

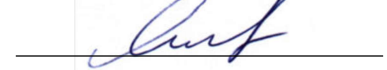
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра інформаційних управляючих систем та технологій

"Допущений до захисту"

Завідувач кафедри ІУСТ



к.т.н, доц. Михелєв І.Л.

" ____ " _____ 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти "магістр"

на тему:

Системний аналіз IoT-засобів для розробки системи мікроклімату «Розумний дім»

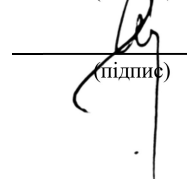
Виконав: студент групи 6145 м



Абрамян А.А.

Керівник роботи:

доцент кафедри ІУСТ, к.т.н.,
(посада, науковий ступінь вчене звання)



Гайда А.Ю..

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами

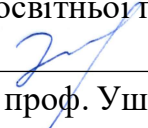
Кафедра інформаційних управляючих систем та технологій

Спеціальність 124 "Системний аналіз"

Освітня програма "Системний аналіз"

"ЗАТВЕРДЖУЮ"

Гарант освітньої програми


д.ф-м.н, проф. Ушкац М.В.

"___" _____ 2024 р.

**ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ
на здобуття ступеня вищої освіти "магістр"**

Студенту Абрамян Артур Арамович
(прізвище, ім'я, по батькові)

1. Тема роботи: Системний аналіз IoT-засобів для розробки системи мікроклімату «Розумний дім»

Керівник роботи _____

Затверджені наказом ректора № 1055-уч. від "08" жовтня 2024 року.

2. Термін подання роботи: 15 грудня 2024 р.

3. Вихідні дані до проекту (роботи) ДСТУ щодо обробки інформації, літературні джерела, технічна документація на існуючі аналоги систем, матеріали наукового стажування

4. Перелік питань, що належать до розробки (найменування розділів)

Аналіз існуючих систем розумного дому, аналіз та вибір засобів розробки, Розробка проектних рішень по системі, Розробка робочої документації системи «розумний дім» з віддаленим доступом, Розрахунок економічного ефекту від впровадження ІС, Охорона праці, Охорона навколишнього середовища

5. Перелік презентаційних матеріалів _____

Аналіз існуючих рішень, Структура житлового будинку, Схема організаційно-функціональної структури, Триланкова архітектура інформаційної системи, Схема розміщення технічних засобів, Діаграма класів системи, Загальний алгоритм поведінки системи, Фізична модель БД системи, Знімки екрану при роботі підсистеми

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Гайда А.Ю.	05.09.2024	10.12.2024
2	Гайда А.Ю.	05.09.2024	10.12.2024
3	Гайда А.Ю.	05.09.2024	10.12.2024
4	Гайда А.Ю.	05.09.2024	14.11.2024
5	Гайда А.Ю.	05.09.2024	14.11.2024
6	Гайда А.Ю.	05.09.2024	14.11.2024

7. Дата видачі завдання 05 вересня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів роботи	Примітка
1	Дослідження та аналіз предметної галузі		
2	Розробка концепції		
3	Розділ з розробки проектних рішень по системі (та її частинах)		
4	Реалізація проекту та розробка робочої документації		
5	Обчислення економічної ефективності		
6	Вирішення питань охорони праці та навколишнього середовища		
7	Оформлення пояснювальної записки		
8	Подання магістерської роботи на перевірку на плагіат		
9	Подання магістерської роботи рецензенту		
10	Подання магістерської роботи на державний захист		

Студент


(підпис)

Абрамян А.А.
(ПІБ)

Керівник роботи


(підпис)

Гайда А.Ю.
(ПІБ)

Анотація

Представлена робота присвячена розробленню інтернет технологій для створення системи "Розумний дім" з віддаленим доступом з метою поліпшення умов життя.

Під час роботи аналізуються існуючі рішення, формуються вимоги до системи, розроблюються концепції системи і проводиться вибір найкращої, розробляється технічне завдання. Далі проводиться розробка основних проектних рішень за технічним, інформаційним та програмним забезпеченням, формується робоча документація за програмним забезпеченням.

Дипломна робота виконана на _____ аркушах машинописного тексту, містить _____ малюнків, _____ таблиць, _____ додатків.

Ключові слова: Розумний дім, Інтернет речей, мікроклімат, Java, сенсори

Abstract

The present work is devoted to development of the project of the information system microclimate in Smart house management system with the purpose to improve quality of life.

During operation the requirements to the system are formed, the concepts of the system are developed and the choice by best is made, the requirement specification is developed. Development of main design solutions on technical, information and software further will be carried out, working documentation under the software is formed.

The degree operation is stated on _____ sheets of the typewritten text, contains _____ figures, _____ tables, _____ application

Keywords: Smart house, IoT, microclimate, Java, sensors

ЗМІСТ

Вступ

1 Формування вимог до системи керування мікрокліматом, розробка концепції та постановка задачі

1.1 Формування вимог до інформаційної системи

1.2 Розробка концепції інформаційної системи

1.3 Постановка задачі

2 Розробка проектних рішень по системі

2.1 Загальносистемні рішення

2.2 Рішення з організаційного забезпечення

2.3 Рішення з інформаційного забезпечення

2.4 Рішення з технічного забезпечення

2.5 Рішення з математичного забезпечення. Опис алгоритму

2.6 Рішення з програмного забезпечення. Опис програмного забезпечення

3 Розробка робочої документації

4 Охорона праці

Висновки

Список використаних джерел

Додатки

Додаток А Технічне завдання

Додаток Б Програма і методика випробувань

Додаток В Керівництво користувача

Додаток Г Текст програмних модулів

Вступ

Будь-який будинок - будь-то адміністративний, виробничий або житловий складається з деякого набору підсистем, що відповідають за виконання певних функцій, які вирішують різні завдання в процесі функціонування цієї будівлі. В міру ускладнення цих підсистем і збільшення кількості, виконуваних ними функцій, управління ними ставало все складніше. Також стрімко зростають витрати на утримання обслуговуючого персоналу, ремонт і обслуговування цих підсистем. Вперше ці проблеми постали при експлуатації великих адміністративних і виробничих комплексів.

Сучасна будівля такого типу - це місто в мініатюрі. Фактично в ньому діють всі служби, які були раніше неодмінними атрибутами міського господарства. У таких будівлях зазвичай існує адміністративна служба або адміністратор, які використовують і обслуговують цю систему практично цілодобово. Хоча є чимало засобів автоматики, які самі справляються з покладеними на них завданнями, такими, як опалення, вентиляція, підтримка мікроклімату, освітлення, пожежна сигналізація, димознищення, контроль входу / виходу і т.п., але управління і обслуговування всіх цих систем вимагає наявності адмініструючого персоналу.

Його обов'язком є контроль роботи цих підсистем і вжиття заходів у разі виходу їх з ладу. Але є ситуації, коли навіть дії кваліфікованого персоналу можуть виявитися неефективними. Це випадки виникнення загрози будівлі і знаходяться в ньому людям, що мають глобальний характер - пожежа, землетрус і інші стихійні лиха, терористичні атаки. Тут потрібно вживати екстраординарних заходів в лічені частки секунди. Реакція і коректність дій людей в критичній ситуації може виявитися недостатньою.

Традиційні системи забезпечення різних аспектів життєдіяльності в минулому проектувалися як автономні. Такі системи, що створювалися окремо для кожної функції і об'єднані для довільної частини будівлі. У будівлях встановлювалися системи тільки з тими можливостями і з тим ступенем складності, які були необхідні на поточний момент побудови будівлі. Подальше розширення і модернізація даних систем були складними і дорогими завданнями через безліч різних чинників. Витрати на експлуатацію такої системи складаються з витрат на експлуатацію кожної

автономної системи в окремо, вартості навчання персоналу Вартість експлуатації цих систем висока - в силу їх автономності кожна з них підтримується окремо. Вартість навчання користувачів настільки ж висока, оскільки вони повинні бути ознайомлені з експлуатацією кожної автономної системи.

1 Аналіз існуючих рішень та формування вимог до інформаційної системи, розробка її концепції і постановка задачі

1.1 Аналіз існуючих рішень

Існує два варіанти перетворення свого житла в розумний дім. Варіант перший - самостійне створення проекту, установка, об'єднання і настройка певних датчиків в один комплекс. Варіант другий - придбати обладнання на основі готових рішень.

Системи розумного дому можуть бути дротові та бездротові. Виділяють серед систем розумного дому централізовані та децентралізовані. У деяких системах розумного будинку використовується відкритий протокол, а в деяких закритий.

Переваги дротових систем у тому, що вони мають надійний зв'язок з компонентами своєї системи. Бездротові легше встановлювати. Централізовані системи управляються з єдиного блоку, який називається базою. А децентралізовані не мають центральної бази. Системи розумного будинку ґрунтовані на відкритих протоколах мають можливість розширення компонентами, які підтримують сумісний протокол, але вироблені іншим виробником.

1.1.1 BroadLink

Розумний будинок BroadLink - в представленому списку має найнижчий ціник, але від того не стає гірше. Навпаки, має деякі переваги. Система являє собою безпосередньо блок управління і набір різноманітних датчиків, що підключаються по мережі Wi-Fi. Серед об'єктів управління Broadlink присутні: контроль температури, мікроклімату, музичного обладнання. Відмінною особливістю є управління всіма параметрами через додаток для смартфона.

Переваги:

- Додаток для управління є для різних операційних систем смартфона.
- Зручність додавання нових пристроїв в мережу.
- Зручний і зрозумілий функціонал.

Недоліки:

- Не велика кількість одночасно працюючих пристроїв.
- Відсутні багато функцій в управлінні. Такі, як відеоконтроль.

1.1.2 Smart Bus

Розумний будинок Smart Bus - дана технологія створювалася і розроблялася в Канаді. Сьогодні її реалізували на великій кількості об'єктів в різних країнах. Відрізняється високотехнологічним обладнання і порівняно низькою ціною. В даному ціновому діапазоні тільки ця технологія має розподілений інтелект.

Переваги:

- Дизайн. Оздоблювальними матеріалами для панелей і датчиків служать скло і метал.
- Немає необхідності в центральному процесорі, так як застосовується технологія розподіленого інтелекту.
- Швидко встановлюється і налаштовується.
- Має встановлені режими настройки будинку: робота, гості, відпустка, ніч.

Недоліки:

- В одній мережі одночасно може бути до 64 приладів.

1.1.3 C-Bus

Розумний будинок C-Bus - це преміум рішення від великого виробника з світовим ім'ям. «На борту» у цій системи є відмінне устаткування для керування світловими приладами. Серед всіх користувачів системи виділяються футбольні стадіони, хмарочоси світових корпорація і особисті будинку популярних персон. Це обумовлено високою технологічністю і величезними можливостями. У дорогих система контролю, таких як C-Bus дизайн і програмне забезпечення мають вже більш високу вартість.

Переваги:

- Сумісність. Виробник розробляє не тільки ПО, а й обладнання, завдяки чому виключаються проблеми сумісності.
- Можливість управління необмеженою кількістю пристроїв.
- Є можливо налаштувати сценарну освітлення будівель.
- Наявність систем економії, таких як економія електроенергії, економія води та газу.
- Простота інтеграції. C-Bus без проблем інтегрується у вже готові системи управління або адміністрування будівель.

Недоліки:

- Самостійно встановити таку систему вже навряд чи вийде. Зважаючи на складнощі.

- Дороге обслуговування.
- Висока ціна на обладнання.

