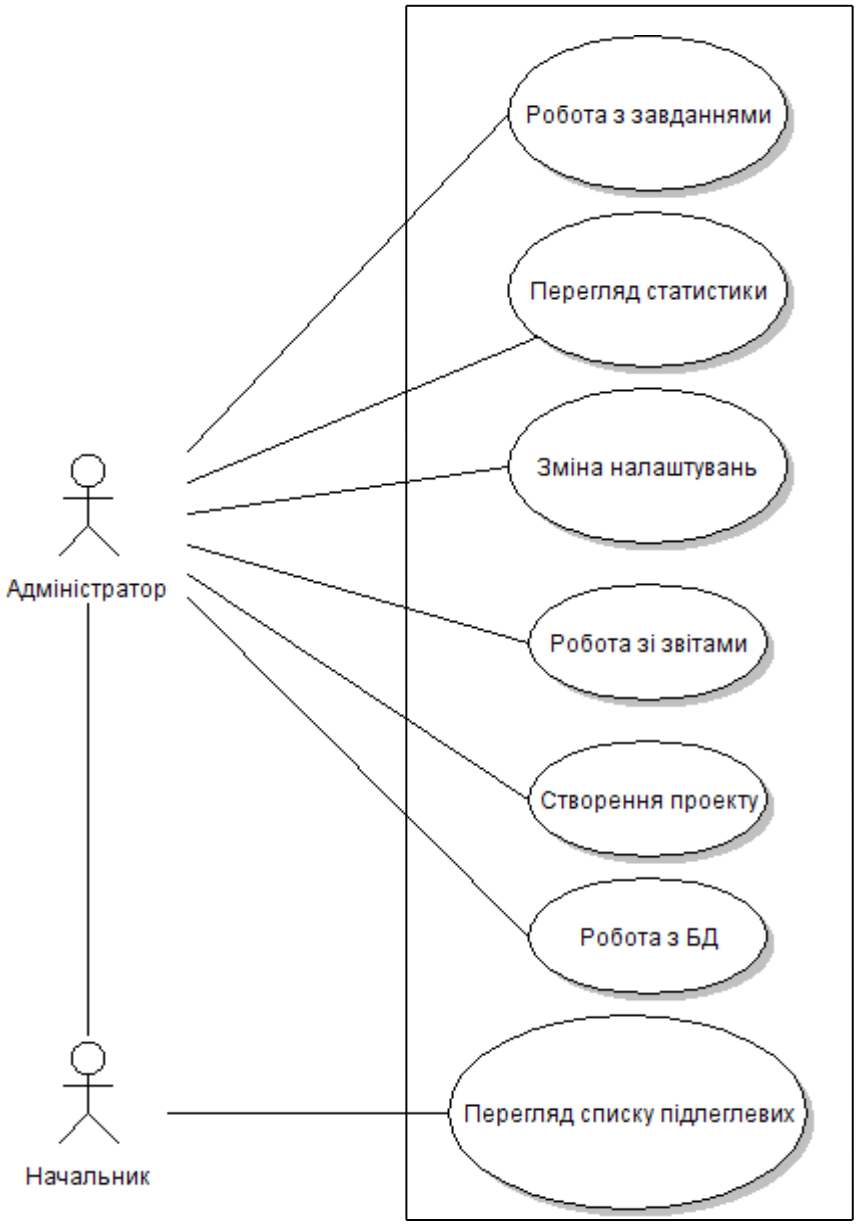

Рисунок 2.1. Схема організаційно-функціональної структури

У даному випадку, користувач є і адміністратором свого проекту і користувачем.

У ході своєї роботи, працівники підрозділів використовують наступні функції

Користувач –перегляд нових завдань, створення нового звіту та перегляд минулих, перегляд особистої статистики, та налаштування, створення проекту, робота за БД та побудова навігації Web-ресурсу.

2.1.2 Опис функцій, що автоматизуються

Задачею інформаційної системи, що створюється, є автоматизація наступних функцій користувача:

- аналіз інформації для полегшеної обробки даних
- Облік завдань та інформацію про їхній статус
- Формування звітів
- Зменшення потреби платформи до системних вимог комп'ютера

Усі інші вимоги не є головними і можуть змінюватися під час розробка або експлуатації

Для створення інформаційної системи необхідно розробити базу даних для збереження інформації про завдання, про отриманні працівниками завдання і іншу. Також повинно бути передбачене програмне забезпечення для адміністрування системи.

2.1.3 Опис постановки задачі

На підставі аналізу вимог і обмежень, а також попередніх концепцій реалізації системи, сформулюємо опис постановки задачі. Необхідно розробити систему управління завданнями, що буде задовольняти перерахованим вище функціональним вимогам. Необхідно також забезпечити надійність збереження і безпека доступу до інформації системи. Система повинна обслуговувати безперебійну роботу користувачам, причому зміни, внесені користувачами повинні бути доступні негайно.

Програмне забезпечення повинне бути побудоване на платформі CentOS Linux /Apache2.4/MySQL/JBoss/Java/Eclipse3.2 і реалізовувати графічний інтуїтивно зрозумілий WEB інтерфейс користувача.

2.1.4 Проектна оцінка надійності системи

Для оцінки надійності Системи використовується принцип визначення показників надійності системи за характеристиками надійності комплектуючих елементів, що дозволяє вести розрахунок в процесі проектування системи, виходячи з надійності елементів та вузлів.

Виходячи з важливості серверної станції для функціювання підсистеми загалом, доцільно визначити надійність саме серверної станції. Можна виділити наступні компоненти серверної станції, вихід з ладу яких є найбільш ймовірним:

- блок живлення;
- Smart-UPS APC 420;

Потрібно розрахувати ймовірність безвідмовної роботи кожного з перелічених вище компонентів, а також надійність підсистеми в цілому.

У розрахунку використовуються дані, приведені в технічній документації кожного з компонентів групи елементів.

Середній час роботи блоку живлення ПК:

$T_{ср}$ = 10 000 годин.

Середній час роботи Smart-UPS APC 420:

$T_{ср}$ = 12 000 годин.

Розрахунок буде проводитися для визначення ймовірності безвідмовної роботи кожного з перелічених елементів протягом 1000 годин за формулою

де t – час, протягом якого розраховується надійність;

λ_0 – інтенсивність відмов елемента підсистеми протягом середнього часу безвідмовної роботи $T_{ср}$.

Інтенсивність відмов елемента підсистеми за середній час безвідмовної роботи $T_{ср}$ розраховується за формулою $\lambda_0 = \frac{1}{T_{ср}}$

Для блоку живлення:

$\lambda_0 = 1 / 10\,000 = 0{,}0001$

$P(t)_{рс} = \exp(-0{,}0001 * 1000) = 0{,}920$

Для UPS:

$\lambda_0 = 1 / 12\,000 = 0{,}000083$

$P(t)_с = \exp(-0{,}000083 * 1000) = 0{,}943$

Під час визначення надійності підсистеми $P_п(t)$ через відомі показники надійності її елементів вводяться два припущення: відмови елементів статично незалежні, відмова будь-якого елемента приводить до відмови підсистеми. Прийняті припущення дозволяють використовувати теорему добутку ймовірностей, яка розраховується за формулою

$$P_п(t) = \prod_{i=1}^{n} P_i(t),$$

де $P_i(t)$ – ймовірність безвідмовної роботи і-го елемента;

n – кількість елементів.

Надійність технічного забезпечення Системи протягом 1000 годин роботи визначається добутком ймовірностей безвідмовної роботи блоків живлення, UPS та комутатора.

$P_п(t) = P(t)_{рс} * P(t)_с * P(t)_{мс} = 0{,}920 * 0{,}943 = 0{,}867$

Проведений розрахунок показав, що ймовірність безвідмовної роботи Системи протягом 1000 годин складає більше ніж 85%.

2.1.5 Загальний опис системи

Система являє собою інструмент керування процесом виробництва програм, побудованих на основі мережі підприємства «ТОВ». Основними функціональними характеристиками системи є:

- ведення загальної інформаційної бази;
- авторизований доступ до даних на основі прав доступу;
- відображення змін;
- можливість створення і редагування завдань і оперативного контролю ходу їхнього виконання;

Система має звичний графічний інтерфейс користувача, що знижує витрати на навчання персоналу, і побудована з максимальним використанням існуючого на підприємстві в даний момент устаткування і програмного забезпечення.

2.1.6 Захист від несанкціонованого доступу

Захист системи від несанкціонованого доступу реалізовано на системному рівні. При запуску системи відбувається авторизація користувача. Доступ до даних і функцій дозволяється тільки при правильному введенні імені користувача і пароля.

2.2 Рішення по організаційному забезпеченню

2.2.1 Організаційна структура підприємства

У ході впровадження і використання системи, її функціями охоплюються всі суб'єкти підприємства. Схема організаційної структури підприємства при взаємодії із системою (у виді діаграми варіантів використання) приведена на рисунку 2.2

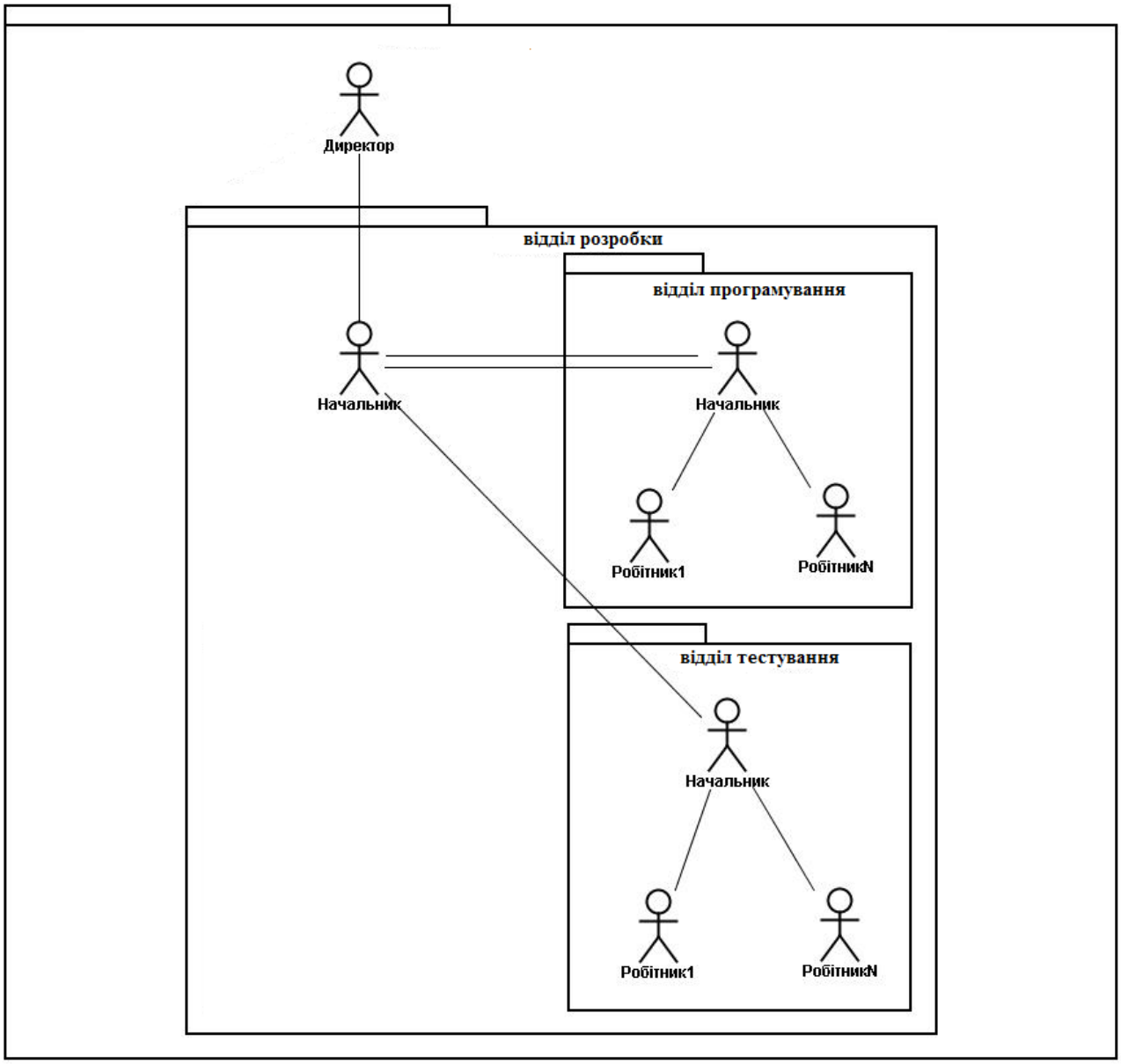

Рисунок 2.2.- Суб'єкти підприємства, які взаємодіють із системою.