1.1.4 Crestron

Розумний будинок Crestron - це абсолютно новий рівень. Якщо цю систему встановлюють і в звичайний житловий будинок, то розгорнуться на повну і показати все спектр своїх можливостей вона може на тільки найдорожчих і вимогливих об'єктах. Система є дуже гнучкою, що дозволяє завжди знайти саме ту конфігурацію, яка повністю влаштує. Crestron – має великий набір інтегрованих можливостей. Висока вартість виправдана наданими можливостями системи.

Переваги:

- В одну систему можна об'єднати різні системи управління будинком.
- Функціонал дисплеїв управління.
- Інтерфейс користувача може бути дуже великим (сенсорні панелі, кнопкові панелі, пульти).
- Управління зовнішніми об'єктами в оточенні будинку (системи безпеки, освітлення по периметру, відкриття і закриття воріт).

Недоліки:

- Вкрай висока ціна.
- Складність в обслуговуванні. У разі виходу з ладу однієї або декількох систем, необхідно залучати фахівця для вирішення.

1.1.5 Порівняння існуючих рішень

Таблиця 1.1 – Порівняння існуючих систем розумного дому.

Назва	Ціна	Функціональність	Складність встановлення	Надійність
BroadLink	10	6	4	10
Smart Bus	7	6	10	8
C-Bus	5	8	5	7
Crestron	3	8	3	7

Таким чином, серед всіх представлених проектів розумних будинків є, як і не дуже дорогі, але з широким базовим функціоналом, так і системи з величезною кількістю можливостей по контролю багатьох параметрів будівлі. Такі системи відрізняються високою вартістю встановлення і налаштування.

Тому є доцільним розглянути варіант створення індивідуального рішення системи "Розумний дім" з віддаленим доступом.

1.2 Опис предметної області

Результатом розроблення інтернет технологій для створення системи "Розумний дім" з віддаленим доступом є інформаційна система, яка повинна допомагати автоматизувати щоденні задачі, наприклад, вимкнення світла, керування мікрокліматом, керування інтенсивністю освітлення.

Управляюча система є середовищем, складовими елементами якого є мікроконтролери, під'єднані до комп'ютерної мережі, сервер, база 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.

Система, 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):

- 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. 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 (MariaDB, MySQL, PostgreSQL, Oracle та 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.

1.2.1 Технології рівня користувача

Web-оглядач і Web-інтерфейс

Web-оглядач чи Web-браузер – 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, Mozilla FireFox, Opera, Google Chrome.

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 формувати 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 форматowanego 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 кожному 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 утворюють 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).

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. Останнім часом йде активне впровадження CSS Level 3. Але, на жаль не всі браузеры його коректно підтримують. Тому далі розглядатимемо 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.

1.2.2 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 пароллю, яку можна використовувати для обмеження доступу до ресурсів сервера.

Найбільш розповсюджені Web-сервера: Apache Server, MS IIS, Netscape Enterprise Server, NGINX сервер.

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 i JSP

Технологія Servlet була розроблена компанією Sun Microsystems, і була призначена для того щоб серверна частина отримуючи запит від клієнта у якості відповіді повернула згенеровану HTML сторінку яка і відображає дані. Servlet – це спеціальний Java клас, який реалізує необхідні API для отримання запитів від WEB клієнта та після обробки надання результату. Сам по собі Servlet не чого робити не вміє він є частиною системи яка знаходиться у контейнері(Java web сервер) який вміє отримати запит і передати його необхідному додатку. Технологія JSP (JSPx) є „надбудовою” над сервлетами і призначена 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, WildFly, Jetty, Tomcat, Glassfish, Payara та ін. Майже усі вони виконують одні і ті ж самі задачі:

- трансляція JSP сторінок у сервлети;
- компіляція JSP сторінок;
- зв'язок з базою даних
- обмін даними між програмою та клієнтом;
- обмін даними між програмою та БД;
- автоматизація аутентифікації та авторизації користувачів

1.2.3 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: Oracle, MySQL, PostgreSQL.

EJB

EJB – ще одна технологія від Oracle (Sun Microsystems) яка призначена для реалізації Моделі у Java програмі. Вона реалізує зв'язок з БД незалежно від реалізації, та представлення у вигляді Java об'єктів. EJB поділяється на дві складові

- EntityBeans (класи які є об'єктним представленням інформації з БД)
- SessionBeans (класи та інтерфейси які контролюють, змінюють, і т.д. EntityBeans)

1.2.4 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-сервер надсилає запит Java-серверу.
- Java-сервер передає запит відповідному додатку (системі).
- Система отримавши запит, передає його відповідному сервлету, або JSP.
- Servlet(JSP) обробляючи запит звертається (якщо необхідно) до SessionBean
- SessionBean звертається до Java-сервера з проханням надати зв'язок з необхідним сервером БД та проводить аутентифікацію користувача по логіну і пароллю та вибирає потрібну базу.
- Отримавши зв'язок SessionBean проводить необхідний запит і отримавши результати повертає їх контролеру (Servlet або JSP)
- Контролер отримав результат генерує 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)

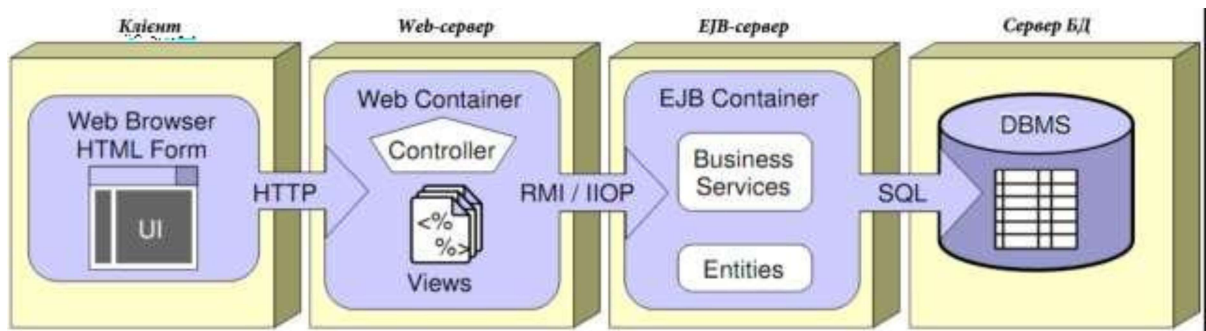


Рисунок 1.2 – Схема взаємодії технологій інформаційної системи

1.3 Формування вимог до системи управління мікрокліматом

Об'єктом автоматизації є мій власний дім. Мій дім має потребу у створенні єдиної системи управління кліматом будівлі.

1.3.2 Цілі, критерії й обмеження створення інформаційної системи

Метою розробки інформаційної системи є поліпшення умов життя шляхом автоматизації управління мікрокліматом.

Основні критерії створення інформаційної системи:

- Соціальний критерій. Інформаційна система розрахована на користувачів з високим рівнем володіння комп'ютером, тому достатньо простого інтерфейсу з необхідними функціями

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

- Критерій мобільності. Користувач повинен мати можливість працювати із системою з будь-якого комп'ютера локальної мережі

- 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.

1.3.3 Функції та задачі інформаційної системи

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

- Управління системою опалювання
- Управління системою кондиціонування
- Управління вентиляванням
- Управління освітленням
- Підтримка необхідних кліматичних умов у приміщенні

Для створення інформаційної системи необхідно розробити базу даних для збереження інформації про кімнати, відповідальних за температуру в них. Також повинно бути передбачене апаратно-програмне забезпечення для збору інформації про мікроклімат в приміщенні і необхідні бути передбачені засоби для адміністрування системи

1.3.4 Висновки і пропозиції

Таким чином, необхідно розробити автоматизовану інформаційну систему, що дозволить підвищити якість життя в будинку, а також зменшить витрати на електроенергію. Система є глобальної інформаційною системою і вирішує задачі, що пов'язані з управлінням мікроклімату. Інформаційна система повинна включати базу даних, програмне забезпечення жителя, а також відповідне організаційне, технічне та інші види забезпечення.

1.4 Розробка концепції інформаційної системи

1.4.1 Опис результатів вивчення об'єкта автоматизації

Інформаційна система розробляється як система управління кліматом житловому будинку. Для створення системи потрібна розробка бази даних інтерфейсу користувача, а також розробка механізму отримання і передачі інформації про кліматичні умови із приміщень.

1.4.2 Аналіз вимог до інформаційної системи

Крім функціональних вимог до інформаційної системи, необхідно врахувати вимоги до системи в цілому:

- Простий і зручний інтерфейс користувача.
- Доступ до системи через локальну мережу з будь-якого комп'ютера.
- Багатокористувачева робота.
- Висока масштабуємість системи.
- Висока продуктивність.
- Надійність.
- Конфіденційність інформації.

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

- Web-інтерфейс.
- Багаторівневу архітектуру.
- 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).

1.4.3 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. Крім того, в усіх системах маєтся презентаційне програмне забезпечення (ПЗ), ПЗ обробки бізнес-правил і даних, а також ПЗ, що відповідає за збереження інформації. 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, масштабованість 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.

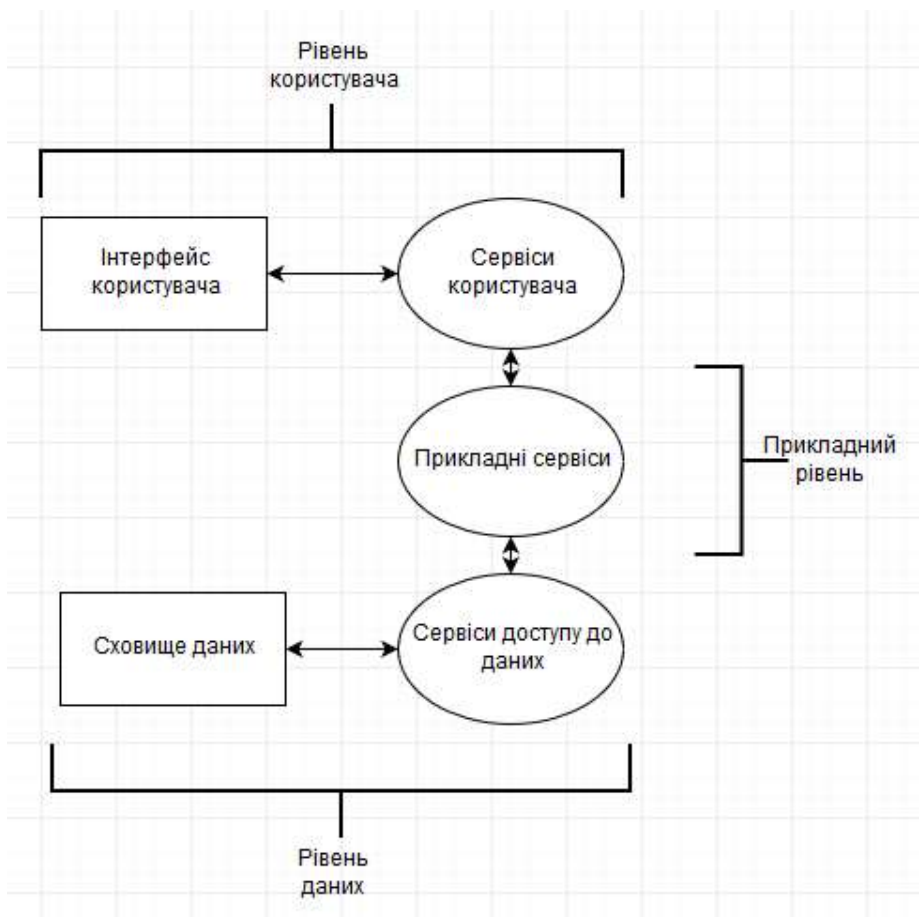


Рисунок 1.3 - Логічна архітектура програмного забезпечення інформаційної системи

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 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. 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.