2.2.2 Інструкція користувача

Для нормального функціонування системи, всі операції, проведені в ній користувачами, повинні відповідати Інструкції користувача, приведеної в Додатку Б.

2.2.3 Заходи щодо організаційного забезпечення

Ефективна робота системи керування діяльністю підприємства вимагає організаційних заходів, цілями яких є підтримка даних про процеси, що відбуваються, в актуальному стані і грамотній своєчасній реакції користувачів на інформацію, що її надається системою:

– навчання користувачів роботі в системі згідно виконуваних функцій (посадових обов'язків);

– досвідчена експлуатація системи по будівництву програми, що була побудована раніше;

– мінімалізація часу, що проходить між зміною версій документів і зміни їхнього стану (іншими словами, негайна реакція керівництва на зміни у виробничому процесі);

– обов'язкова участь у формуванні даних усіх суб'єктів підприємства; регулярний оперативний контроль зв'язаних процесів керівниками підрозділів і контроль виробничих циклів.

2.2.4 Організаційно-кадрові заходи

Для підтримки апаратних і програмних засобів у робочому стані, а також рішення поточних задач настроювання, установки і відновлення загальносистемного програмного й апаратного забезпечення на підприємстві повинна бути введена штатна одиниця системного адміністратора, ща являє собою користувача, або для цих цілей застосовуватися аутсорсінг (outsourcing – надання IT-функцій сторонніми організаціями).

2.3 Рішення по інформаційному забезпеченню

2.3.1 Опис інформаційного забезпечення системи

Одну з основ інформаційного забезпечення системи складає визначення її внутрішнього стану, що буде відбито в моделі класів. На даному етапі нас цікавлять довгоживущі і постійні сутності бізнесів-процесів. Цей вид класів будемо називати класами-сутностями (чи класами предметної області).

Для створення моделі класів складемо словник даних, ґрунтуючись на сутностях, виявлених на етапі бізнесу-аналізу.

Користувач – людина яка взаємодіє з системою

Начальник – користувач, на якого є хочаб одне посилання у іншого користувача

Звіт - звіт формується користувачем, та відсилається безпосередньому начальнику

Повідомлення – користувачі можуть обмінюватися повідомленнями

Завдання – завдання зформоване начальником да користувача

Дія – зміна статусу завдання (створенне, відхиленне, то що) проходить через створення користувачем „дії”

На даному етапі проектування сутності словника даних стають кандидатами для класів проектованої системи. Модель бізнесів-сутностей системи показана на рисунку 2.3.

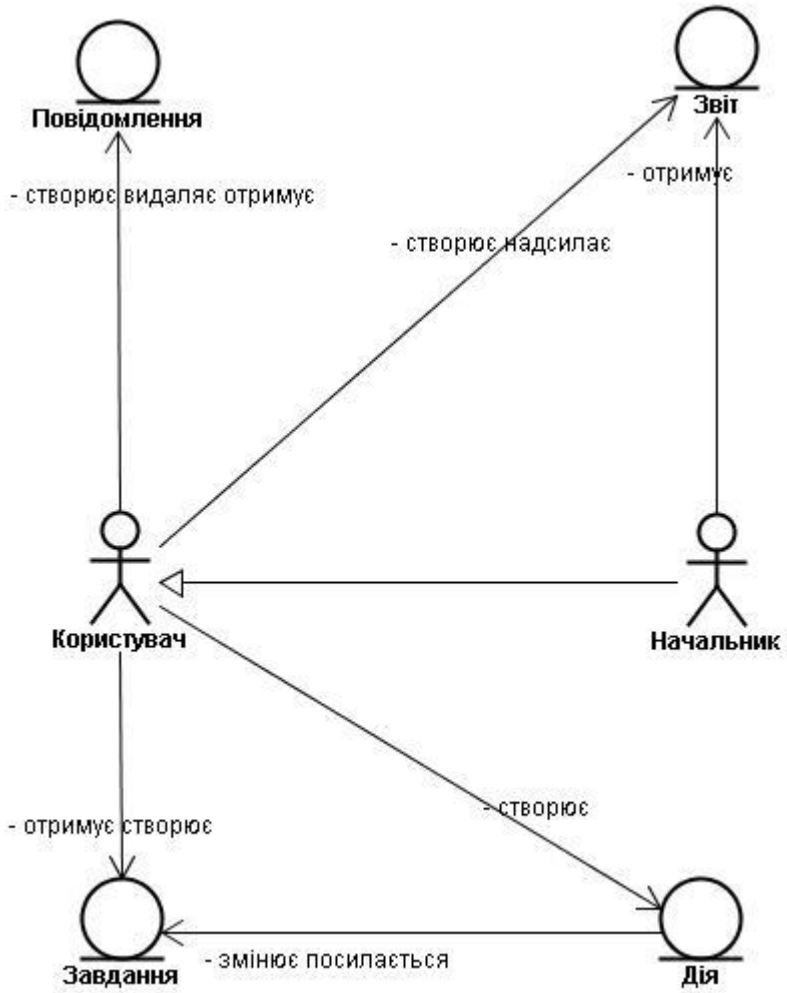

Рисунок 2.3 - Діаграма бізнес-сутностей системи.

Для подальшої деталізації сутностей і зв'язків між ними використовуємо діаграму «сутність-зв'язок» (ERD), що дозволить найбільше повно представити

концептуальну модель даних, а також перейти в наслідку до їх логічного представлення. Побудована на підставі аналізу бізнесів-сутностей системи модель у термінах «сутність-зв'язок» у нотації UML приведена на рисунку 2.4.

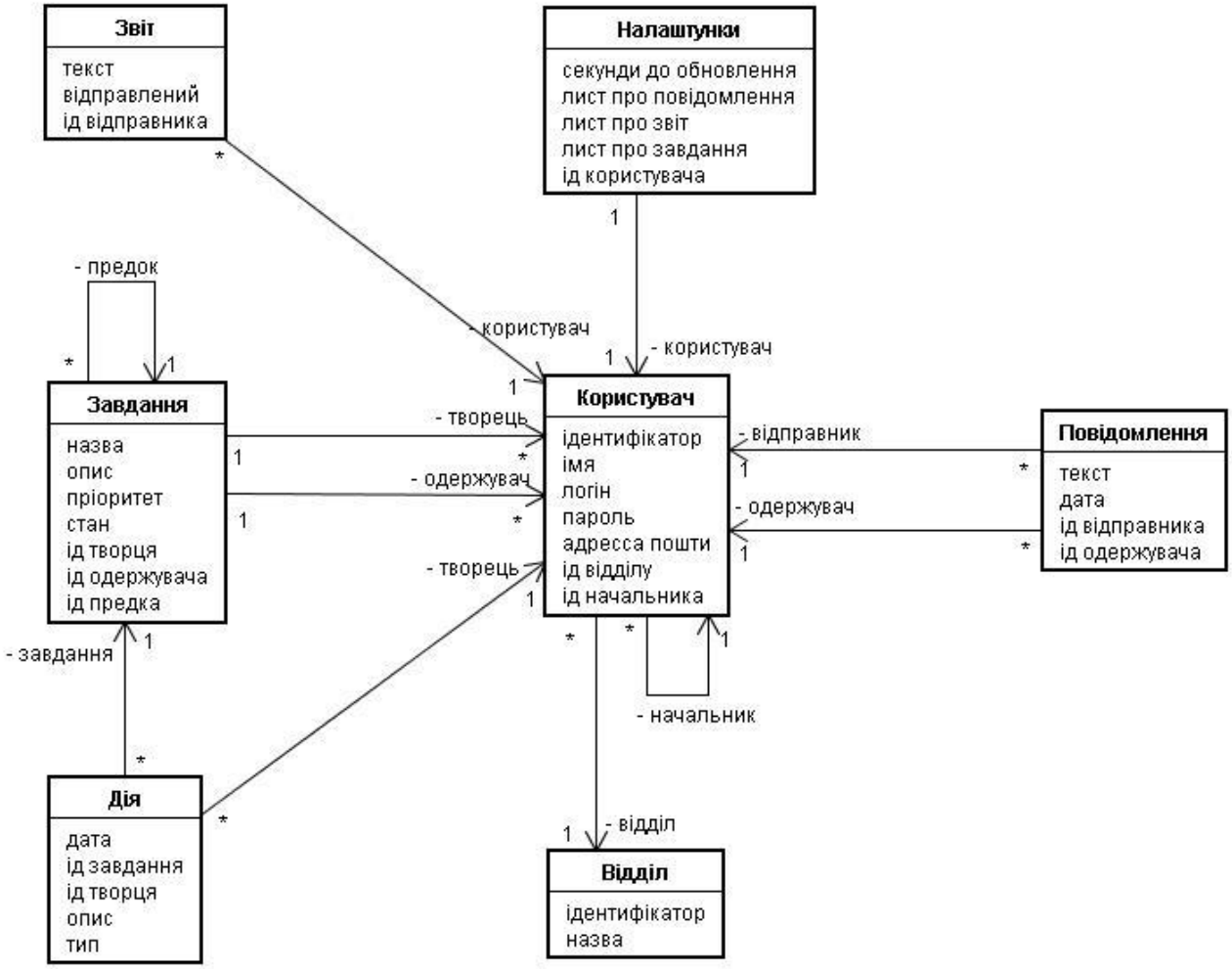

Рисунок 2.4. - Інформаційна модель об’єкту управління.

2.4 Рішення по технічному забезпеченню

2.4.1 Схема автоматизації

Раціональний вибір технічних засобів, необхідних для впровадження автоматизованої системи буде значною мірою визначати загальний успіх чи невдачу проекту. На підставі прийнятих загальносистемних рішень були визначені склад і організація комплексу технічних засобів.

Вимоги до продуктивності:

Обчислювальний комплекс повинний забезпечувати достатню продуктивність для рішення передбачуваних задач при бажаній мінімізації фінансових витрат на його придбання. Крім того, потрібно забезпечити сприятний час реакції системи при роботі в інтерактивних режимах, що складають значну частину від всіх експлуатаційних режимів. Цим буде забезпечена комфортна робота користувачів. Проектована система буде пред'являти підвищені вимоги до обчислювальної потужності. У такий спосіб, тільки сучасне апаратне рішення з пропонованих на ринку буде відповідати вимогам до продуктивності. Розподіл програмних компонентів майбутньої системи дозволить не збільшити навантаження на існуючі клієнтські робочі місця і не висувати вимог до їх модернізації.

Вимоги до надійності:

Розв'язувана задача висуває стандартні вимоги до надійності апаратних засобів без вимоги забезпечення повної отказоустойчивість. Основною вимогою в даній групі буде вимога до схоронності даних. Передбачаються наступні заходи для його виконання:

- включення сервера в електромережу через пристрій безперебійного живлення (UPS);

- використання спеціального пристрою резервного копіювання; як такий пристрій можливе застосування CD-RW привода, або окремого сховища даних.

Вимоги до експлуатаційних характеристик:

З перерахованих вимог можна зробити висновок про застосовність для сервера БД і клієнтських робочих місць стандартного апаратного забезпечення на платформі x86. До сервера БД пред'являється традиційний набір вимог для такого рішення.

2.4.2 Структурна схема комплексу обчислювальних засобів

Проектована програмна система буде виконана за схемою “Клієнт-сервер”, тому апаратне рішення буде складатися з обчислювальної підсистеми і

мережі передачі даних. Принципова схема виконана у виді UML діаграми розміщення наведена на рисунку 2.5.