1.4.4 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).

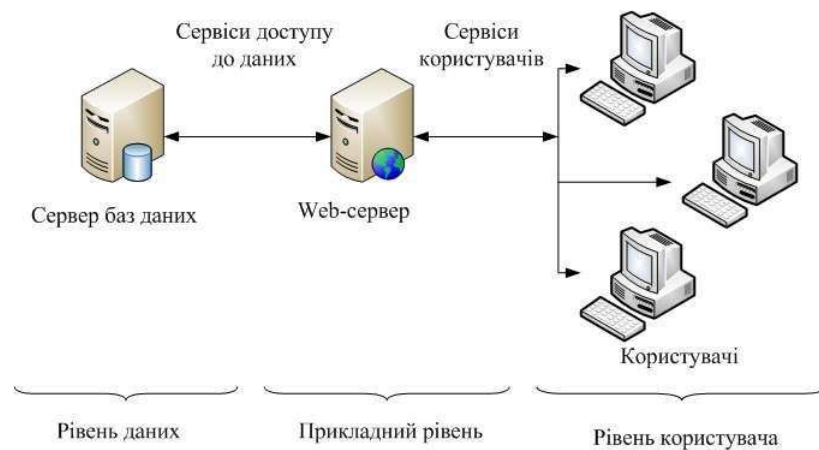


Рисунок 1.4 – Триланкова фізична архітектура програмного забезпечення інформаційної системи

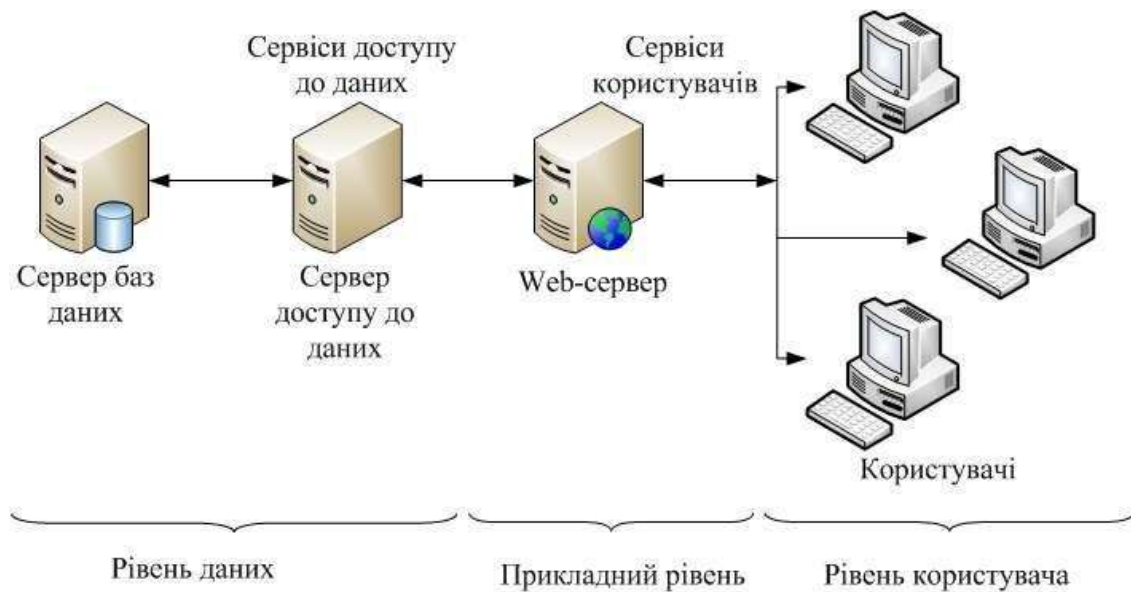


Рисунок 1.5 – Чотириланкова фізична архітектура програмного забезпечення інформаційної системи

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.

Таким чином, чотириланкова архітектура є більш перспективною. Вона в більшій мері задовольняє вимогам до інформаційної системи.

1.4.5 Вибір технологій для розробки інформаційної системи

Відповідно до обраної фізичної архітектури для інформаційної системи управління роботами, система є розподіленою і має чотири ланки: сервер баз даних, сервер доступу до даних, Web-сервер і клієнт.

Існують наступні технології для створення розподілених систем:

- DCOM/COM+, REST, 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 найгібкішою та одною з найсучасніших технологій, а також реалізується на мові високого рівня Java яка не залежить від операційної системи тому і обираємо цю технологію.

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

Найбільш перспективною й універсальною технологією є REST.

REST - підхід до архітектури мережевих протоколів, які забезпечують доступ до інформаційних ресурсів. Дані повинні передаватися у вигляді невеликої кількості стандартних форматів (наприклад, HTML, XML, JSON). Будь-який REST протокол (HTTP в тому числі) повинен підтримувати кешування, не повинен залежати від мережевого прошарку, не повинен зберігати інформації про стан між парами «запит-відповідь».

Антиподом REST є підхід, заснований на RPC. Підхід RPC дозволяє використовувати невелику кількість мережевих ресурсів з великою кількістю методів і складним протоколом. При підході REST кількість методів і складність протоколу суворо обмежені, що призводить до того, що кількість окремих ресурсів має бути великою.

REST, як і кожен архітектурний стиль відповідає ряду архітектурних обмежень. Це гібридний стиль який успадковує обмеження з інших архітектурних стилів.

Перша архітектура від якої він успадковує обмеження — це клієнт-серверна архітектура. Її обмеження вимагає розділення відповідальності між компонентами, які займаються зберіганням та оновленням даних (сервером), і тими компонентами, які займаються відображенням даних на інтерфейсі користувача та реагування на дії з цим інтерфейсом (клієнтом). Таке розділення дозволяє компонентам еволюціонувати незалежно.

Наступним обмеженням є те, що взаємодії між сервером та клієнтом не мають стану, тобто кожен запит містить всю необхідну інформацію для його обробки, і не покладається на те, що сервер знає щось з попереднього запиту.

Відсутність стану не означає що стану немає. Відсутність стану означає, що сервер не знає про стан клієнта. Коли клієнт, наприклад, запитує головну сторінку сайту, сервер відповідає на запитання і забуває про клієнта. Клієнт може залишити сторінку відкритою протягом кількох років, перш ніж натиснути посилання, і тоді сервер відповість на інший запит. Тим часом сервер може відповідати на запити інших клієнтів, або нічого не робити — для клієнта це не має значення.

Таким чином, наприклад дані про стан сесії (користувача, який аутентифікувався) зберігаються на клієнті, і передаються з кожним запитом. Це покращує масштабованість, бо сервер після закінчення обробки запиту може звільнити всі ресурси, задіяні для цієї операції, без жодного ризику втратити цінну інформацію. Також спрощується моніторинг і зневаження, бо для того аби розібратись, що відбувається в певному запиті, досить подивитись лише на той один запит. Збільшується надійність, бо помилка в одному запиті не зачіпає інші.

Мінусом цього обмеження є те, що знижується продуктивність через те, що в кожен запит тепер доводиться додавати дані сесії з клієнта. Також збереження стану на різних клієнтах важче підтримувати, бо реалізації клієнтів можуть відрізнятись, тоді як середовище сервера повністю під контролем розробника.

Додатковим обмеженням стилю REST є те, що системи, написані в цьому стилі, повинні підтримувати кешування, тобто дані, які передаються сервером, повинні містити інформацію про те, чи можна їх кешувати, і якщо можна, то як довго. Це дозволяє збільшувати продуктивність, уникаючи зайвих запитів, але також зменшує надійність системи, через те, що дані в кеші можуть бути застарілими.

Всі компоненти в архітектурі REST підтримують однорідний інтерфейс. Це зменшує зв'язність між компонентами і сервісами які вони надають і дозволяє нескладно змінювати компоненти при потребі.

Наступним обмеженням для REST є поділ на шари абстракції. Кожен компонент потрапляє в якийсь шар, і спілкується лише з компонентами в шарі під ним або в шарі над ним. Обмеження знання системи одним шаром зменшує складність компонентів. Саме тому REST є однією з найрозповсюджених технологій.

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 /найпродуктивніший 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 платформі 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.

1.4.6 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;
- система управління базами даних;
- 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-серверами, що задовольняють зазначеним вимогам і мають високу продуктивність, є NGINX. На Web-сервері повинна бути встановлена ОС яка підтримує цей сервер.

Так як використовуються технології організовані на Java на комп'ютері, то операційна система на сервері доступу до БД та управління системами опалювання, кондиціонування та вентиляції оптимально буде ОС Debian

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

- JBoss
- Payara
- GlassFish
- Tomcat

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.
- PostgreSQL .

- Sybase.
- Oracle (Standard Edition чи Enterprise Edition).

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 5.0, JavaScript, CSS 3 (наприклад, Firefox, Chrome, Opera ...).

Мережне середовище

Будинок має локальну комп'ютерну мережу Ethernet та WIFI, 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

Система розробляється для досить невеликого будинку, тому у вигляді серверу можна застосувати старий ноутбук, наприклад, Asus Eee PC 1001PX, або Chromebook.

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

- Мікропроцесор Intel Core i5.
- Оперативна пам'ять RAM 8Гб.
- Жорсткий диск з 40 Гб вільного місця і більше.

- Мережна карта Ethernet (100 Мбіт).

Засоби розробки

У якості засобу розробки планується використання об'єктно-орієнтованої мови програмування Java – OpenJDK 17. Це обґрунтовується, по-перше, незалежністю від платформи та, по-друге, безкоштовністю.

1.4.7 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.

Пропонується наступна конфігурація системи:

- Asus Eee PC 1001PX
 - процесор Intel Atom N450 з частотою 1666 МГц
 - 1 Гб оперативної пам'яті
 - Ethernet (RJ-45) 100 Мбит/с
 - Wi-Fi 802.11n
- Esp8266 NodeMCU в необхідній кількості з датчиками температури та вологості DHT-22 та датчиком вуглекислого газу MQ135 , а також датчиками переміщення(звичайний PIR датчик) і освітленности GY-2561, та радіомодулем з частотою 433 мГц(для управління вимикачами світла)

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 системою управління користувачами з різних підрозділів для виконання ними різних функцій. Схема цієї взаємодії, побудована в нотації варіантів використання системи (use case) показана на рисунку 2.1.