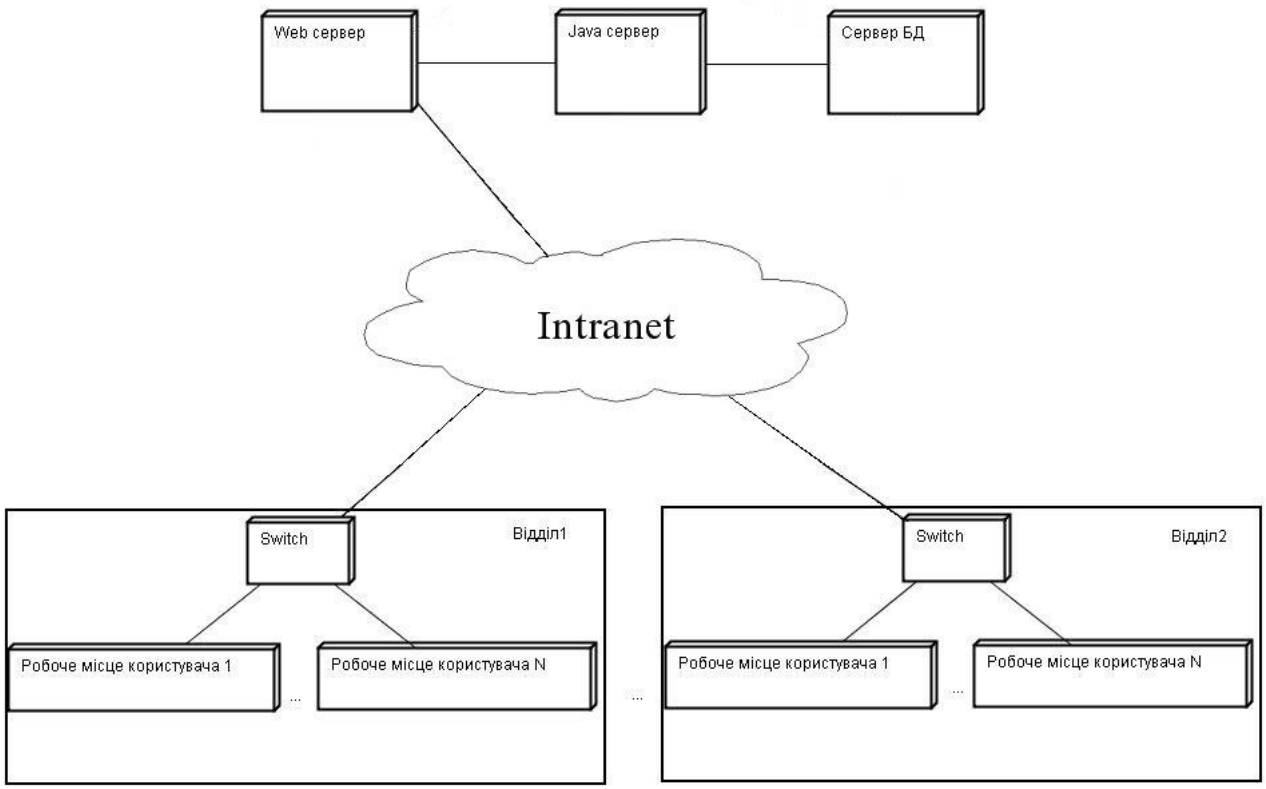

Рисунок 2.5 - Діаграма розміщення комплексу технічних засобів.

2.5 Опис алгоритмів функціонування системи

Для уточнення алгоритмів функціонування розроблювальної системи використовуємо засобу об’єктно-орієнтованного аналізу й опису поводження об'єктів. При цьому будемо розглядати як поводження системи в цілому (для уточнення виду, характеру і порядку взаємодії між класами), так і реалізацію функціональності окремих класів системи. Діаграма класів системи, побудована на підставі аналізу сутностей і операцій над ними, представлена на рисунку 2.4.

Розглянута вище діаграма класів являє собою логічну модель статичного представлення системи, що моделюється. Мова йде про те, що на даній діаграмі зображуються тільки взаємозв'язки структурного характеру, що не залежать від часу реакції системи на зовнішні події. На відміну від цього представлення, загальний алгоритм поводження системи може бути описаний за допомогою діаграми станів (statechart diagram). Загальний алгоритм поводження важливий для визначення взаємодій між компонентами системи й уточнення причин, що викликають та чи інша відповідна дія. Алгоритм поводження системи в нотації діаграми станів приведений на рисунку 2.6.

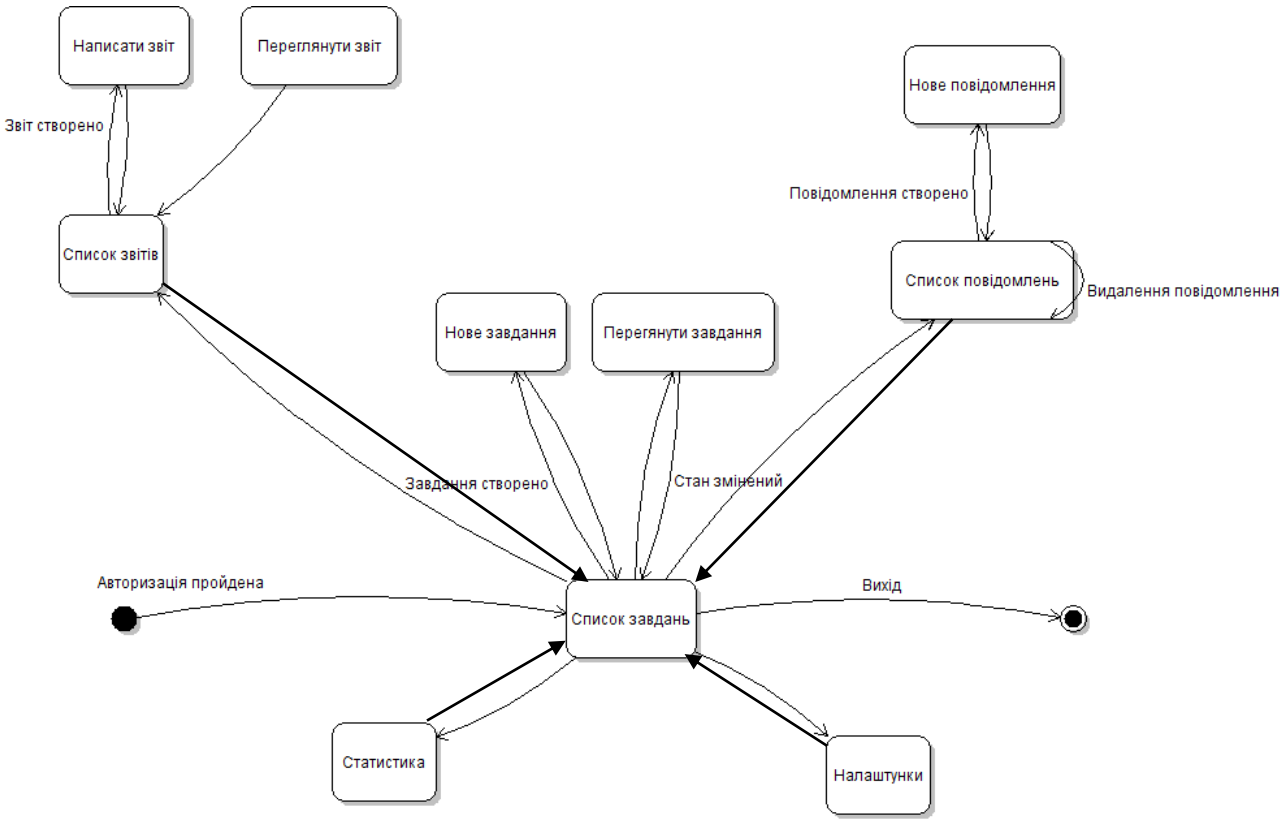

Рисунок 2.6 - Загальний алгоритм поводження системи.

Після успішного входу в систему користувач бачить екран з двома списками завдань, перший завдання які він отримав, другий які він створив. При переході по ссилці на перегляд отриманного завдання, користувач може відмовитися від нього, завершити або створити нове підзавдання. Якщо ж відбувається перегляд створенного користувачем завдання тоді він може його видалити. Усі перелiковані дії повертають на попередній екран.

З цього ж екрану користувач може перейти:

- на екран де може змінити свої налаштунки (кількість секунд для обновлення екрану завданнів, отримувати листа при отриммані завдання, звіту або повідомлення);

- на екран статистики де відображується статистика за обраний період, кількість виконаних та отриманих завдань, а також їх перелік;

- на екран звітів, де бачить свої звіти, створити новий звіт або відіслати існуючий начальству;

- на екран повідомлень, де бачить повідомлення адресованні йому, або надіслані ним, він може їх видалити, створити нове;

Користувач може вийти з системи, користувач може завершити дію веб оглядача, але у цілях безпеки рекомендується виходити з системи.

2.6 Рішення по програмному забезпеченню

2.6.1 Структура і функції частин програмного забезпечення

Функціональність і структуру проектованої автоматизованої системи поряд з моделлю поводження визначить модель її станів. Будемо будувати модель станів шляхом уточнення моделі бізнес-сутностей, побудованої нами на підставі вимог користувачів до функцій системи й оброблюваних даних. Вхідними даними для даного етапу проектування будуть моделі сутностей, а результатом стане модель специфікації проектних конструкцій розроблювального програмного забезпечення. У відповідності з прийнятим для даної роботи підходом об’єктно-орієнтованного аналізу і проектування виконаємо подальшу специфікацію проектних рішень у виді діаграм класів. Класи визначать стан пропонованої програмної системи (через значення своїх атрибутів) і, значною мірою, її поводження (через операції чи методи класів). Діаграма бізнесів-сутностей приведена на рисунку 2.6.

Сформульовані користувальницькі вимоги тепер приймемо за обмеження. Поряд з моделлю сутностей для проектування класів будемо користатися побудованими раніше діаграмами прецедентів (рисунок 2.1), у відповідності до рекомендацій методики Rational Unified Process. При цьому

модель класів буде відповідати словнику даних, що був отриманий на підставі аналізу прецедентів, і який є текстовим аналогом моделі сутностей.

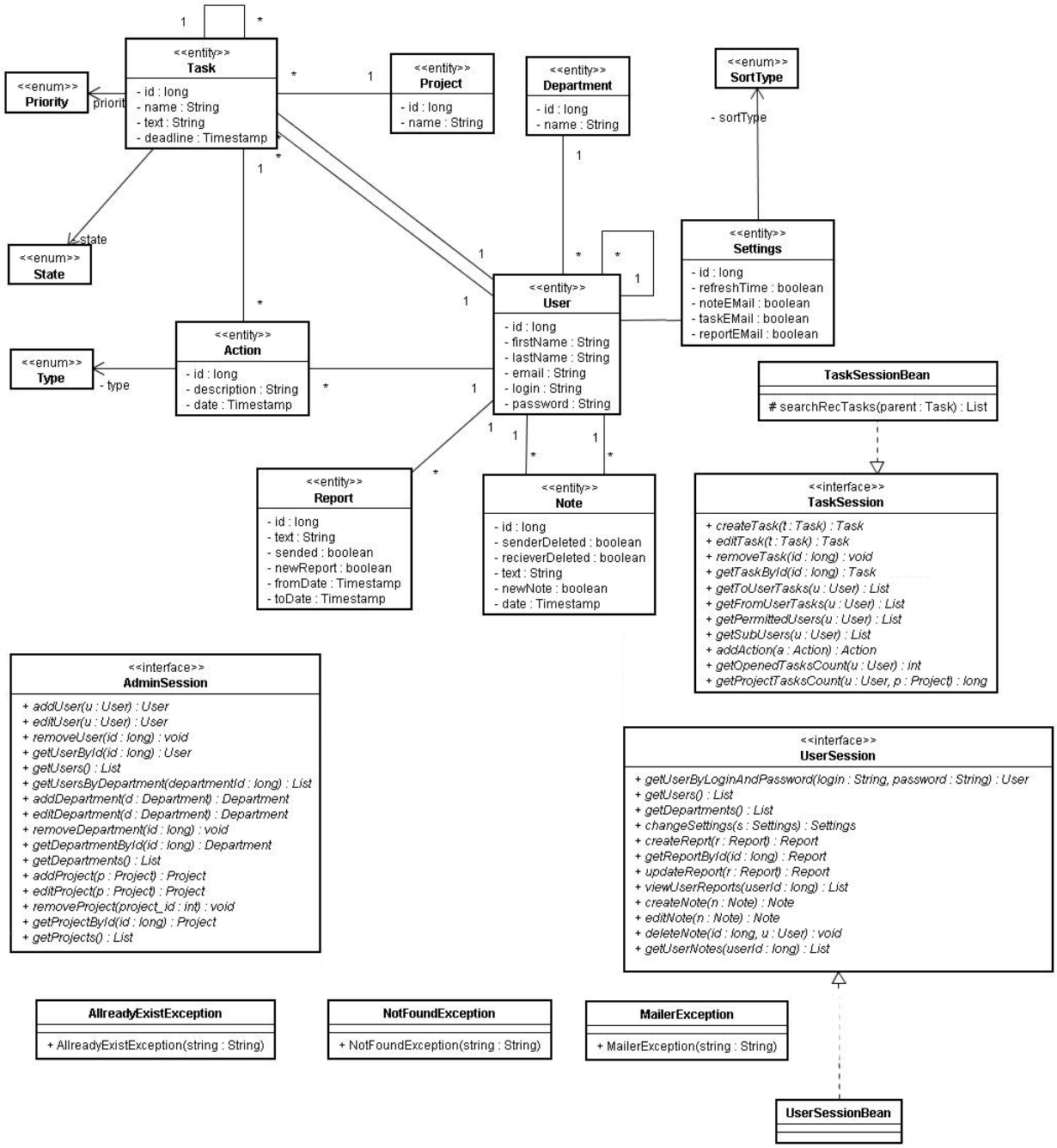

Рисунок 2.7 - Діаграма класів системи

Укажемо відповідальність кожного з приведених на діаграмі класів:

User (Користувач) – людина яка взаємодіє з системою

Settings (Начальник) – користувач, на якого є хочаб одне посилання у іншого користувача

Report (Звіт) - звіт формується користувачем, та відсилається безпосередньому начальнику

Note (Повідомлення) – користувачі можуть обмінюватися повідомленнями

Task (Завдання) – завдання зформоване одним користувачем до іншого

Action (Дія) – зміна статусу завдання (створене, відхилене, то що) проходить через створення користувачем „дії”

CharEncodingFilter – фільтр який преобразує дані яки приходять від Http запиту та відповіді

LookupUtil – клас реалізуючий технологію JNDI для отримання екземплярів SessionBean класів у веб модулі

AdminSession – інтерфейс описуючий дії адміністратора системи

AdminSessionBean - клас який його реалізує

UserSession – інтерфейс описуючий дії користувча, як то створення звіту, відправлення повідомлення то що

UserSessionBean – клас який його реалізує

TaskSession – інтерфейс описуючий методи для роботи з завданнями

TaskSessionBean – клас який його реалізує

NotFoundException – виключення екземпляр якого створюється якщо не знайдений екземпляр у базі даних

AllreadyExistException – виключення екземпляр якого створюється якщо не може бути створений обєкт (користувач з тим логіном який вже використовується системою)

2.6.2 Логічна модель бази даних

Для забезпечення можливості зберігати дані в реляційній базі даних, з моделі класів була отримана логічна модель даних. Для цього серед усіх класів

були визначені постійні. Це такі класи, об’єкти яких повинні зберігати свої значення не тільки під час роботи програми. Далі було проведено відображення постійних класів на таблиці моделі даних. При цьому атрибути класів відобразилися в поля реляційних таблиць. Асоціації між класами змінилися в зв’язки між реляційними таблицями з використанням механізму зовнішніх ключів. Для забезпечення можливості підтримки цілісності даних по суттєвостям були використані первинні ключі. Обрана СКБД не підтримує зберігання ієрархії спадкування класів, тому при будуванні логічної моделі даних ієрархії відображалися на таблиці за принципом одна таблиця на клас. Метод згортання типів в супертип при якому ієрархія змінюється в одну таблицю був відхилений із-за великої надмірності даних. Розроблена логічна модель наведена на рисунку 2.9 .

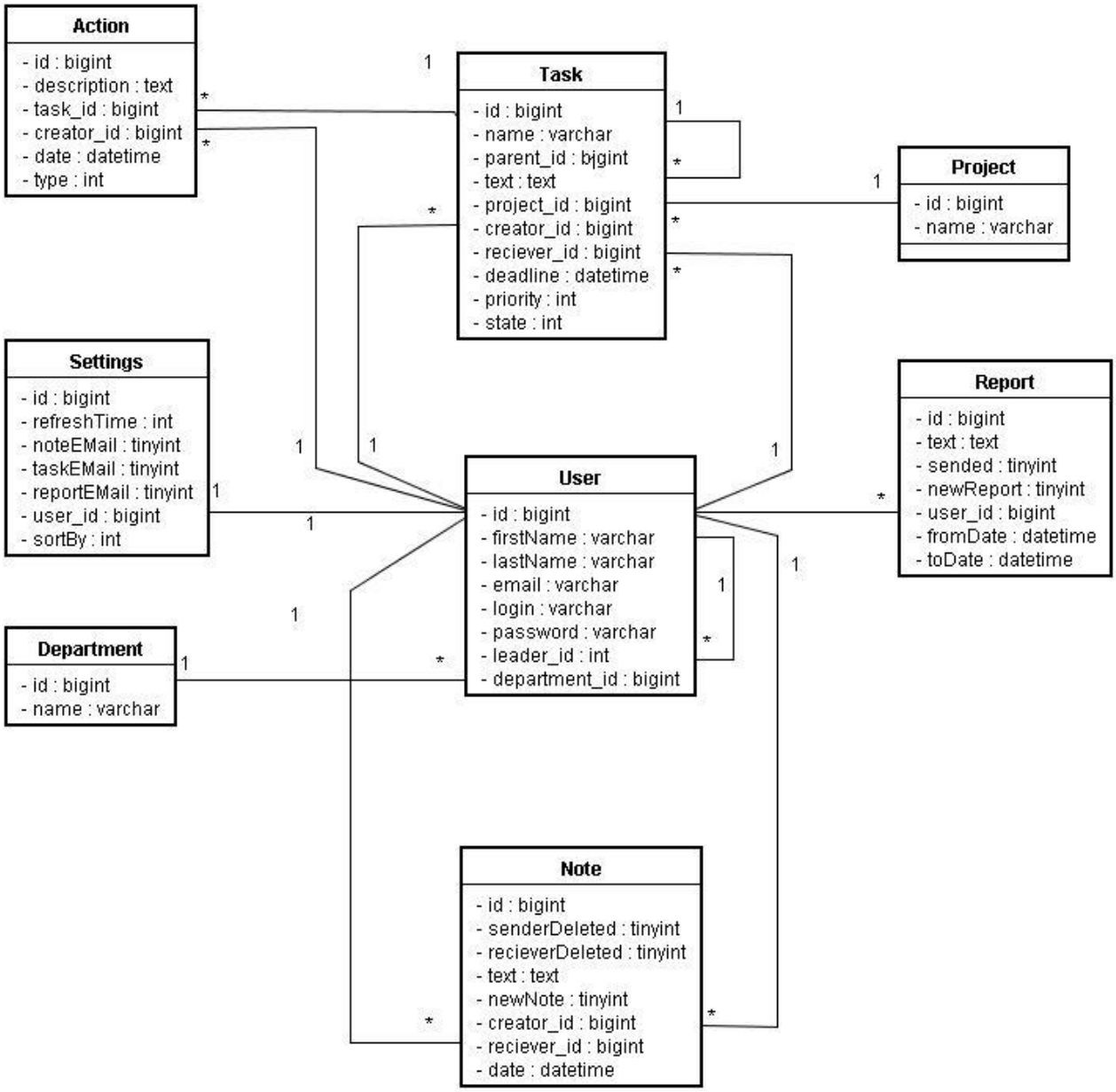

Рисунок 2.8 - Логічна модель бази даних

2.6.3 Методи і засоби розробки програмного забезпечення

При аналізі і проектуванні системи застосовувався об’єктно-орієнтованний підхід (ООП), виходячи з чого програмне забезпечення системи повинне бути розроблене з його використанням. Основними принципами ООП є спадкування, інкапсуляція і поліморфізм.

Використання ООП у розробці дозволяє:

- використовувати отримані в результаті аналізу і проектування логічні моделі для первинної кодогенерації;

- скоротити час розробки за рахунок повторного використання коду програмних компонентів;
- проводити налагодження і тестування ґрунтуючись на поводженні об'єктів, без знання їхньої внутрішньої структури;
- цілком використовувати потенціал обраних засобів розробки ПО.

Для проектування й аналізу системи використовувалася CASE-система Rational Rose виробництва компанії Rational Software. На сьогоднішній день це один із самих могутніх CASE-засобів на ринку і де-факто стандарт галузі. Rational Rose використовується на всіх етапах проектування від аналізу до автоматичної генерації коду програмної системи і має засобу для роботи з всіма інструментами, наданими мовою UML.

Основним засобом розробки програмного забезпечення системи є мова програмування високого рівня Java, використовуються такі технології: Servlets, EJB3, JSPx та Struts. У якості середовища був векористаний продукт Eclipse v3.1

На Web-сервері повинен бути встановлений HTTP-сервер. Найрозповсюдженими HTTP-сервера, що задовольняючі зазначеним вимогам і мають високу продуктивність, є Apache Server. На Web-сервері повинна бути встановлена ОС яка підтримує цей сервер.

Так як використовуються технології організовані на Java то операційна система на сервері доступу до БД може бути будь яка з перелічиних:

- Macintosh
- MS Windows
- Linux

Також потрібний Java WEB сервер встановлений на сервері будь який з:

- JBoss
- Sun Application Server

На сервері баз даних повинна бути встановлена серверна система управління базами даних (СУБД).

Необхідна сучасна серверна реляційна СУБД, що має високу продуктивність, здатна працювати з великими обсягами інформації і підтримує стандарт SQL-92. Крім того, СУБД повинна працювати під управлінням Windows. Серверними СУБД, що задовольняють зазначеним вимогам, на сьогодні є:

- MS SQL Server.
- MySQL.

Програмне забезпечення, з яким буде працювати користувач, являє собою Web-оглядач з підтримкою HTML 4.1, JavaScript 1.4, CSS 2 (наприклад, MS Internet Explorer 11, FireFox 12, Opera 10.5, ...).

Висновки за розділом 2

РОЗДІЛ 3.РОЗРОБКА РОБОЧОЇ ДОКУМЕНТАЦІЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ ЗБИРАННЯ ТА СИСТЕМАТИЗАЦІЇ ДАНИХ З ВЕБ-САЙТІВ

Інформаційна система збирання та систематизації даних з веб-сайтів призначена для автоматизації роботи працівників підприємства на будь якому рівні, від начальника відділу до працівника і не тільки. Функції, що автоматизуються системою, пов'язані з організацією робочого часу працівників, отримання завдань та іншого документообігу, як то звіт та статистичні показники працівників.

Основними функціями, що автоматизуються є:

- Облік відділків та їх працівників.
- Облік завдань та інформацію про їхній статус.
- Формування звітів.
- Усі інші вимоги не є головними і можуть змінюватися під час розробки або експлуатації.

При впровадженні програмного забезпечення системи основними цілями є:

- простий і зручний інтерфейс користувача
- доступ до системи через Internet з будь-якого комп'ютера глобальної мережі
- висока масштабуємість системи.
- висока продуктивність
- надійність
- конфіденційність інформації

Для ефективного використовування програмного забезпечення важливо впровадити на підприємстві організаційні і технічні заходи, що передбачено в п. 2.2 та п. 2.4. Тому що підвищується рівень складності експлуатованого ПО, необхідно відповідне підвищення кваліфікації співробітників.

3.1 Результати розробки інформаційної системи збирання та вилучення даних з веб-сайтів

Спочатку зайшовши на веб-сайт ПП «ТОВ» працівник бачить перед собою захист системи від несанкціонованого доступу, який реалізовано на системному рівні. При запуску системи відбувається авторизація користувача. Доступ до даних і функцій дозволяється тільки при правильному введенні імені користувача і пароля.

Авторизація користувача відбувається прямо на першій сторінці веб-сайта, як це ми можемо побачити на Рисунку 3.1

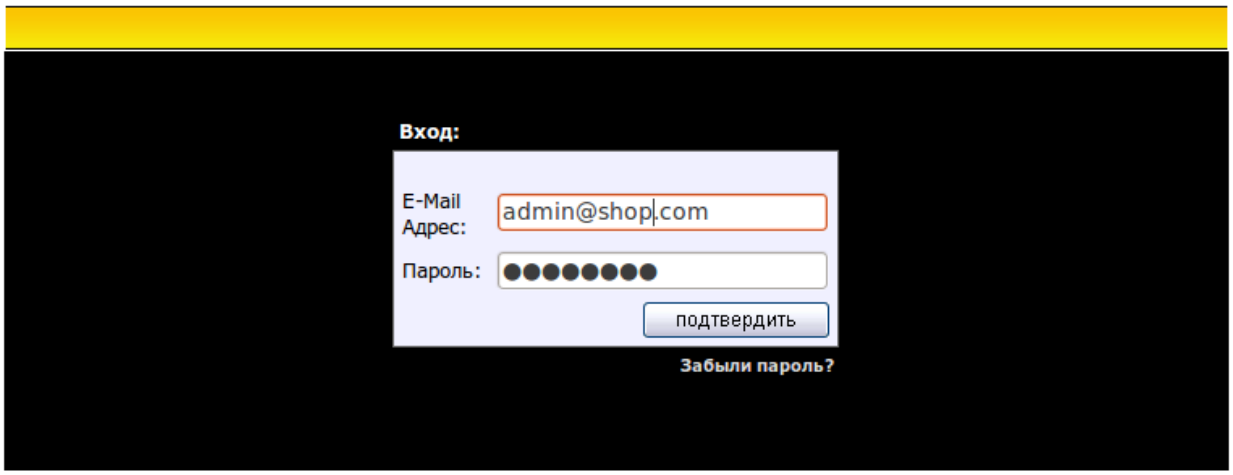

Рисунок 3.1 Авторизація користувача

Після успішного входу в систему користувач бачить екран з двома списками завдань, перший завдання які він отримав, другий які він створив. При переході по лінці на перегляд отриманного завдання, користувач може відмовитися від нього, завершити або створити нове підзавдання. Якщо ж відбувається перегляд створенного користувачем завдання тоді він може його видалити. Усі перелічені дії повертають на попередній екран.