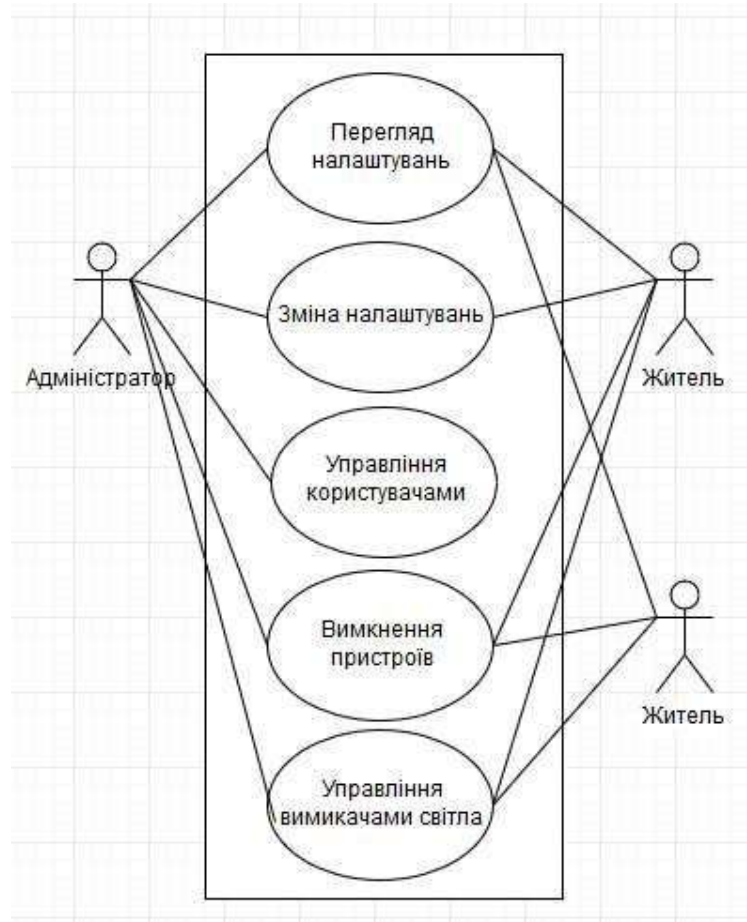


Рисунок 2.1. Схема організаційно-функціональної структури

У ході своєї роботи, жителі будинку використовують наступні функції

Користувач – перегляд та зміна заданих параметрів, вимкнення функцій

Адміністратор – перегляд та управління (зміна, видалення, створення нового) користувачами.

2.1.2 Опис функцій, що автоматизуються.

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

- Моніторинг клімату приміщень
- Управління кліматорегулюючим обладнанням
- Вимкнення світла в приміщеннях
- Моніторинг датчиків руху
- Моніторинг датчиків освітлення
- Усі інші вимоги не є головними і можуть змінюватися під час розробка або експлуатації

Для створення системи необхідно розробити єдину базу даних для збереження інформації про приміщення та їх жителів. 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 обладнанням, що буде задовольняти перерахованим вище функціональним вимогам. Необхідно також забезпечити надійність збереження і безпеку доступу до інформації Системи. Система повинна обслуговувати одночасну безперебійну роботу багатьом користувачів, причому зміни, внесені одними користувачами повинні бути доступні всім іншим користувачам негайно.

Програмне забезпечення повинне бути побудоване на платформі Debian jessie/nginx/MySQL/Payara/Java/IntelliJ IDEA і реалізовувати графічний інтуїтивно зрозумілий WEB інтерфейс користувача.

2.1.4 Загальний опис системи.

Система являє собою інструмент управління процесом розробки програмного забезпечення, побудований на основі мережі підприємства яке

спеціалізується на розробці програмних продуктів. Основними функціональними характеристиками системи є:

- ведення загальної інформаційної бази з розподіленим доступом;
- авторизований доступ до даних на основі прав доступу;
- відображення змін, внесених з будь-якого робочого місця;
- 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.

2.1.5 Захист від несанкціонованого доступу

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

Авторизація відбувається за рахунок вбудованого механізму авторизації серверу Payara, для реалізації цього механізму, програміст повинен лише реалізувати вікно авторизації, що містить форму для введення даних користувача. Крім того, адміністратор системи повинен внести права користувачів у відповідний конфігураційний файл, що використовується сервером Payara.

2.2 Рішення по організаційному забезпеченню

2.2.1 Організаційна структура підприємства

У ході впровадження і використання системи, її функціями охоплюються всі приміщення будинку. Схема організаційної структури при взаємодії із системою приведена на рисунку 2.2

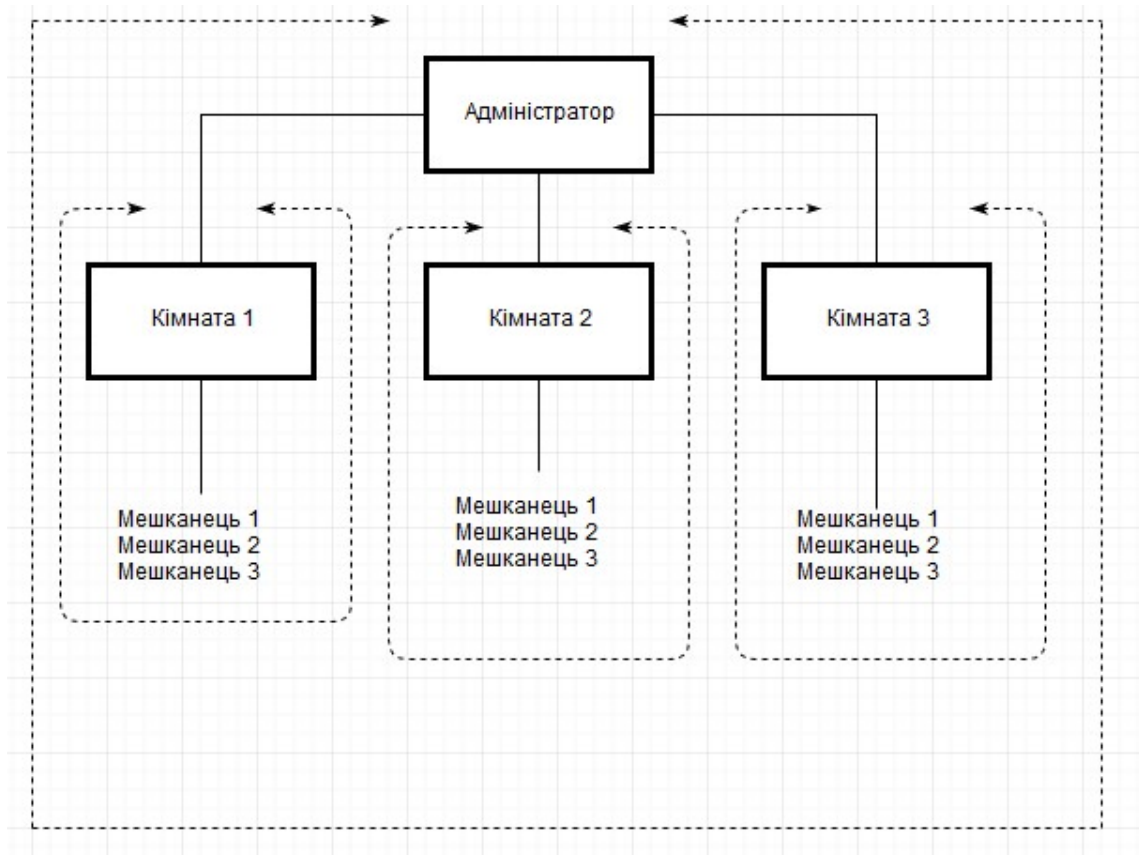


Рисунок 2.2. Структура житлового будинку

2.2.2 Інструкція користувача

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

2.2.3 Заходи щодо організаційного забезпечення

Ефективна робота системи 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;
- мінімізація 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 у будинку повинна бути введена штатна одиниця адміністратора.

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.

Житель – людина яка взаємодіє з системою

Дія – зміна параметрів

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

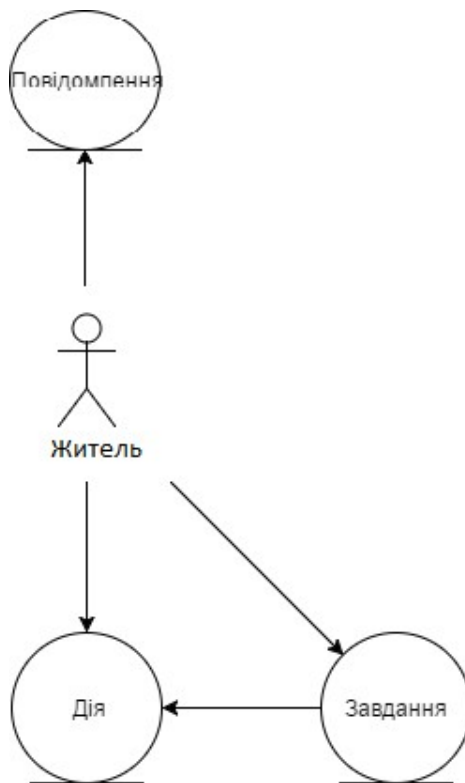


Рисунок 2.3. - Діаграма бізнес-сутностей системи.

На даному етапі проектування сутності словника даних стають кандидатами для класів проектованої системи.

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 чи успіх 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 не буде пред'являти підвищених вимог до обчислювальної потужності. Створювану систему можна віднести до класу облікових задач, у яких загальна продуктивність системи визначається швидкістю операцій введення-висновку при неповному залученні потужності центрального процесора. У такий спосіб будь-яке 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 відмовостійкості. 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 на сервері БД апаратного RAID контролера з можливістю відзеркалення дисків (дане рішення має досить сприятну вартість);

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 схеми розміщення наведена на рисунку 2.4.



Рисунок 2.4. - Схема розміщення комплексу технічних засобів.

2.5 Опис алгоритмів функціонування системи

Для уточнення алгоритмів функціонування розроблюваної системи використовуємо засобу об'єктно-орієнтованого аналізу й опису поведінки об'єктів. При цьому будемо розглядати як поведінку 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 2.5.