З цього ж екрану користувач може перейти:

- на екран де може змінити свої налаштунки (кількість секунд для обновлення екрану завдань, отримувати листа при отримані завдання, звіту або повідомлення);

– на екран статистики де відображується статистика за обраний період, кількість виконаних та отриманих завдань, а також їх перелік;

– на екран звітів, де бачить свої звіти, створити новий звіт або відіслати існуючий начальству;

– на екран з формою для скачування товарів, які замовив клієнт.

Працівник отримав завдання і йому потрібно зібрати та систематизувати данні з веб-сайту, який вказав клієнт. Для цього працівнику необхідно в форму на веб сторінці ввести лінк з товаром та вибрати потрібну кількість картинок які потрібно викачати з веб-сайта для потрібного клієнту товару. Форма для збирання даних показана на Рисунку 3.2.

Рисунок 3.2 Форма для збирання даних з веб-сайтів

Після того як працівник ввів лінк на потрібній товар, необхідну кількість картинок, номер категорії товару він може натиснути кнопку «Отправить запрос». Через декілька секунд повертається інформація з потрібного веб-сайту. Працівник може побачити її нижче заповненої форми, як зображено на Рисунку 3.3

21-54 Сохранено 3 файлов, объемом 158.8Kb.

in_set=C46-bali/regal blue','AE5-red/orange','DL3-black

values[products_options_values_name]=dl3-black уже есть
values[products_options_values_name]=DL3-black уже есть
insert_val=('715','1','C46-bali/regal blue'),('716','1','AE5-red/orange'),
insert_val_link=('3','715'),('3','716'),

```
<div id="swatchContainer0" class="swatchContainer">
                        <script type="text/javascript">
                        //var mainSwatchContainer ='swatchContainer0';
                        </script>

                        <div onmouseout="hideSwatch(0,this);" onmouseover="moveSwatch(0,this,'C46-bali/regal blue','V835986_C46', false,'false');" class="swatch">
                              <div class="hide"><div class="swatch-black-border"></div><div class="swatch-white-border"></div></div>
                              <a class="swatchLink" href="javascript: void {};">
```

21-54 Сохранено 3 файлов ткани small, объемом 4.9Kb.
21-54 Сохранено 3 файлов ткани large, объемом 129.8Kb.

Рисунок 3.3 Дані зібрані з веб-сайту

Наступним кроком працівника є те, що він систематизує отриманні данні. А потім за вимогою клієнта або зберігає ці дані в .xml форматі або в базі даних, яку створюють на вимогу клієнта, якщо він цього забажає. І наприкінці виконаної роботі працівник пише звіт про виконану роботу.

Висновки до розділу 3

Було проведено розробку інформаційної системи збирання та вилучення даних з веб-сайтів. Описано головні правила користування системою.

РОЗДІЛ 4. РОЗРАХУНОК ЕКОНОМІЧНОГО ЕФЕКТУ ВІД ВПРОВАДЖЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ УПРАВЛІННЯ ЗАВДАННЯМИ

4.1 Загальні положення

На даний момент сучасні обчислювальні засоби широко використовуються в повсякденній діяльності людини. Сьогодні обчислювальна техніка дозволяє вирішувати найрізноманітніші задачі: від звільнення людини від монотонної ручної праці до розробки складних технічних розрахунків, що не під силу виконати одній людині. Сучасні засоби програмування дозволяють вирішувати задачі, що стояли в недавнім минулому перед цілими колективами наукових співробітників.

Розвиток нових технологій і ріст застосування обчислювальної техніки привели до широкого застосування ЕОМ для вирішення проблем, пов'язаних з обробкою інформації, її контролем і аналізом. Сьогодні на основі засобів обчислювальної техніки розробляються і впроваджуються різні автоматизовані інформаційні системи (управління, проектування, технологічної підготовки виробництва). Успіх у впровадженні цих систем, їхня роль в інтенсифікації розвитку народного господарства нашої країни багато в чому залежить від фахівців із системотехніки, які знають методику аналізу і проектування цих систем, можливості обчислювальної техніки і які володіють математичними методами, що використовуються при постановці та вирішенні задач.

Ініормаційна система управлгння завданнями призначена для автоматизації роботи процівників підприємства на будь якому рівні, від начальника відділу до процівника і нетільки. Функції, що автоматизуються системою, пов'язані з організацією робочого часу працівників отримання завданів та іншого документообігу, як то звіт та статистичні показники працівників

Найбільш важливим моментом для розробника, з економічної точки зору, є процес формування вартості системи. Очевидно, що вона являє собою дуже специфічний товар з безліччю особливостей.

Створення автоматизованої системи вимагає одноразових витрат на її розробку, придбання необхідних технічних засобів і поточних витрат на функціонування системи. Економія від функціонування автоматизованої системи визначається з урахуванням витрат на її експлуатацію. Відношення цієї економії до витрат на створення автоматизованої системи характеризує економічну ефективність капітальних вкладень. Економічні показники визначаються по

діючим на момент розрахунку оптовим цінам, тарифам і ставкам заробітної плати.

4.2 Розрахунок витрат на створення і впровадження системи

Витрати на розробку системи складаються з витрат на зарплату розробника, на амортизацію ЕОМ, на якій виконується розробка, на експлуатацію цієї ЕОМ, на засоби розробки та витрат на матеріали і комплектуючі.

Виходячи з того, що основна заробітна плата розробників програмного забезпечення складає 24000 грн./міс, вартість сучасної ПЕОМ складає 30000 грн. і вартість кіловат-години електроенергії складає 2,64 грн, розрахуємо вартість розробки системи.

1) Розрахунок заробітної плати зробимо по формулі 4.1:

$$З_{зп} = ЗП_{раз} \cdot T_{раз}\ , \qquad (4.1)$$

де $ЗП_{раз}$ – зарплата розробника, за місяць;

$T_{раз}$ – тривалість розробки (дослідження, створення, налагодження і впровадження).

Для розробки даної системи необхідно 6 місяців (за експертною оцінкою часу на розробку аналогічних систем). З цього випливає, що загальна сума витрат на заробітну плату складе:

$$З_{зп} = ЗП_{раз} \cdot T_{раз} = 24000 \cdot 6 = 144000 \text{ (грн);}$$

2) витрати на амортизацію ЕОМ, на якій виконується розробка, розраховуються по формулі 4.2:

$$З_{амор} = Ц_{эвм} * А * Т_{разр} \qquad (4.2)$$

де $Ц_{эвм} = 30000$ грн. – балансова вартість ЕОМ;

А = 60% - амортизація за рік**;**

$T_{разр} = 0,5$ року – час, необхідний для розробки системи, виражений в місяцях, і тому ділиться на 12, щоб одержати число років.

$$З_{амор} = 30000 * 0,6 * 0,5 = 9000 \text{ (грн);}$$

3) витрати на експлуатацію ЕОМ, на якій виконується розробка, полягають в оплаті споживаної нею електричної енергії і розраховуються по формулі 4.3:

$$З_{експл} = P_{ЭВМ} \cdot T_{разр} \cdot N_{рд} \cdot N_{рч} \cdot Э_{ст}\ , \qquad (4.3)$$

де $P_{ЭВМ} = 2,64$ кВт/год – потужність ЕОМ;

$T_{разр}$ = 6 міс. – тривалість розробки;

$N_{рд}$= 21 день – число робочих днів у місяці;

$N_{рч}$= 8 годин – число годин у робочому дні;

$Э_{ст}$ = 2,64 грн – вартість 1 кВт електроенергії.

$$З_{экспл} = 2{,}64 \cdot 6 \cdot 21 \cdot 8 \cdot 2{,}64 = 7025{,}3568 \text{ (грн);}$$

4) витрати на матеріали і комплектуючі вироби враховують витрати на магнітні носії даних, папір для друкувальних пристроїв, літературу, необхідну для розробки данної системи, картридж і тонер до принтера, а також непередбачені витрати. Розрахунок приведений у таблиці 4.1 (Ціни взяті відповідно до прайслистів та договірних цін на даний вид матеріалів).

Таблиця 4.1 – Розрахунок витрат на матеріали і комплектуючі вироби

№	Найменування виробів	Вартість, грн.
1	Папір	140.00
2	Картридж і тонер до принтера	252.00
3	USB flash	222.00
4	Література	200.00
5	Непередбачені витрати	1500.00
Разом		**2314.00**

5) для розробки системи необхідні наступні засоби розробки Eclipse SDK v3.1 – даний продукт є цілком безкоштовним, тому витрати склали 0.00грн.

6) загальний кошторис витрат на створення системи, з урахуванням п.1) – п.5) приведений в таблиці 4.3.

Таблиця 4.3 – Загальний кошторис витрат на створення системи

№	Найменування витрат	Сума, грн.
1	Заробітна плата основна	144000.00
2	Заробітна плата додаткова (20% від п.1)	28800.00
3	Витрати на амортизацію ЕОМ	9000.00
4	Витрати на експлуатацію ЕОМ	7025.40
5	Витрати на матеріали і комплектуючі вироби	2314.00
6	Витрати на програмні засоби розробки	0.00
Разом на створення системи		**191139.400**

До витрат на впровадження системи можна віднести витрати на придбання технічного забезпечення, вартість програмного забезпечення, вартість навчання кадрів, витрати на монтаж і настроювання мережі.

Оскільки склад і параметри технічних засобів, які вже є, відповідають вимогам, то їх вартість при розрахунку витрат враховувати не будемо.

Виходячи з вимог до програмного забезпечення, а також проаналізувавши цінову політику можемо прийняти наступне (див. таблицю 4.4).

Таблиця 4.4 – Перелік програмного забезпечення, необхідного для впровадження системи

№	Найменування ПО	Сума, грн.	Кількість	Разом, грн.
1	CentOS Linux	0	3	0
2	Apache	0	1	0
3	MySQL 5	0.000	1	0.000
4	Jboss	0	1	0
Разом				**0.00**

Витрати на програмне забезпечення склали **0.00** грн так як використовувалися безкоштовні продукти.

Інвестиційні витрати на впровадження системи з урахуванням вартості навчання кадрів і витрат на монтаж і настроювання мережі приведені в таблиці 4.5.

Таблиця 4.5 – Інвестиційні витрати на впровадження системи

№	Найменування інвестицій	Сума, грн.
1	Вартість устаткування	–
2	Вартість програмного забезпечення	0
3	Вартість підготовки кадрів	5000.00
4	Монтаж і настроювання локальної мережі	10000.00
5	Супроводження системи	3000.00
Разом на впровадження системи		**18000.00**

Разом загальна сума витрат на створення і впровадження системи складе 209139,40 грн.

4.3 Економічна ефективність розробки і впровадження системи

Основним показником економічної ефективності функціонування системи є підвищення ефективності керування інформацією у вигляді зниження витрат на керування при одночасному збільшенні швидкості і якості одержання потрібного результату.

Крім багатьох інших негативних ефектів, ручна обробка інформації спричиняє наступні негативні економічні ефекти:

- високі витрати на складування паперових документів (сейфи, шафи, папки й ін.);
- підвищені витрати на канцтовари;
- витрати, пов'язані з роботою виявлення раніше допущених помилок (людський фактор).

До числа основних факторів, що визначають приріст прибутку в зв'язку з упровадженням програми, відносяться:

- підвищення продуктивності праці;
- вивільнення робочого часу.

Крім того, не піддається прямій грошовій оцінці підвищення оперативності керування, якість одержуваних результатів, поліпшення організації праці і т.д.

Обов'язковою умовою визначення економічної ефективності програми є порівнянність усіх показників у часі, за цінами й іншими нормами, використовуваними для визначення показників, за змістом і колом елементів витрат.

За даними, які були отримані шляхом експериментальної оцінки, було встановлено, що використання даної системи дозволить вивільнити 0,1 робочого часу вчителя. Оскільки на даний момент на підприємстві працює 44 людини, то загалом система умовно вивільнить 4 робітника.

Визначимо пряму економічну ефективність ґрунтуючись на тому, що впровадження системи вивільнить 4 працівника .

Зарплата 4 працівників у рік складає $\Delta C = (24000*12)*4 = 96000$ (грн.)

Річний економічний ефект розраховується по формулі 4.4:

$$Э_{год} = \Delta C_n - Ц_{сопр} - E_n \cdot k \ , \qquad (4.4)$$

де E_n – нормативний коефіцієнт економічної ефективності капітальних вкладень у галузь – для обчислювальної техніки приймається – 0,6;

k – додаткові капітальні вкладення з урахуванням витрат на проектування, створення і функціонування системи

$$Э_{год} = 115200 - 300 - 0,6 \cdot 19256,00 = 103347 \text{ (грн.)}$$

Строк окупності системи розраховується по формулі 4.5:

$$T = \frac{k}{\Delta C - Цсоп} = \frac{19256,00}{115200 - 300} \approx 0,167 \text{ (року)} \qquad (4.5)$$

Отже строк окупності системи складає 2 місяці.

РОЗДІЛ 5. ОХОРОНА ПРАЦІ

5.1 Загальні положення

Під охороною праці розуміється система збереження життя й здоров'я працівників у процесі трудової діяльності, що включає правові, соціально-економічні, організаційно-технічні, санітарно-гігієнічні, лікувально-профілактичні, реабілітаційні й інші заходи.

Ці заходи мають своєю ціллю створення умов праці, що відповідають вимогам збереження життя й здоров'я працівників у процесі трудової діяльності. На практиці під охороною праці нерідко мають на увазі техніку безпеки, виробничу санітарію й т.і. Така позиція помилкова. Охорона праці - це система заходів, де техніка безпеки або гігієна праці є її складовими.

Суттєвий вплив на стан організму працівника, його працездатність здійснює мікроклімат (метеорологічні умови) у виробничих приміщеннях, під яким розуміють клімат внутрішнього середовища цих приміщень, що визначається діючою на організм людини сукупністю температури, вологості, руху повітря та теплового випромінювання нагрітих поверхонь.

Метою проведення цієї роботи є детальний розгляд та розрахунки системи освітлення та засобів захисту від електромагнітного випромінювання для певного виробничого приміщення. Також у роботі маємо висвітити питання вимог нормативних документів щодо безпечних та здорових умов праці у виробничих приміщеннях з використанням комп’ютерної техніки та детально розглянути зменшення електромагнітного випромінювання.

5.2Аналіз шкідливих та небезпечних факторів, їхній негативний вплив на роботу інформаційної системи інформаційного забезпечення міста, причини виникнення, санітарні норми

У ДСТ 12.0.003-74 "ССБТ. Небезпечні і шкідливі виробничі фактори. Класифікація", приводиться класифікація елементів умов праці, що виступають у ролі небезпечних і шкідливих виробничих факторів. Вони підрозділяються на 4 групи: фізичні, хімічні, біологічні і психофізіологічні.

Експлуатація ЕОМ супроводжується виникненням ряду шкідливих і небезпечних виробничих факторів фізичного і психофізіологічного характеру. Працівники, що обслуговують ЕОМ, піддаються під час роботи впливові наступних небезпечних і шкідливих факторів:

фізичних: підвищення рівня шуму, статичної електрики, електромагнітних випромінювань, підвищена напруженість електричних і магнітних полів, недостатня освітленість робочого місця, підвищена яскравість світла;

психофізіологічних: фізичні і нервово-психічні перевантаження (розумова перенапруга, монотонність праці, емоційні перевантаження).

Санітарними нормами і правилами встановлені наступні припустимі значення:

рівень шуму: 50 дб;

гранично припустима напруженість електростатичного поля на робочому місці при тривалості впливу на людину до 1 години: 60 кв/м.

Інтенсивність впливу неіонізуючих магнітних полів на обслуговуючий персонал (оператори, програмісти) залежить від потужності джерел випромінювання, режиму роботи, конструктивної особливості випромінюючих пристроїв, технічного стану устаткування, розташування робочого місця й ефективності захисних заходів. Гранично припустима напруженість електричної складового електромагнітного поля складає 10 В/м, магнітної складової 0.3 А/м. Тривалий вплив неіонізуючих електромагнітних випромінювань на організм людини з інтенсивністю перевищуючі гігієнічні нормативи приводить до функціональних змін з боку нервової, серцево-судинної, нейрогуморальної й іншої систем організму.

Фізичні перевантаження найчастіше торкаються зорової системи або кістково-м'язову систему (плечовий суглоб, шия, верхня частина спини). У цілому користувачі ЕОМ не зіштовхуються з якимсь особливим психічним стресом або емоційними проблемами, дослідження показують одиничні випадки дуже напружений, зухвалий стрес, роботи на ЕОМ. Проте,

рекомендується вживати заходів до запобігання надмірного дискомфорту і стомлення працюючих на ЕОМ. У більшості користувачів ЕОМ виникає кістково-м'язовий дискомфорт через статичну напругу м'язів, незручної пози або часто повторюваних одноманітних рухів. імовірність виникнення стійких порушень дуже мала, однак монотонні рухи при роботі на ЕОМ можуть викликати хронічні захворювання.

Важливим фактором надійної роботи засобів ВТ і високої працездатності обслуговуючого персоналу є кліматичні умови. Під кліматичними умовами виробничого середовища відповідно ДО ДЕРЖСТАНДАРТУ 12.1.005-76 розуміють сполучення температури, відносній вологості і швидкості руху повітря. Ці мікрокліматичні фактори як кожний окремо, так і в різних сполученнях впливають на функціональну діяльність людини, його самопочуття і здоров'я і на надійність роботи засобів ВТ.

Температура повітря є одним з основних параметрів, що характеризують тепловий стан мікроклімату. Основними джерелами теплоти в приміщеннях є ЕОМ і допоміжне устаткування, прилади освітлення, що обслуговує персонал. Потрібно враховувати і зовнішні джерела надходження теплоти (теплоту, що надходить через вікна від сонячної радіації і приплив теплоти через непрозорі конструкції, що обгороджують).

На роботу устаткування великий вплив робить відносна вологість повітря. При вологості повітря до 40% підвищується знос магнітних голівок, виходить з ладу ізоляція проводів. При вологості повітря більш 75-80% знижується опір ізоляції, змінюються робочі характеристики елементів ЕОМ, зростає інтенсивність їхніх відмовлень.

Швидкість руху повітря - вектор усередненої швидкості переміщення повітряних потоків під дією різних сил, що спонукують, також впливає на роботу друкувальних пристроїв.

Атмосферний тиск у приміщеннях також нормується. При зниженому тиску повітря погіршується відвід тепла від елементів ЕОМ, знижуються ізоляційні властивості повітря. Перераховане вище обумовлює підвищені

вимоги до кліматичних умов середовища в приміщеннях з ЕОМ.

Оптимальні і припустимі норми мікроклімату:

- температура повітря 18-25 ºС;
- відносна вологість повітря 40-70%;
- швидкість руху повітря не більш 0.3 м/с;
- атмосферний тиск 1013.25 + 266 кпа.

Природне і штучне освітлення в приміщеннях регламентується СНіП 23-05-95 у залежності від характеру зорової роботи, системи і виду освітлення, тла, контрасту об'єкта з тлом. Характеристика зорової роботи визначається найменшим розміром об'єкта розрізнення.

Штучне освітлення нормується кількісними і якісними показниками (показниками засліпленості і дискомфорту, коефіцієнтом пульсації освітленості).

Природне освітлення характеризується тим, що створювана освітленість змінюється в залежності від часу доби, року, метеорологічних умов. Тому як критерій оцінки природного освітлення прийнята відносна величина коефіцієнт природної освітленості.

5.3 Розрахунок системи штучного освітлення

У приміщенні необхідно організувати змішане освітлення, тобто сполучення природного і штучного освітлення. Природне освітлення створюється бічним освітленням через вікно. Штучне освітлення використовується при недостатньому природному освітленні. У даному приміщенні використовується загальне штучне освітлення.

Розрахунок його здійснюється по методу світлового потоку з урахуванням потоку, відбитого від стін і стелі.

Потрібно зробити розрахунок загального освітлення виробничого приміщення. Довжина A = 3,0 м, ширина B = 6,0 м, висота H = 3 м. Висота робочої поверхні h_p = 1 м. Для освітлення використовується світильник УПМ-15. Мінімальна освітленість лампи накалювання по нормах E_{min}=100 лк. Коефіцієнт відображення стелі S_n=50%, стін S_c=30%, робочої поверхні

S_p=10%. Напруга мережі 220 В. $K_з$ = 1,3 та Z = 1,15.

Відстань від стелі до робочої поверхні:

$H_0 = H – h_p = 3 – 1 = 2$ *м*

Відстань від стелі до світильника:

$h_c = 0,2\ H_0 = 0,2 * 2 = 0,4$ *м*

Висота підвісу світильника над освітленою поверхнею:

$h = H_0 – h_c = 2 – 0,4 = 1,6$ *м*

Висота підвісу світильника над підлогою:

$H_n = h + h_p = 1,6 + 1 = 2,6$ *м*

Для досягнення найбільшої рівномірності освітлення приймаємо відношення

$L / h = 1,5.$

Відстань між центрами світильників:

L = 1,5 h = 1,5 * 1,6 = 2,4 м

Необхідна кількість світильників:

$N = S / L^2 = (3 * 6) / (2,4)^2 = 18/5,76 = 3,125$

Приймаємо кількість світильників: 3.

Індекс приміщення:

i = (A B) / (h (A + B)) = (3 * 6) / (1,6 (3 + 6)) = 18 / 14,4 = 1,25

Коефіцієнт використання світлового потоку $\eta = 0,69$

Світловий потік однієї лампи:

$Ф = (E_{min}\ S\ K_З\ Z) / (N\ \eta) = (100 * 18 * 1,3 * 1,15) / (3 * 0,69) = 1300$ *лм*

По найденному световому потоку из таблицы “Параметры ламп накаливания общего назначения с расчётными напряжениями 130 и 220 В” выбираем лампу накаливания мощностью 300 Вт, имеющую световой поток 2090 лм, наиболее близкий к расчётному.

Фактична освітленість:

$E_ф = E_{min}\ Ф_л / Ф_р = 100 * 2090 / 1300 = 160,7$ *лк*

Загальна потужність:

$P_{общ} = P_л N = 300 * 3 = 900 Вт = 0,9 кВт$

5.4 Розробка заходів щодо зменшення впливу шкідливих та небезпечних факторів (щодо захисту від електромагнітного випромінювання)

Під впливом ЕМП та випромінювань спостерігаються загальна слабкість, підвищена втома, пітливість, сонливість, а також розлад сну, головний біль, біль в ділянці серця. З'являється роздратування, втрата уваги, зростає тривалість мовнорухової та зоровомоторної реакцій підвищується межа нюхової чутливості. Виникає ряд симптомів, які є свідченням порушення роботи окремих органів — шлунку, печінки селезінки, підшлункової та інших залоз. Пригнічуються харчовий та статевий рефлекси.

Реєструються зміни артеріального тиску, частота серцевого ритму форма електрокардіограми. Це свідчить про порушення діяльності серцево-судинної системи. Фіксуються зміни показників білкового та вуглеводного обміну, збільшується вміст азоту в крові та сечі, знижується концентрація альбуміну та зростає вміст глобуліну, збільшується кількість лейкоцитів, тромбоцитів, виникають й інші зміни складу крові.

Для зменшення впливу ЕМП на персонал та населення, яке находиться у зоні дії радіоелектронних засобів, потрібно вжити ряд захисних заходів. До їх числа можуть входити організаційні, інженерно-технічні та лікарсько-профілактичні.

Здійснення організаційних та інженерно-технічних заходів покладено передусім на органи санітарного нагляду. Разом з санітарними лабораторіями підприємств та установ, які використовують джерела електромагнітного випромінювання, вони повинні вживати заходів з гігієнічної оцінки нового будівництва та реконструкції об'єктів, котрі виробляють та використовують радіо засоби, а також нових технологічних процесів та обладнання з використанням ЕМП, проводити поточний санітарний нагляд за об'єктами, які використовують джерела випромінювання, здійснювати організаційно-методичну роботу з підготовки спеціалістів та інженерно-технічний нагляд.

Ще на стадії проектування повинне бути забезпечене таке взаємне розташування випромінюючих та опромінюваних об'єктів, яке б зводило до мінімуму інтенсивність опромінення. Оскільки повністю уникнути опромінення неможливо, потрібно зменшити ймовірність проникнення людей у зони з високою інтенсивністю ЕМП, скоротити час перебування під опроміненням. Потужність джерел випромінювання мусить бути мінімально потрібною.

Виключно важливе значення мають інженерно-технічні методи та засоби захисту: колективний (група будинків, район, населений пункт), локальний (окремі будівлі, приміщення) та індивідуальний. Колективний захист спирається на розрахунок поширення радіохвиль в умовах конкретного рельєфу місцевості. Економічно найбільш доцільно використовувати природні екрани — складки місцевості, лісонасадження, нежитлові будівлі. Встановивши антену на горі, можна зменшити інтенсивність поля, яке опромінює населений пункт, у багато разів. Аналогічний результат дає відповідна орієнтація діаграми напрямленості, особливо високо спрямованих антен, наприклад, шляхом збільшення висоти антени. Але висока антена складніша, дорожча, менш стійка. Крім того, ефективність такого захисту зменшується з відстанню.