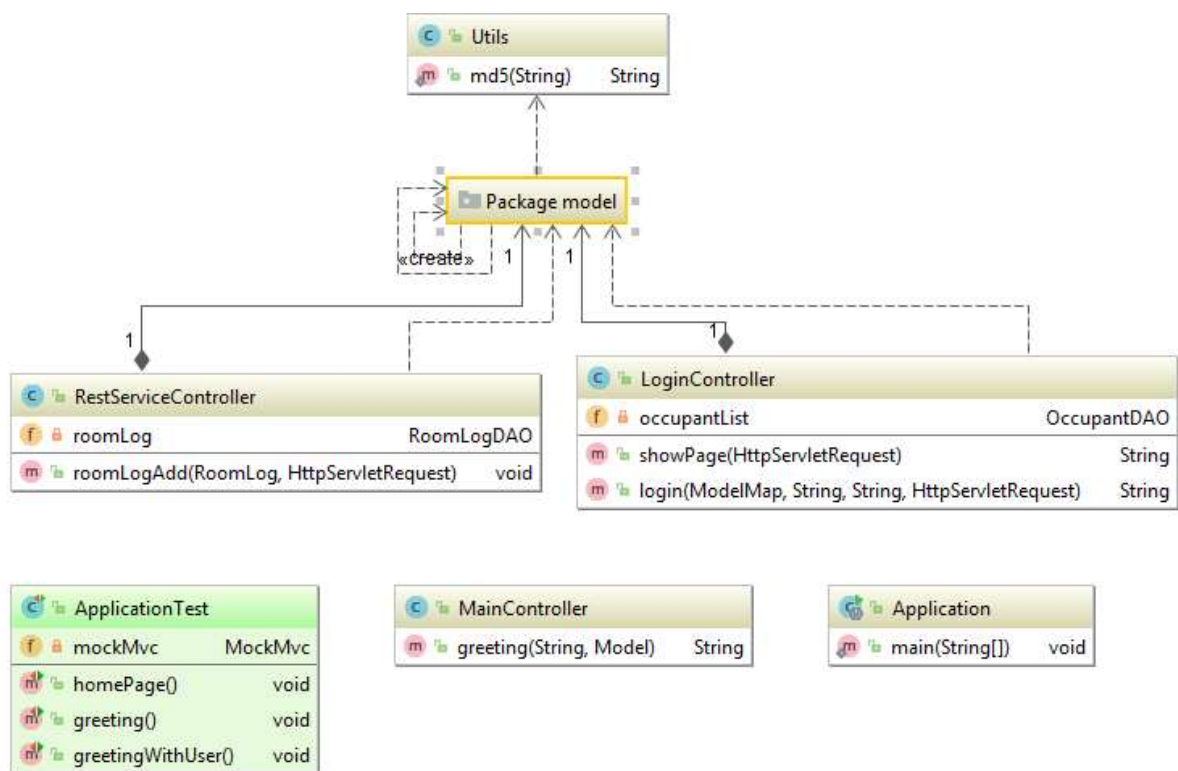


Рисунок 2.5.1 - Діаграма класів системи

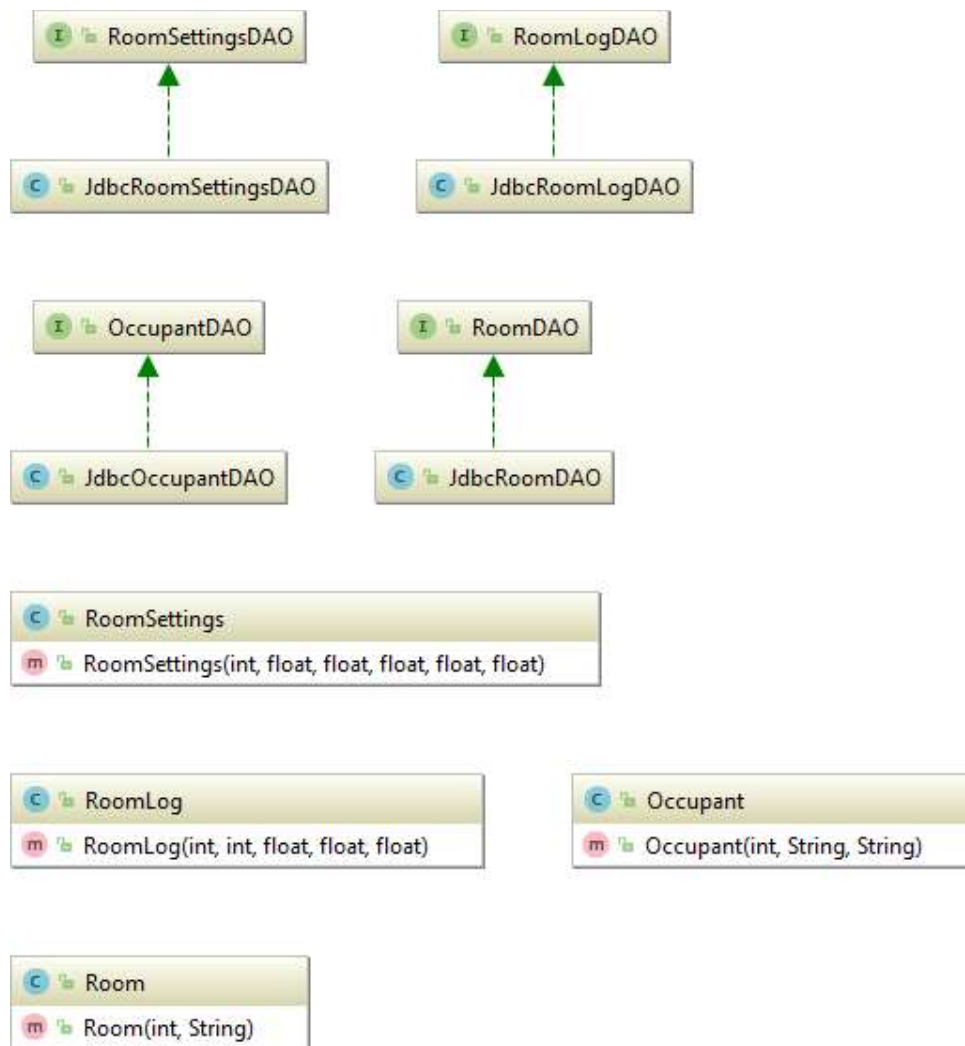


Рисунок 2.5.2 - Діаграма класів пакету `model`

Occupant(Житель) - користувач який може редагувати налаштування

Room(Кімната) – клас кімнати який зберігає в дані про кімнату

RoomSettings – клас який зберігає налаштування для кімнати

RoomLog – клас який зберігає статистичні дані від кімнат

LoginController – сервлет який перевіряє належність жителя до кімнати

MainController – сервлет що займається перевіркою параметрів кімнат

RestController – сервлет приймає повідомлення від датчиків від

мікроконтролерів

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 поведінки системи може бути описаний за допомогою діаграми станів (statechart diagram). Загальний алгоритм поведінки важливий для визначення взаємодій між компонентами системи й уточнення причин, що викликають та чи інша відповідна дія. Алгоритм поведінки системи в нотації діаграми станів приведений на рисунку 2.6.

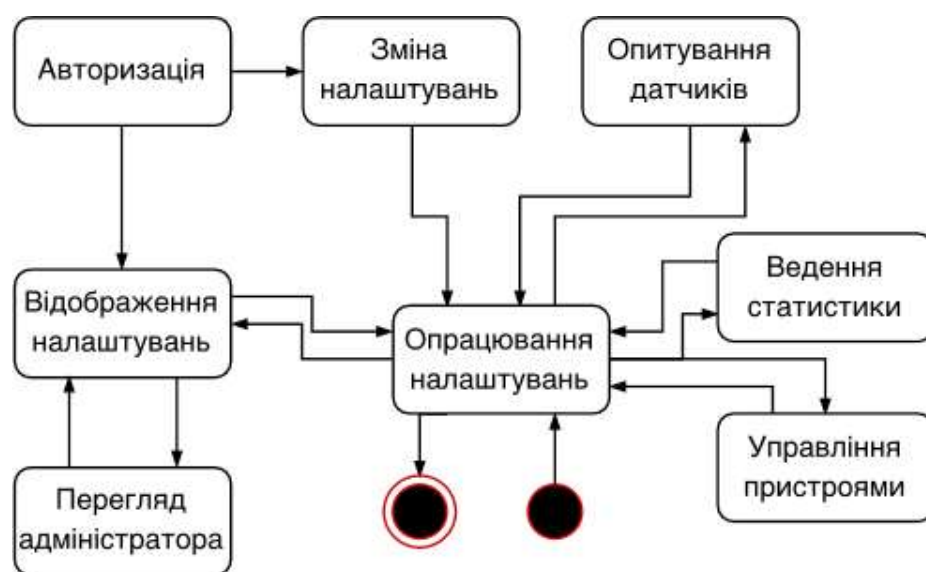


Рисунок 2.6. - Загальний алгоритм поведінки системи.

2.6 Рішення по програмному забезпеченню

2.6.1 Структура і функції частин програмного забезпечення

Функціональність і структуру проектованої автоматизованої системи поряд з моделлю поведінки 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 у даної роботи підходом об'єктно-орієнтованого аналізу і проектування виконаємо подальшу специфікацію проектних рішень у виді діаграм класів. Класи визначають стан пропонованої програмної системи (через значення своїх атрибутів) і, значною мірою, її поведінку (через операції чи методи класів).

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.

2.6.2 Фізична модель бази даних

Для забезпечення можливості зберігати дані в реляційній базі даних, з моделі класів була отримана фізична 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 сутностям були використані первинні ключі. Обрана СКБД не підтримує зберігання ієрархії спадкування класів, тому при будівництві фізичної 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 фізична модель наведена на рисунку 2.7.

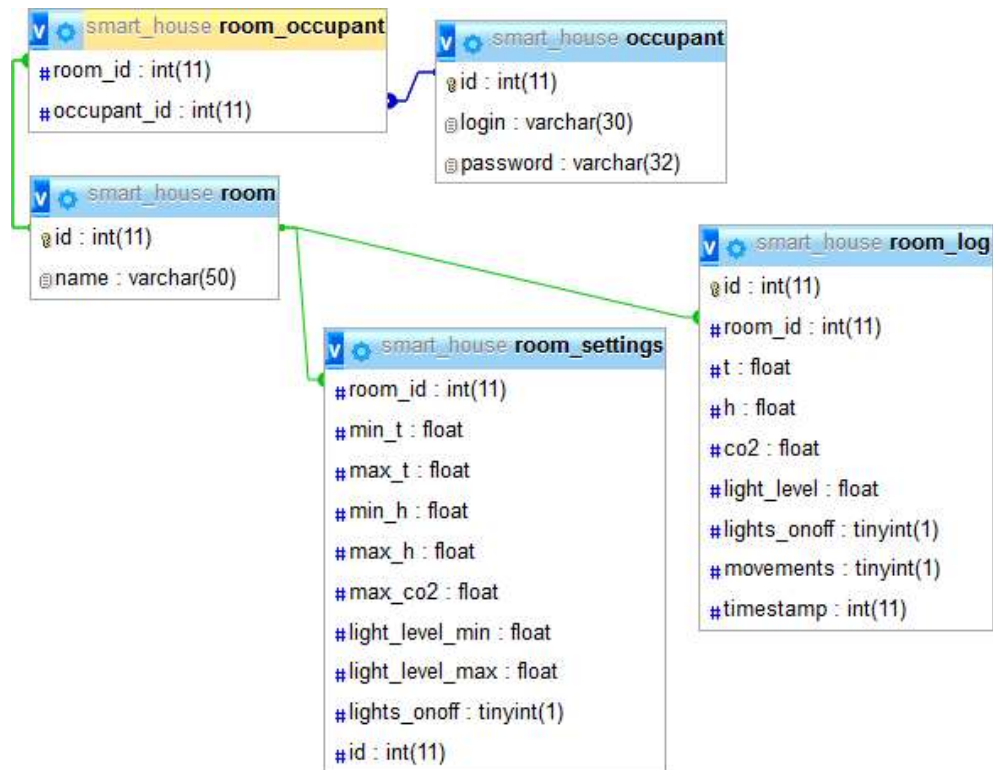


Рисунок 2.7. Фізична модель бази даних

2.6.3 Методи і засоби розробки програмного забезпечення

При аналізі і проектуванні Системи застосовувався об'єктно-орієнтований 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 система проектування мовою UML з відкритим кодом - ArgoUML виробництва спільноти Tigris Software – Tigris.org. На сьогоднішній день це один із потужних засобів проектування на ринку. Мова проектування UML - де-факто стандарт галузі. ArgoUML використовується на всіх етапах проектування від аналізу до автоматичної генерації коду програмної системи і має засоби 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 JSF. У якості середовища розробки був використаний продукт IntelliJ IDEA 2018.3

На Web-сервері повинен бути встановлений HTTP-сервер. Найрозповсюдженими HTTP-серверами, що задовольняють зазначеним вимогам і мають високу продуктивність, є Apache Server та nginx. На Web-сервері повинна бути встановлена ОС яка підтримує один з них.

Так як використовуються технології організовані на Java а також Raspberry Pi то операційна система на сервері доступу до БД може бути тільки Linux:

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

- JBoss
- Payara
- GlassFish
- Tomcat

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:

- MySQL.
- PostgreSQL .

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 5, JavaScript, CSS 3 (наприклад, FireFox, Opera, Chrome ...).

Програмне забезпечення для мікроконтролерів для зчитування показань датчиків та відправки їх на сервер розробляється з використанням Arduino IDE for esp8266 з набором сумісних бібліотек

3 Розробка робочої документації системи «Розумний дім» з віддаленим доступом

Система «Розумний дім» з віддаленим доступом призначена для автоматизації управління мікрокліматом, управлінням освітленням, управління автоматичним вимкненням світла. Функції, що автоматизуються системою, пов'язані з управлінням системами опалювання, кондиціювання та управління освітленням.