При захисті від випромінювання екрана повинне враховуватись затухання хвилі при проходженні через екран (наприклад, через лісову смугу). Для екранування можна використовувати рослинність. Спеціальні екрани у вигляді відбивальних і радіопоглинальних щитів дорогі, малоефективні і використовуються дуже рідко.

Локальний захист дуже ефективний і використовується часто. Він базується на використанні радіозахисних матеріалів, які забезпечують високе поглинання енергії випромінювання у матеріалі та віддзеркалення від його поверхні. Для екранування шляхом віддзеркалення використовують металеві листи та сітки з доброю провідністю. Захист; приміщень від зовнішніх випромінювань можна здійснити завдяки обклеюванню стін металізованими

шпалерами, захисту вікон сітками, металізованими шторами. Опромінення у такому приміщенні зводиться до мінімуму, але віддзеркалене від екранів випромінювання розповсюджується у просторі та потрапляє на інші об'єкти.

До інженерно-технічних засобів захисту також належать:

1) конструктивна можливість працювати на зниженій потужності v процесі налагоджування, регулювання та профілактики;

2) робота на еквівалент налагоджування;

3) дистанційне керування.

Радіопоглинальні матеріали можуть використовуватися для захисту навколишнього середовища від ЕМП, яке генерується джерелом, що Знаходиться в екранованому об'єкті. Крім того, радіопоглиначами для Захисту від віддзеркалення личкуються стіни безлунких камер — приміщень, де випробовуються випромінювальні пристрої. Радіопоглинаючі матеріали використовуються в кінцевих навантаженнях, еквівалентах системах.

РОЗДІЛ 6. ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

6.1 Загальні аспекти впливу інформаційних технологій на навколишнє середовище

Розвиток інформаційних технологій значно вплинув на різні сфери діяльності людини, зокрема на обробку, зберігання та передачу інформації. Використання програмних систем, зокрема систем збирання та систематизації даних з веб-сайтів, дозволяє автоматизувати багато процесів, зменшити використання паперових носіїв та підвищити ефективність роботи з інформацією.

Разом із тим, використання комп’ютерної техніки та серверної інфраструктури потребує електричної енергії, що може мати певний вплив на навколишнє середовище. Основними екологічними аспектами використання інформаційних систем є споживання електроенергії, утворення електронних відходів та використання природних ресурсів під час виробництва комп’ютерного обладнання.

Впровадження сучасних інформаційних технологій повинно здійснюватися з урахуванням принципів енергоефективності та раціонального використання ресурсів.

6.2 Енергоефективність використання комп’ютерної техніки

Під час розробки та експлуатації програмної системи для збирання та систематизації даних з веб-сайтів використовується комп’ютерна техніка та мережеве обладнання. Для зменшення негативного впливу на навколишнє середовище доцільно застосовувати енергоефективні пристрої та програмні рішення.

Основними заходами підвищення енергоефективності є:

- використання сучасних енергоефективних комп’ютерів та серверів;
- застосування режимів енергозбереження;
- оптимізація програмного коду для зменшення навантаження на процесор;

- використання віртуалізації та хмарних технологій;
- автоматичне вимкнення обладнання у періоди простою.

Оптимізація програмних алгоритмів збору та обробки даних дозволяє зменшити обчислювальні витрати та скоротити споживання електроенергії.

6.3 Зменшення використання паперових ресурсів

Однією з переваг використання інформаційних систем є можливість зменшення використання паперових документів. Розроблена система дозволяє зберігати інформацію у цифровому вигляді, що зменшує потребу в друку документів.

Використання електронних баз даних, електронних звітів та автоматизованих систем обробки інформації сприяє збереженню природних ресурсів, зокрема деревини, яка використовується для виробництва паперу. Крім того, зменшення використання паперу сприяє зниженню витрат на друк та транспортування документів.

6.4 Утилізація електронних відходів

Важливою проблемою сучасного суспільства є накопичення електронних відходів. Комп’ютери, сервери, периферійні пристрої та інше обладнання після завершення терміну експлуатації потребують правильної утилізації.

Для мінімізації негативного впливу електронних відходів на навколишнє середовище необхідно дотримуватися таких заходів:

- передача відпрацьованого обладнання спеціалізованим підприємствам з утилізації;
- повторне використання придатних компонентів;
- модернізація існуючого обладнання замість його повної заміни;
- використання екологічно сертифікованої техніки.

Раціональне поводження з електронними відходами дозволяє зменшити забруднення довкілля важкими металами та іншими шкідливими речовинами.

6.5 Використання екологічно орієнтованих інформаційних технологій

Сучасні інформаційні технології активно впроваджують принципи «зелених» обчислень (Green IT). Цей підхід передбачає створення та використання програмних і апаратних засобів, що забезпечують мінімальне споживання енергії та зменшення негативного впливу на довкілля.

До основних принципів Green IT належать:

- оптимізація програмного забезпечення;
- використання енергоефективних дата-центрів;
- застосування хмарних технологій;
- зменшення кількості фізичних серверів за рахунок віртуалізації;
- раціональне використання ресурсів комп'ютерної техніки.

Розробка системи збирання та систематизації даних з веб-сайтів може бути реалізована з урахуванням цих принципів, що дозволить підвищити її екологічну ефективність.

Висновки до розділу

У результаті аналізу екологічних аспектів використання інформаційних технологій встановлено, що розробка та використання системи збирання та систематизації даних з веб-сайтів не має значного негативного впливу на навколишнє середовище за умови раціонального використання комп'ютерної техніки та енергетичних ресурсів.

Використання енергоефективного обладнання, оптимізація програмного забезпечення, зменшення використання паперових носіїв та правильна утилізація електронних відходів сприяють мінімізації екологічного впливу інформаційних систем. Таким чином, впровадження сучасних інформаційних технологій може поєднувати високу ефективність роботи з інформацією з дотриманням принципів охорони навколишнього середовища.

ВИСНОВКИ

В результаті виконання даної дипломної роботи була розроблена інформаційна система збирання та систематизації даних з веб-сайтів. Для цього були виконані аналіз предметної області та постановка задачі, розроблено технічне завдання та проектні рішення інформаційної системи, які включають:

- загальносистемні рішення;
- рішення з інформаційного забезпечення;
- рішення з математичного забезпечення;
- рішення з організаційного забезпечення;
- рішення з програмного забезпечення;
- рішення з технічного забезпечення.

У рамках цих проектних рішень були розроблені програмне забезпечення для ефективного збирання даних з веб-сайтів і технічні засоби для автоматизації інформаційних потоків системи управління підприємства, а також технічна документація, заявлена у технічному завданні.

Крім того було виконано:

- розрахунок економічної ефективності впровадження інформаційної системи;
- дослідження з охорони праці та охорони навколишнього середовища.

З урахуванням усього вищенаведеного можна сказати, що цілі, поставлені на початку даної дипломної роботи були повністю виконані.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Kumar R., Raghavan P., Rajagopalan S., Sivakumar D., Tomkins A., Upfal E. The Web as a graph. In Proceedings of the 19th Symposium on Principles of Database Systems (PODS). ACM Press. 2000. P. 1-10. https://www.researchgate.net/publication/221559387_The_Web_as_a_Graph
2. Ling Liu and Özsu Tamer M. Ontology to appear in the Encyclopedia of Database Systems [Электронный ресурс]. Springer-Verlag. 2008. URL: http://tomgruber.org/writing/ontology-inencyclopedia-of-dbs.pdf (дата звернення: 06.10.2020)
3. Каунг Мьят Хту Анализ языка Веб онтологии (owl) и семантическая веб-технология. Auditorium. 2017. №4 (16). URL: https://cyberleninka.ru/article/n/analiz-yazyka-veb-ontologiiowl-i-semanticheskaya-veb-tehnologiya (дата звернення: 06.10.2020)
4. Ольшевский А.И., Кондратьева А.А. Описание способов представления web-сайтов в виде фреймовой модели для реализации функциональных операций в Интернет-клиентских системах. Искусственный интеллект. 2008. №1. С. 110–116. http://dspace.nbuv.gov.ua/handle/123456789/6551
5. Bing L. Web Data Mining. Springer. 2011. 624 p. https://www.springer.com/gp/book/9783642194597
6. Sheu P., Yu H., Ramamoorthy C., Joshi A., Zadeh L. Link Analysis in Web Mining: Techniques and Applications. Semantic Computing Digital Object Identifier. 2010. P. 69–86. https://www.researchgate.net/publication/229459726_Link_Analysis_in_Web_Mining_Techniques _and_Applications
7. Lai Wei, Huang Xiaodi From graph data extraction to graph layout: Web information visualization. Information Sciences and Interaction Sciences (ICIS), 2010 3rd International Conference on Digital Object Identifier. 2010. P. 224–229. https://bit.ly/3omK2Qa
8. Hall W., Tiropanis T. Web evolution and Web Science. Computer Networks. 2012. Vol. 56. № 18. P. 3859–3865. https://bit.ly/3qB6NBG
9. Saul Schleimer, Daniel S. Wilkerson, Alex Aiken. Winnowing: Local Algorithms for Document Fingerprinting. 2003, 10 с. URL: https://theory.stanford.edu/~aiken/publications/papers/sigmod03.pdf.
10. Ryan Mitchell. Web Scraping With Python: Collecting More Data From The Modern Web. O'Reilly, 2018, 308 с.
11. David Beazley, Brian K. Jones. Python Cookbook, Third Edition. O'Reilly, 2013, 706 с.
12. Data scraping [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/Data_scraping
13. Web scraping [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/Web_scraping 3.
14. Scraping wiki [Електронний ресурс]. – Режим доступу: https://www.scrapesentry.com/scraping-wiki/ 4. 10 Web Scraping Tools to

Extract Online Data [Електронний ресурс]. – Режим доступу: http://www.hongkiat.com/blog/web-scraping-tools/

ДОДАТОК А - ТЕХНІЧНЕ ЗАВДАННЯ

А.1 Загальні відомості

Система розрахована на людей зі всіх куточків країни і світу. що бажають шляхом отримання адекватної, докладної, нової інформації та віддалено замовити послуги. А також програма розрахована на користування нею адміністраторів пунктів постачання послуг.

А.2 Призначення і цілі

Інформаційна система призначена для користування нею жителів та туристів і постачальників послуг.

Головною ціллю створення ІС є підвищення інформативності, а отже рівня послуг та прибутку міста.

Побічною ціллю розробки ІС є написання і захист дипломного проекту.

А.3 Вимоги до інформаційної системи

Вимоги до функціональних характеристик

Дана ІС повинна надавати можливість віддалено:

- дізнатись інформацію про місця проживання, ціни, рівень комфорту;
- дізнатись інформацію про транспорт, ціни, розклад маршрутів, зміни;
- дізнатись інформацію про місця харчування, ціни, меню;
- зареєструватись постачальникам послуг;
- постачальникам послуг викласти інформацію про вільні місця;
- бронювати послуги (проживання, транспорт, харчування);
- відпочиваючим обмінюватись відгуками;
- дізнатись історію міста, переглянути зручну карту, фото, статті, новини місцевості;
- переглядати інформацію про погоду.

Вимоги до надійності :

1) ІС повинна безвідмовно працювати. Допустимим вважається середній % працездатності системи за місяць не менше 99%.

2) ІС не повинна втрачати дані-середній % втрати пакетів за місяць повинен бути не більше 0,2%.

3) ІС повинна швидко обробляти інформацію. Допустимим на даному етапі функціонування системи вважається обробка до 100 заявок за хвилину.

А.4 Склад та зміст робіт по створенню системи

Створення ІС складається з визначення вимог, проектування, написання коду програми, проведення тестування. Всі моделі програми повинні на 100% бути покриті тестами білого ящика- Unit test.

Також існує необхідність залучення до взаємодії адміністраторів послуг.

А.5 Порядок контролю і прийому робіт

Контролюється робота випробуваннями програми. Випробування проводяться відповідно до програми і методики випробувань. Приймає роботу керівник дипломного проекту Робота виконується з року дата здачі роботи року. У випадку виявлення помилок або невідповідності технічному завданню, протягом семи днів проводиться випробування.

А.6 Вимоги до змісту і складу робіт по підготовці об’єкта автоматизації до вводу системи в дію

Вимоги до змісту і складу робіт по підготовці об’єкта автоматизації до вводу системи в дію представимо у вигляді таблиці А.1.