Основними функціями, що автоматизуються є:

- Управління системами мікроклімату.
- Моніторинг мікроклімату в приміщеннях.
- Управління вимикачами в приміщеннях
- Моніторинг освітлення в приміщеннях
- Усі інші вимоги не є головними і можуть змінюватися під час розробка 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 або мобільного пристрою.
- Багатокористувацька робота.
- Висока масштабуємість системи.
- Висока продуктивність.
- Надійність.
- Конфіденційність інформації.

Для ефективного використання програмного забезпечення важливо впровадити організаційні і технічні заходи, що передбачено в п. 2.2 та п. 2.4. При роботі з програмним забезпеченням системи варто керуватися документами «Технічне завдання» (Додаток А), «Програма та методика випробувань» (Додаток Б), «Керівництво користувача» (Додаток В)..

4 РОЗРАХУНОК ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ ВПРОВАДЖЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ МОНІТОРИНГУ

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 недавньому 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.

Обґрунтувати доцільність розробки з економічної точки зору можна, розрахувавши собівартість програмного забезпечення і додатковий прибуток, що буде отримана в результаті впровадження. Основним показником, що визначає економічну доцільність витрат на створення програмного забезпечення є річний економічний ефект. Характерним показником економічної ефективності створення і функціонування програмного продукту, що розраховується на стадії створення, впровадження і використання, є ефективність витрат і розрахунковий коефіцієнт економічної ефективності капітальних вкладень.

На першому етапі розраховується собівартість розробки програмного забезпечення. Потім капітальні витрати, необхідні для придбання

устаткування. На підставі цього можна одержати річний економічний ефект і строки окупності даного проекту.

4.1 Розрахунок витрат на створення й експлуатацію інформаційної системи

Розробка програмного забезпечення виконується двома програмістами відділу, місячний оклад кожного складає 1500 грн. Додаткова заробітна плата складає 20% від основної. Виходячи з цього, основна і додаткова заробітна плата розроблювачів системи 3600 грн./міс, а вартість сучасної ПЗВМ складає 4000 грн. (приведена вартість машини на базі Athlon X2). При вартості кіловат-години електроенергії рівної 0.50 грн., розраховується вартість розробки програми. Витрати на допоміжні матеріали приведені в Таблиці 4.1.

Таблиця 4.1 – Витрати на допоміжні матеріали

Пункти витрат	Сума грн.
Папір	50.00
Заправлення картриджа до принтера	100.00
Дискові носії (CD/DVD)	30.00
Література	100.00
Непередбачені витрати	20.00
Разом матеріали і комплектуючі вироби.	145.00

Вартість програми розрахуємо по формулі:

$$C_{\text{пр}} = (З_{\text{зп}} + З_{\text{сз}} + З_{\text{зг}} + З_{\text{е}}) * T + З_{\text{м}}, \quad (3.1)$$

де T - тривалість розробки, міс.

$З_{\text{зп}}$ - основна і додаткова заробітна плата обслуговуючого персоналу, грн.

$З_{\text{сз}}$ - відрахування на соціальні заходи (38.5% від основної і додаткової заробітної плати), грн.

$З_{\text{зг}}$ - загальногосподарські витрати (10% від основної заробітної плати), грн.

$З_{\text{е}}$ - витрати на електроенергію.

$З_{\text{м}}$ - витрати на основні і допоміжні матеріали.

При споживанні потужності 0.5 Вт, тривалості роботи на місяць, рівної $21 \cdot 6 = 126$ годин, вартості кіловат-години електроенергії 5грн.,

$$З_e = 126 \cdot 0.5 \cdot 5 = 315 \text{ грн.}$$

Таблиця 3.2 – Витрати на розробку системи

Найменування витрат	Одиниця виміру	Кількість
Тривалість розробки	Міс.	3
Основна і додаткова заробітна плата	грн.	3600
Відрахування на соціальні заходи	грн.	1350
Загальногосподарські витрати	грн.	360
Витрати на основні і допоміжні матеріали	грн.	145
Витрати на електроенергію	грн.	315

Відповідно до формули 3.1 вартість програми:

$$C_{\text{пр}} = (3600 + 1350 + 360 + 315) \cdot 3 + 145 = 17020 \text{ грн.}$$

Амортизаційні відрахування на устаткування складають 60% балансової вартості в рік:

$$A_{\text{об}} = 4000 \cdot 0,6 = 2400 \text{ грн.}$$

У масштабах підприємства річні витрати на основні і допоміжні матеріали (гнучкі диски, папір) визначаються в розмірі 5% вартості основного устаткування:

$$B_m = 4000 \cdot 0,05 = 200 \text{ грн.}$$

Річний обсяг робіт ПЗВМ у годинах визначається в такий спосіб:

$$\Phi_m = 253,3 \cdot T_z, \quad (3.2)$$

де T_z - це середнє місячне завантаження устаткування (близько 4 г.) 253,3 середня кількість робочих днів у році. Отже, річний обсяг роботи ПЕОМ складе:

$$\Phi_m = 253,3 \cdot 4 = 1013 \text{ годин}$$

Витрати на електроенергію $З_e$ при 1013 годинах роботи устаткування в рік складуть:

$$З_e = 1013 \cdot 0,24 \cdot 0,5 = 121,5 \text{ грн.}$$

Експлуатаційні витрати для ПЗВМ за рік складуть:

$$З_{\text{оп}} = 2400 + 200 + 121,5 = 2612,5 \text{ грн.}$$

Отже, у перший рік витрати на створення й експлуатацію програми складуть:

$$З_{се} = 17020 + 2612,5 = 19632,5 \text{ грн.}$$

4.2 Розрахунок економічної ефективності

Визначимо пряму економічну ефективність, ґрунтуючись на тому, що впровадження програмного продукту вивільняє 2,4 працівника (за експертною оцінкою фахівців підприємства).

Зарплата 2,4 працівника в рік складає:

$$(1300 * 12) * 2,4 = 37440 \text{ грн.}$$

Пряма економічна ефективність впровадження розроблювальної системи показана в Таблиці 3.3.

Таблиця 3.3 – Пряма економічна ефективність впровадження розроблювальної системи

У перший рік	$37440 - 18782 = 9866,325 \text{ грн.}$
У наступні	$37440 - 2612,5 = 15293,5 \text{ грн.}$

4.3 Розрахунок строку окупності програми

Витрати окупляться за:

$$(18782 / 9866,325) * 12 = 22 \text{ місяців.}$$

У такий спосіб витрати на створення програмного продукту окупляться протягом 22 місяців.

Висновки

Виходячи з розрахунків, впровадження системи окупить себе через 22 місяці. Отже можна зробити висновок, що розробка підсистеми для автоматизованого управління проектами на рівні компанії економічно вигідна й обґрунтована.

5 Охорона праці

Вступ

Охорона праці – це система законодавчих актів, соціально-економічних, організаційних, технічних і лікувально-профілактичних заходів і засобів, що забезпечують безпеку, збереження здоров'я і працездатність людини в процесі праці.

Охорона праці виявляє і вивчає можливі причини виробничих нещасливих випадків, професійних захворювань, аварій, вибухів, пожеж і розробляє систему заходів і вимог з метою усунення цих причин, і створення безпечних і сприятливих для людини умов праці.

З питаннями охорони праці нерозривно пов'язане і вирішення питань охорони природи.

Складність задач, які стоять перед охороною праці, вимагає використання досягнень і висновків багатьох наукових дисциплін, прямо або побічно пов'язаних із задачами створення здорових і безпечних умов праці.

Оскільки головним об'єктом охорони праці є людина в процесі праці, то при розробці вимог виробничої санітарії використовуються результати досліджень ряду медичних і біологічних дисциплін.

Особливо тісний зв'язок існує між охороною праці, науковою організацією праці, ергономікою, інженерною психологією і технічною естетикою.

Успіх у вирішенні проблем охорони праці здебільшого залежить від якості підготовки фахівців у цій області, від їхнього уміння приймати правильні рішення в складних і мінливих умовах сучасного виробництва.

5.1 Аналіз небезпечних і шкідливих факторів в приміщенні ІТ-компанії

При організації умов праці необхідно враховувати вплив на працюючих небезпечних і шкідливих виробничих факторів, які можуть привести до травми або іншого раптового різкого погіршення здоров'я та захворювання або зниження працездатності.

Небезпечні і шкідливі виробничі фактори (ДСТ 12.0.003-74) підрозділяються по природі дії на чотири групи: фізичні, хімічні, біологічні і психофізіологічні.

До небезпечних фізичних факторів відносяться: машини і механізми, що рухаються; різні підйомно-транспортні пристрої і переміщувані вантажі; незахищені рухливі елементи виробничого устаткування (приводні і передавальні механізми, різальні інструменти, пристосування, що обертаються і переміщуються й ін.); відлітаючі частки оброблюваного матеріалу та інструменту, електричний струм, підвищена температура поверхонь устаткування й оброблюваних матеріалів і т.д.

Шкідливими для здоров'я фізичними факторами є: підвищена чи знижена температура повітря робочої зони; висока вологість і швидкість руху повітря; підвищені рівні шуму, вібрації, ультразвуку і різних випромінювань – теплових, іонізуючих, електромагнітних, інфрачервоних і ін. До шкідливих фізичних факторів відносяться також запиленість і загазованість повітря робочої зони; недостатня освітленість робочих місць, проходів і проїздів; підвищена яскравість світла і пульсація світлового потоку.

Хімічні небезпечні і шкідливі виробничі фактори за характером дії на організм людини підрозділяються на наступні підгрупи: загальнотоксичні, подразнюючі, сенсibilізуючі (ті, що викликають алергійні захворювання), канцерогенні (ті, що викликають розвиток пухлин), мутогені (ті, що діють на статеві клітини організму). У цю групу входять численні пари і гази: пари бензолу і толуолу, окис вуглецю, сірчистий ангідрид, окисли азоту, аерозолі свинцю та ін., токсичні пили, що утворюються, наприклад, при обробці різанням берилію, свинцоватих бронз і латуней і деяких пластмас зі шкідливими наповнювачами. До цієї групи відносяться агресивні рідини (кислоти, луги), що можуть заподіяти хімічні опіки шкіряного покриву при зіткненні з ними.

До біологічних небезпечних і шкідливих виробничих факторів відносяться мікроорганізми (бактерії, віруси й ін.) та макроорганізми (рослини і тварини), вплив яких на працюючих викликає травми або захворювання.

До психофізіологічних небезпечних і шкідливих виробничих факторів відносяться фізичні перевантаження (статичні та динамічні) і нервово-психічні перевантаження (розумові перенапруги, перенапруга аналізаторів слуху, зору та ін.).

Між шкідливими і небезпечними виробничими факторами спостерігається визначений взаємозв'язок. У багатьох випадках наявність шкідливих факторів сприяє прояву травмонебезпечних факторів. Наприклад, надмірна вологість у виробничому приміщенні і наявність струмопровідного пилу (шкідливі фактори) підвищують небезпеку ураження людини електричним струмом (небезпечний фактор).

Рівні впливу на працюючих шкідливих виробничих факторів нормовані гранично-допустимими рівнями, значення яких зазначені у відповідних стандартах системи стандартів безпеки праці і санітарно-гігієнічних правил.

Гранично припустиме значення шкідливого виробничого фактора (за ДСТ 12.0.002-80) – це граничне значення величини шкідливого виробничого фактора, вплив якого при щоденній регламентованій тривалості протягом усього виробничого стажу не приводить до зниження працездатності і захворювання як у період трудової діяльності, так і до захворювання в наступний період життя, а також несприятливо не впливає на здоров'я потомства.