Таблиця А.1- Вимоги до змісту і складу робіт

	Посадова особа	Робота	Строк здачі
	розробник ІС	визначити вимоги до системи	01.02.23
		розробити проектні рішення системи	04.03.23

		закодувати програму	30.04.23
		провести тестування	15.05.23
		розробити робочу документацію	01.06.23
	адміністратор ІС	залучити адміністраторів послуг	10.06.23
		викласти ознайомлювальну інформацію про місто(фото, історичні статті, опис)	15.06.23
	адміністратор готельного комплексу	зареєструватись, викласти докладну інформацію про свою організацію(контактну інформацію,режим роботи, ціни, наявність, рівень комфорту і т.ін.)	25.06.23
	адміністратор комплексу харчування	зареєструватись, викласти докладну інформацію про свою організацію(контактну інформацію, режим роботи, меню, ціни, і т.ін.)	30.06.23
	адміністратор транспорту	зареєструватись, викласти розклад маршрутів,ціни	30.06.23

А.7Вимоги до документації

До програмної документації мають бути включені документи:

1) опис програми;

2) інструкція користувачам (адміністраторам у вигляді документу, а туристам та жителям у вигляді web-сторінки в ІС);

3) програма і методика випробувань.

ДОДАТОК Б - ТЕКСТ ПРОГРАМИ

```
class AdminhotelsController < ApplicationController
  # GET /adminhotels
  # GET /adminhotels.xml
  def index
    @adminhotels = Adminhotel.find(:all)

    respond_to do |format|
      format.html # index.html.erb
      format.xml  { render :xml => @adminhotels }
    end
  end

  # GET /adminhotels/1
  # GET /adminhotels/1.xml
  def show
    @adminhotel = Adminhotel.find(params[:id])

    respond_to do |format|
      format.html # show.html.erb
      format.xml  { render :xml => @adminhotel }
    end
  end

  # GET /adminhotels/new
  # GET /adminhotels/new.xml
  def new
    @adminhotel = Adminhotel.new
```

```
    respond_to do |format|
      format.html # new.html.erb
      format.xml  { render :xml => @adminhotel }
    end
  end

  # GET /adminhotels/1/edit
  def edit
    @adminhotel = Adminhotel.find(params[:id])
  end

  #GET /adminhotels
  #GET /adminhotels.xml
  def to_new_hotel
     format.html { render :action => "to_new_hotel" }
  end

  # POST /adminhotels
  # POST /adminhotels.xml
  def create
    @adminhotel = Adminhotel.new(params[:adminhotel])

    respond_to do |format|
      if @adminhotel.save
        flash[:notice] = 'Adminhotel was successfully created.'
        format.html { redirect_to(@adminhotel) }
        format.xml     { render :xml => @adminhotel, :status => :created,
:location=> @adminhotel }
      else
        format.html { render :action => "new" }
```

```
        format.xml  { render :xml => @adminhotel.errors, :status =>
:unprocessable_entity }
      end
    end
  end

  # PUT /adminhotels/1
  # PUT /adminhotels/1.xml
  def update
    @adminhotel = Adminhotel.find(params[:id])

    respond_to do |format|
      if @adminhotel.update_attributes(params[:adminhotel])
        flash[:notice] = 'Adminhotel was successfully updated.'
        format.html { redirect_to(@adminhotel) }
        format.xml  { head :ok }
      else
        format.html { render :action => "555" }
        format.xml  { render :xml => @adminhotel.errors, :status =>
:unprocessable_entity }
      end
    end
  end

  # DELETE /adminhotels/1
  # DELETE /adminhotels/1.xml
  def destroy
    @adminhotel = Adminhotel.find(params[:id])
    @adminhotel.destroy
```

```
    respond_to do |format|
      format.html { redirect_to(adminhotels_url) }
      format.xml  { head :ok }
    end
  end
end
```

Контролер для туриста

```
class TrevelersController < ApplicationController
  # GET /trevelers
  # GET /trevelers.xml
  def index
    @trevelers = Treveler.find(:all)

    respond_to do |format|
      format.html # index.html.erb
      format.xml  { render :xml => @trevelers }
    end
  end

  # GET /trevelers/1
  # GET /trevelers/1.xml
  def show
    @treveler = Treveler.find(params[:id])

    respond_to do |format|
      format.html # show.html.erb
      format.xml  { render :xml => @treveler }
    end
  end
```

```
  # GET /trevelers/new
  # GET /trevelers/new.xml
  def new
    @user.resource=Treveler.new
    #@treveler = Treveler.new

    respond_to do |format|
      format.html # new.html.erb
      format.xml  { render :xml => @treveler }
    end
  end

  # GET /trevelers/1/edit
  def edit
    @treveler = Treveler.find(params[:id])
  end

  # POST /trevelers
  # POST /trevelers.xml
  def create
    @treveler = Treveler.new(params[:treveler])

    respond_to do |format|
      if @treveler.save
        flash[:notice] = 'Treveler was successfully created.'
        format.html { redirect_to(@treveler) }
        format.xml  { render :xml => @treveler, :status => :created, :location =>
@treveler }
      else
```

```
        format.html { render :action => "new" }
        format.xml  { render :xml => @treveler.errors, :status =>
:unprocessable_entity }
      end
    end
  end

  # PUT /trevelers/1
  # PUT /trevelers/1.xml
  def update
    @treveler = Treveler.find(params[:id])

    respond_to do |format|
      if @treveler.update_attributes(params[:treveler])
        flash[:notice] = 'Treveler was successfully updated.'
        format.html { redirect_to(@treveler) }
        format.xml  { head :ok }
      else
        format.html { render :action => "edit" }
        format.xml  { render :xml => @treveler.errors, :status =>
:unprocessable_entity }
      end
    end
  end

  # DELETE /trevelers/1
  # DELETE /trevelers/1.xml
  def destroy
    @treveler = Treveler.find(params[:id])
    @treveler.destroy
```

```
    respond_to do |format|
      format.html { redirect_to(trevelers_url) }
      format.xml  { head :ok }
    end
  end
end
```

Модель для користувача

```
require 'digest/sha1'

class User < ActiveRecord::Base
  include Authentication
  include Authentication::ByPassword
  include Authentication::ByCookieToken

  belongs_to :resource, :polymorphic=> true

  validates_presence_of     :login
  validates_length_of       :login,    :within => 3..40
  validates_uniqueness_of   :login
  validates_format_of       :login,       :with => Authentication.login_regex, :message => Authentication.bad_login_message

  validates_format_of       :name,       :with => Authentication.name_regex, :message => Authentication.bad_name_message, :allow_nil => true
  validates_length_of       :name,     :maximum => 100

  validates_presence_of     :email
```

```
    validates_length_of       :email,    :within => 6..100 #r@a.wk
    validates_uniqueness_of   :email
    validates_format_of       :email,    :with => Authentication.email_regex, :message => Authentication.bad_email_message

    attr_accessible :login, :email, :name, :password, :password_confirmation


    # Authenticates a user by their login name and unencrypted password. Returns the user or nil.
    #
    # uff.  this is really an authorization, not authentication routine.
    # We really need a Dispatch Chain here or something.
    # This will also let us return a human error message.
    #
    def self.authenticate(login, password)
      return nil if login.blank? || password.blank?
      u = find_by_login(login.downcase) # need to get the salt
      u && u.authenticated?(password) ? u : nil
    end
    def login=(value)
      write_attribute :login, (value ? value.downcase : nil)
    end

    def email=(value)
      write_attribute :email, (value ? value.downcase : nil)
    end
    protected
  end
```

Вид для туриста

```
<h1>New treveler</h1>

<% form_for(@user.resource) do |f| %>
  <%= f.error_messages %>

      <p>
          <label for='firstName'>firstName</label>
          <%= f.text_field :firstName  %>
      </p>
      <p>
          <label for ='secondName'>secondName</label>
          <%= f.text_field :secondName %>
      </p>
      <p>
          <label for='surname'>surname</label>
          <%= f.text_field :surname %>
      </p>
      <p>
          <label for='login'>login</label>
          <%=f.text_field :login %>
      </p>
      <p>
              <label for='name'>name</label>
              <%=f.text_field :name %>
      </p>
      <p>
              <label for='email'>email</label>
              <%=f.text_field :email %>
      </p>
```

```
            <p>
                    <label for='password'>password</label>
                    <%=f.text_field :password %>
            </p>
            <p>
                    <label
for='password_confirmation'>password_confirmation</label>
                    <%=f.text_field :password_confirmation %>
            </p>
            <p>
              <%= f.submit "Create" %>
            </p>
      <% end %>

      <%= link_to 'Back', trevelers_path %>
```

ДОДАТОК В - ОПИС СИСТЕМИ

В.1 Загальні відомості

Розроблено інформаційну систему- «**Дослідження технологій збирання та систематизації Даних з веб-сайтів**».

Програмне забезпечення написано сучасною об'єктно-орієнтованою інтерпретовоною мовою програмування Ruby з використанням Agile framework Ruby on Rails (ROR).

Для функціонування інформаційної системи необхідно, щоб було установлено на сервері: ruby version 1.8.7, framework RAILS VERSION 2.3.3, сервер баз даних MySql 5.0.67; http- сервер Apache 2, http- proxy Mongrel; на клієнті: web- браузер(Mozilla Firefox, A Web browser, Opera, Opera mini). Завдяки багатоплатформеності Ruby on Rails, цим програмним продуктом продуктом можна користуватися як в операційній системі Linux, так і в Windows.

В.2 Функціональне призначення

Інформаційна система призначена для користування нею жителів та туристів і постачальників послуг.

Функціональне призначення ІС надавати можливість віддалено:

- дізнатись інформацію про місця проживання, ціни, рівень комфорту;
- дізнатись інформацію про транспорт, ціни, розклад маршрутів, зміни;
- дізнатись інформацію про місця харчування, ціни, меню;
- зареєструватись постачальникам послуг;
- постачальникам послуг викласти інформацію про вільні місця;
- бронювати послуги (проживання, транспорт, харчування);
- відпочиваючим обмінюватись відгуками;
- дізнатись історію міста, переглянути зручну карту, фото, статті, новини місцевості;

– переглядати інформацію про погоду .

В.3 Опис логічної структури

Схеми алгоритмів ІС представлено у розділі 2.5.3- Алгоритм рішення.

ІС є програмним інтерфейсом. Завдяки використанню Rest (архітектура мережевих протоколів, що забезпечують доступ до інформаційних ресурсів), ІС може інтегруватись з програмами постачальників курортних послуг.

Структура програмного забезпечення складається класів. Класи в свою чергу діляться на класи моделі, класи контролери, класи види (архітектура MVC (Model-View-Controller)).

Класи моделі:

User, Treveler, AdministratorTransportComplex, AdministratorCafe, AdministratorHotel, Service, Hotel, Room, Cafe, Table, TransportCompex, Vechicle, ReservationHotel, ReservationTransport, ReservationCafe.

Класи контролери:

SessionsController, ServicesController, Home1Controller, UsersController, HotelsController, CafesController, TransportsComplexController, AdministratorHotelsController, AdministratorCafesController, AdministratorTransportComplexsController, RoomsController TablesController, VehiclesController, ReservationRoomsController, ReservationTablesController, ReservationVehiclesController, ApplicationController, ActionController, AuthenticateSystemController.

В. 4 Виклик і завантаження

Клієнти викликають програму віддалено з web - браузерів, установлених на їх комп'ютерах.

ДОДАТОК Г - ПОСІБНИК КОРИСТУВАЧА

Галузь застосування – місто, де є готельний, транспортний і комплекс харчування.

Дана ІС надає можливість:

- переглядати фото місцевості, історію міста;
- переглядати карту місцевості;
- переглядати інформацію про погоду;
- переглядати інформацію про місця проживання;
- переглядати інформацію про транспорт;
- переглядати інформацію про місця харчування;
- надавати можливість бронювати послуги(проживання, транспорт, харчування);
- надавати можливість за реєструватися постачальникам послуг;
- надавати можливість відпочиваючим обмінюватись відгуками.

Рівень підготовки користувача може бути початковим.

Користувачеві- туристу потрібно мати елементарні навички користування інтернет-додатками, електронною поштою.

Користувач- адміністратор має вміти користуватись програмами створення звітів, прийому замовлень.

Користувачу не потрібно турбуватись про порядок завантаження програм. Тому що йому не потрібно завантажувати на свій комп'ютер програми.

Завдяки ІС, яка основана на моделі SaaS адміністратор курортного комплексу звільняється від рутинної процедури прийому замовлення. Йому не треба піклуватись за збереження бази даних. База даних замовлень його організації зберігається на хостингу системи. Багато клієнтів віддалено отримують дану інформаційну систему як сервіс. Замовники платять не за володіння ІС як такою, а за її використання.