Науково-технічний прогрес вніс серйозні зміни в умови виробничої діяльності працівників розумової праці. Їхня праця стала більш інтенсивною, напруженою, потребує значних витрат розумової, емоційної і фізичної енергії. Це зажадало комплексного рішення проблем ергономіки, гігієни й організації праці, регламентації режимів праці і відпочинку.

На даний час комп'ютерна техніка широко застосовується у всіх галузях діяльності людини. При роботі з комп'ютером людина піддається впливу ряду небезпечних і шкідливих виробничих факторів: електромагнітних полів (діапазон радіочастот: ВЧ, УВЧ і СВЧ), інфрачервоного та іонізуючого випромінювань, шуму і вібрації, статичної електрики та ін.

Робота з комп'ютером характеризується значною розумовою напругою і нервово-емоційним навантаженням операторів, високою напруженістю зорової роботи і досить великим навантаженням на м'язи рук при роботі з клавіатурою ЕОМ. Велике значення має раціональна конструкція і розташування елементів

робочого місця, що важливо для підтримки оптимальної робочої пози людини-оператора.

У процесі роботи з комп'ютером необхідно дотримуватися правильного режиму праці і відпочинку. В противному випадку в персонала відзначається значна напруга зорового апарату з появою скарг на незадоволеність роботою, головні болі, дратівливість, порушення сну, утому і хворобливі відчуття в очах, у попереку, в області шиї і руках.

Розглянемо шкідливі виробничі фактори, які спостерігаються в приміщеннях, де здійснюється робота з електронно-обчислювальною технікою.

Освітлення

Правильно спроектоване і виконане виробниче освітлення поліпшує умови зорової роботи, знижує стомлюваність, сприяє підвищенню продуктивності праці, благотворно впливає на виробниче середовище, створюючи позитивний психологічний вплив на працюючого, підвищує безпеку праці і знижує травматизм.

Недостатність освітлення приводить до напруги зору, послабляє увагу, приводить до передчасної стомленості. Надмірно яскраве освітлення викликає осліплення, роздратування і різь в очах. Неправильний напрямок світла на робочому місці може створювати різкі тіні, відблиски, дезорієнтувати працюючого. Усі ці причини можуть привести до нещасливого випадку або профзахворювання, тому настільки важливий правильний розрахунок освітленості.

Існує три види освітлення – природне, штучне і сполучене (природне і штучне разом).

Природне освітлення – освітлення приміщень денним світлом, яке проникає через світлові прорізи в зовнішніх конструкціях приміщень. Природне освітлення характеризується тим, що міняється в широких межах в залежності від часу дня, пори року, характеру області і ряду інших факторів.

Штучне освітлення застосовується при роботі в темний час доби і вдень, коли не вдається забезпечити нормовані значення коефіцієнта природного освітлення (похмура погода, короткий світловий день). Освітлення, при якому недостатнє по нормах природне освітлення доповнюється штучним, називається сполученим освітленням.

Штучне освітлення підрозділяється на робоче, аварійне, евакуаційне, охоронне. Робоче освітлення, в свою чергу, може бути загальним чи комбінованим. Загальне – освітлення, при якому світильники розміщуються у верхній зоні приміщення чи рівномірно стосовно до розташування устаткування. Комбіноване – освітлення, при якому до загального додається місцеве освітлення.

Згідно до СНіп II-4-79 у приміщеннях обчислювальних центрів необхідно застосувати систему комбінованого освітлення.

При виконанні робіт категорії високої зорової точності (найменший розмір об'єкта розрізнення 0,3...0,5 мм) величина коефіцієнта природного освітлення (КЕО) повинна бути не нижче 1,5%, а при зоровій роботі середньої точності (найменший розмір об'єкта розрізнення 0,5...1,0 мм) КЕО повинний бути не нижче 1,0%. В якості джерела штучного освітлення зазвичай використовуються люмінесцентні лампи типу ЛБ або ДРЛ, які попарно поєднуються у світильники, та повинні розташовуватися над робочими поверхнями рівномірно.

Вимоги до освітленості в приміщеннях, де встановлені комп'ютери, наступні: при виконанні зорових робіт високої точності загальна освітленість повинна складати 300 лк, а комбінована – 750 лк; аналогічні вимоги при виконанні робіт середньої точності – 200 і 300 лк відповідно.

Крім того усе поле зору повинно бути освітлене досить рівномірно – це основна гігієнічна вимога. Іншими словами, ступінь освітлення приміщення і яскравість екрана комп'ютера повинні бути приблизно однаковими, тому що яскраве світло в районі периферійного зору значно збільшує напруженість очей і, як наслідок, приводить до їх швидкої стомлюваності.

Параметри мікроклімату

Параметри мікроклімату можуть мінятися в широких межах, в той час як необхідною умовою життєдіяльності людини є підтримка сталості температури тіла завдяки терморегуляції, тобто здатності організму регулювати віддачу тепла в навколишнє середовище. Принцип нормування мікроклімату – створення оптимальних умов для теплообміну тіла людини з навколишнім середовищем.

Обчислювальна техніка є джерелом істотного тепловиділення, яке може привести до підвищення температури і зниження відносної вологості в приміщенні.

У приміщеннях, де встановлені комп'ютери, повинні дотримуватися визначені параметри мікроклімату. У санітарних нормах СН-245-71

встановлені величини параметрів мікроклімату, що створюють комфортні умови. Ці норми встановлюються в залежності від пори року, характеру трудового процесу і характеру виробничого приміщення (таблиця 5.1).

Таблиця 5.1 – Параметри мікроклімату для приміщень, де встановлені комп'ютери

Період року	Параметр мікроклімату	Величина
Холодний	Температура повітря в приміщенні	22...24 °С
	Відносна вологість	40...60 %
	Швидкість руху повітря	до 0,1 м/с
Теплий	Температура повітря в приміщенні	23...25 °С
	Відносна вологість	40...60 %
	Швидкість руху повітря	0,1...0,2 м/с

Шум і вібрація

Шумом називають усякий несприятливо діючий на людину звук. З фізичної точки зору звук являє собою механічні коливання пружного середовища.

Слуховий орган людини сприймає у вигляді чутного звуку коливання пружного середовища, які мають частоту приблизно від 20 до 20000 Гц, але найбільш важливий для слухового сприйняття інтервал від 45 до 10000 Гц.

Сприйняття людиною звуку залежить не тільки від його частоти, але і від інтенсивності і звукового тиску.

Несприятлива дія шуму на людину залежить не тільки від рівня звукового тиску, але і від частотного діапазону шуму, а також від рівномірності впливу протягом робочого часу.

У результаті несприятливого впливу шуму на працюючу людину відбувається зниження продуктивності праці, збільшується брак у роботі, створюються передумови до виникнення нещасливих випадків. Усе це обумовлює велике оздоровче й економічне значення заходів щодо боротьби із шумом.

У таблиці 5.2 зазначені граничні рівні звуку в залежності від категорії ваги і напруженості праці, які є безпечними у відношенні збереження здоров'я і працездатності.

Таблиця 5.2 – Граничні рівні звуку, дБ, на робочих місцях.

Категорія напруженості праці	Категорія важкості праці			
	I Легка	II Середня	III Важка	IV Дуже важка
I Мало напружена	80	80	75	75
II Помірковано напружена	70	70	65	65
III Напружена	60	60	—	—
IV Дуже напружена	50	50	—	—

Електромагнітне й іонізуюче випромінювання

Більшість учених вважає, що як короткочасний, так і тривалий вплив усіх видів випромінювання від екрана монітора не небезпечний для здоров'я персоналу, що обслуговує комп'ютери. Однак вичерпних даних щодо безпеки впливу випромінювання від моніторів на працюючих з комп'ютерами не існує і дослідження в цьому напрямку продовжуються.

Припустимі значення параметрів неіонізуючих електромагнітних випромінювань від монітора комп'ютера представлені в таблиці 5.3.

Максимальний рівень рентгенівського випромінювання на робочому місці оператора комп'ютера зазвичай не перевищує 10 мкбер/год, а інтенсивність ультрафіолетового й інфрачервоного випромінювань від екрана монітора знаходиться в межах 10...100 мВт/м².

Таблиця 5.3 – Припустимі значення параметрів неіонізуючих електромагнітних випромінювань (відповідно до СанПіН 2.2.2.542-96)

Найменування параметра	Припустимі
Напруженість електричної складової електромагнітного поля або на відстані 50 см від поверхні відеомонітора	10 В/м
Напруженість магнітної складової електромагнітного поля або на відстані 50 см від поверхні відеомонітора	0,3 А/м
Напруженість електростатичного поля не повинна перевищувати:	20 кВ/м
для дорослих користувачів	
для дітей дошкільних установ і учнів	15 кВ/м
середніх спеціальних і вищих навчальних закладів	

5.2 Розробка заходів щодо зменшення впливу шкідливих факторів на підприємстві JGSoft

Організація і поліпшення умов праці на робочому місці є одним з найважливіших резервів продуктивності праці й економічної ефективності виробництва, а також подальшого розвитку самої працюючої людини. У цьому головний прояв соціального й економічного значення організації і поліпшення умов праці.

Нижче приведені найбільш розповсюджені організаційні заходи і технічні засоби захисту людей, які працюють з комп'ютерами, від шкідливих виробничих факторів.

Колірне оформлення і світлове відображення

Фарбування приміщень і меблів повинні сприяти створенню сприятливих умов для здорового сприйняття, гарного настрою.

Джерела світла, такі як світильники і вікна, що дають відображення від поверхні екрана, значно погіршують точність знаків і спричиняють перешкоди фізіологічного характеру, які можуть виразитися в значній напрузі, особливо при тривалій роботі. Відображення, включаючи відображення від вторинних джерел світла, повинне бути зведене до мінімуму. Для захисту від надлишкової яскравості вікон можуть бути застосовані штори й екрани.

У залежності від орієнтації вікон рекомендується наступна фарбування стін і підлоги:

- *вікна орієнтовані на південь*: стіни – зеленувато-блакитний або світло-блакитний кольори; підлога – зелений;
- *вікна орієнтовані на північ*: стіни – світло-жовтогарячий або оранжево-жовтий кольори; підлога – червонясто-жовтогарячий;
- *вікна орієнтовані на схід*: стіни – жовто-зеленого кольору; підлога – зелений або червонясто-жовтогарячий;
- *вікна орієнтовані на захід*: стіни – жовто-зелений або блакитнувато-зелений кольори; підлога – зелений чи червонясто-жовтогарячий.

У приміщеннях, де знаходиться комп'ютер, необхідно забезпечити наступні величини коефіцієнта відображення: для стелі – 60...70%, для стін – 40...50%, для підлоги – близько 30%. Для інших поверхонь і робочих меблів – 30...40%.

Шум і вібрація

Зниження шуму, створюваного на робочих місцях внутрішніми джерелами, а також шуму проникаючого ззовні, є дуже важливою задачею. Зниження шуму в джерелі надходження можна забезпечити застосуванням пружних прокладок між підставою машини, приладу й опорною поверхнею. В якості прокладки використовуються гума, повсть, пробка, різної конструкції амортизатори. Під настільні шумливі апарати можна підкладати м'які коврики із синтетичних матеріалів, а під ніжки столів, на яких вони встановлені, – прокладки з м'якої гуми, повсті, товщиною 6-8 мм. Кріплення прокладок можливе шляхом приклейки їх до опорних частин.

Можливе також застосування звукоізолюючих кожухів, що не заважають технологічному процесу. Не менш важливим для зниження шуму в процесі експлуатації є питання правильного і своєчасного регулювання, змазування і заміни механічних вузлів шумливого устаткування.

Рациональне планування приміщення і розміщення устаткування є важливим чинником, що дозволяє знизити шум при існуючому устаткуванні ЕОМ. Приміщення необхідно розташовувати в далечині від шумливого і вібруючого устаткування.

Зниження рівня шуму, що проникає у виробниче приміщення ззовні, може бути досягнуте збільшенням звукоізоляції обгороджуючих конструкцій, ущільненням по периметру притворів вікон, дверей.

Таким чином для зниження шуму, створюваного на робочих місцях внутрішніми джерелами, а також шуму, що проникає з поза, слід:

- послабити шум самих джерел (застосування екранів, звукоізолюючих кожухів);
- знизити ефект сумарного впливу відбитих звукових хвиль (звуковбирні поверхні конструкцій);
- застосовувати рациональне розташування устаткування;
- використовувати архітектурно-планувальні і технологічні рішення ізоляцій джерел шуму.

Мікроклімат

Нормалізація мікроклімату приміщення може бути досягнута поліпшенням вентиляції приміщення за рахунок витяжних вентиляторів і нормального функціонування вбудованої вентиляції будинку.

Освітлення

У приміщеннях з обчислювальною технікою, як правило, застосовується бічне природне освітлення. Робочі кімнати і кабінети повинні мати природне освітлення. В інших приміщеннях допускається штучне освітлення.

Освітленість на поверхні столу в зоні розміщення робочого документа повинна бути 300-500 лк, також допускається установка світильників місцевого освітлення для підсвічування документів, але з такою умовою, щоб воно не створювало відблисків на поверхні екрана і не збільшувало освітленість екрана більш ніж на 300 лк.

В якості джерела світла при штучному висвітленні повинні застосовуватися переважно люмінесцентні лампи типу ЛБ. При порядкуванні відбитого освітлення в адміністративно-суспільних приміщеннях допускається застосування металогалогених ламп потужністю до 250 Вт. Допускається застосування ламп накаливання у світильниках місцевого освітлення.

Загальне освітлення варто виконувати у вигляді суцільних або переривчастих ліній світильників, розташованих збоку від робочих місць, паралельно лінії зору користувача при рядному розташуванні ПЕОМ. При периметральному розташуванні комп'ютерів лінії світильників повинні розташовуватися локалізовано над робочим столом, ближче до його переднього краю, зверненого до оператора.

Для забезпечення нормованих значень освітленості в приміщеннях використання ПЕОМ варто проводити чищення стікол віконних рам і світильників не рідше двох разів у рік і проводити своєчасну заміну перегорілих ламп.

Ергономіка робочого місця

Проектування робочих місць, оснащених відеотерміналами, відноситься до числа важливих проблем ергономічного проектування в області обчислювальної техніки.

Робоче місце і взаємне розташування всіх його елементів повинне відповідати антропометричним, фізичним і психологічним вимогам. Велике значення має також характер роботи. Зокрема, при організації робочого місця програміста повинні бути дотримані наступні основні умови: оптимальне розміщення устаткування, що входить до складу робочого місця і достатній робочий простір, що дозволяє здійснювати всі необхідні рухи і переміщення.

Ергономічними аспектами проектування відеотермінальних робочих місць, зокрема, є: висота робочої поверхні, розміри простору для ніг, вимоги до розташування документів на робочому місці (наявність і розміри підставки для документів, можливість різного розміщення документів, відстань від очей користувача до екрана, документа, клавіатури і т.д.), характеристики робочого крісла, вимоги до поверхні робочого столу, регулюємість елементів робочого місця.

Головними елементами робочого місця програміста є стіл і крісло. Основним робочим положенням є положення сидячи.

Робоча поза сидячи викликає мінімальне стомлення програміста. Раціональне планування робочого місця передбачає чіткий порядок і сталість розміщення предметів, засобів праці і документації. Те, що потрібно для виконання робіт частіше, розташовується в зоні легкої досяжності робочого простору.

Для комфортної роботи стіл повинний задовольняти наступним умовам:

- висота столу повинна бути обрана з урахуванням можливості сидіти вільно, у зручній позі, при необхідності спираючись на підлокітники;
- нижня частина столу повинна бути сконструйована так, щоб програміст міг зручно сидіти, не був змушений підгортати ноги;
- поверхня столу повинна мати властивості, що виключають появу відблисків у полі зору програміста;
- конструкція столу повинна передбачати наявність висувних шухляд (не менше ніж 3 для збереження документації, лістингів, канцелярських предметів);
- висота робочої поверхні рекомендується в межах 680-760 мм. Висота поверхні, на яку встановлюється клавіатура, повинна бути близько 650 мм.

Велике значення надається характеристикам робочого крісла. Так, рекомендована висота сидіння над рівнем підлоги знаходиться в межах 420-550 мм. Поверхня сидіння м'яка, передній край закруглений, а кут нахилу спинки – регульований.

Необхідно передбачати при проектуванні можливість різного розміщення документів: збоку від відеотерміналу, між монітором і клавіатурою і т.п. Крім того, у випадках, коли відеотермінал має низьку якість зображення, наприклад помітні мелькання, відстань від очей до екрана роблять більше (близько 700 мм), чим відстань від ока до документа (300-450 мм). Узагалі при високій

якості зображення на відеотерміналі відстань від очей користувача до екрана, документа і клавіатури може бути рівною.

Велике значення також надається правильній робочій позі користувача. При незручній робочій позі можуть з'явитися болі в м'язах, суглобах і сухожиллях. Вимоги до робочої пози користувача відеотермінала наступні:

- голова не повинна бути нахилена більш ніж на 20° ,
- плечі повинні бути розслаблені,
- лікті – під кутом $80^{\circ} \dots 100^{\circ}$,
- передпліччя і кисті рук – у горизонтальному положенні.

Причина неправильної пози користувачів обумовлена наступними факторами: немає гарної підставки для документів, клавіатура знаходиться занадто високо, а документи – низько, нікуди покласти руки і кисті, недостатній простір для ніг.

З метою подолання зазначених недоліків даються загальні рекомендації: краще пересувна клавіатура; повинні бути передбачені спеціальні пристосування для регулювання висоти столу, клавіатури й екрана, а також підставка для рук.

Істотне значення для продуктивної і якісної роботи на комп'ютері мають розміри знаків, щільність їхнього розміщення, контраст і співвідношення яркостей символів і фону екрана. Якщо відстань від очей оператора до екрана дисплея складає 60...80 см, то висота знака повинна бути не менше ніж 3 мм, оптимальне співвідношення ширини і висоти знака складає 3:4, а відстань між знаками – 15...20% їхньої висоти. Співвідношення яскравості фону екрана і символів – від 1:2 до 1:15.

Під час користування комп'ютером медики радять установлювати монітор на відстані 50-60 см від очей. Фахівці також вважають, що верхня частина відеодисплея повинна бути на рівні очей або трохи нижче. Коли людина дивиться прямо перед собою, її очі відкриваються ширше, ніж коли вона дивиться вниз. За рахунок цього, площа огляду значно збільшується, викликаючи зневоднювання очей. До того ж якщо екран установлений високо, а очі широко відкриті, порушується функція моргання. Це значить, що очі не закриваються цілком, не омиваються слізною рідиною, не одержують достатнього зволоження, що приводить до їх швидкої стомлюваності.

Режим праці

Для підтримки тривалої працездатності людини велике значення має режим праці і відпочинку. Під раціональним фізіологічно обґрунтованим режимом праці і відпочинком мається на увазі таке чергування періодів роботи з періодом відпочинку, при якому досягається висока ефективність суспільно-корисної діяльності людини, гарний стан здоров'я, високий рівень працездатності і продуктивності праці.

У таблиці 5.4 представлені зведення про регламентовані перерви, які необхідно робити при роботі на комп'ютері, в залежності від тривалості робочої зміни, видів і категорій трудової діяльності з ВДТ (відеодисплейний термінал) і ПЕОМ (відповідно до СанПіН 2.2.2 542-96 «Гігієнічні вимоги до відеодисплейних терміналів, персональних електронно-обчислювальних машин і організації робіт»).

Таблиця 5.4 – Час регламентованих перерв при роботі на комп'ютері

Категорія роботи з ВДТ чи ПЕОМ	Рівень навантаження за робочу зміну при видах			Сумарний час регламентованих перерв, хв	
	Група А, кількість знаків	Група Б, кількість знаків	Група В, годин	При 8-годинній зміні	При 12-годинній зміні
I	до 20 000	до 15 000	до 2,0	30	70
II	до 40 000	до 30 000	до 4,0	50	90
III	до 60 000	до 40 000	до 6,0	70	120

Примітка. Час перерв даний при дотриманні зазначених Санітарних правил і норм. При невідповідності фактичних умов праці вимогам Санітарних правил і норм час регламентованих перерв варто збільшити на 30%.

У відповідності із СанПіН 2.2.2 546-96 усі види трудової діяльності, пов'язані з використанням комп'ютера, розділяються на три групи:

група А: робота зі зчитування інформації з екрана ВДТ чи ПЕОМ із попереднім запитом;

група Б: робота з введення інформації;

група В: творча робота в режимі діалогу з ЕОМ.

Ефективність перерв підвищується при сполученні з виробничою гімнастикою або організацією спеціального приміщення для відпочинку персоналу зі зручними м'якими меблями, акваріумом, зеленою зоною і т.п.

Створення сприятливих умов праці і правильне естетичне оформлення робочих місць на виробництві має велике значення як для полегшення праці, так і для підвищення його привабливості, що позитивно впливає на продуктивність праці.

6 Охорона навколишнього середовища

ВСТУП

Охорона навколишнього середовища це - система заходів щодо раціонального використання природних ресурсів, збереження особливо цінних та унікальних природних комплексів і забезпечення екологічної безпеки. Це сукупність державних, адміністративних, правових, економічних, політичних і суспільних заходів, спрямованих на раціональне використання, відтворення і збереження природних ресурсів землі, обмеження негативного впливу людської діяльності на навколишнє середовище.

У вирішенні цих проблем особливої актуальності набуває переосмислення людством шляхів подальшого розвитку. Це завдання, безперечно, є пріоритетним, йому мають бути підпорядковані державна політика, господарська діяльність, наукові розробки. З цього погляду, наша держава потребує переходу від здійснення комплексних заходів, спрямованих на охорону довкілля, до сталого еколого-економічного функціонування.

Будь яка країна, що стала на шлях науково-технічного прогресу та широкомасштабного використання його результатів, вже не може і не повинна ігнорувати такі об'єктивні чинники, як вичерпаність багатьох природних ресурсів, насамперед невідтворювальних і невідновлюваних, вразливість навколишнього середовища, екологічну стійкість та екологічну місткість довкілля, межі його екологічної міцності і опірності щодо негативних і шкідливих антропогенних впливів тощо. Всі ці чинники необхідно всебічно враховувати в господарській діяльності і при визначенні темпів та масштабів соціально-економічного розвитку на майбутнє. Цей розвиток має бути врівноваженим і адекватним екологічній ситуації, та узгоджуватися з природничими законами.

Оскільки вчасно не була вирішена задача охорони природи від негативного впливу людини, тепер все частіше виникає необхідність захисту людини від впливу навколишнього середовища.

6.1. Забруднення навколишнього середовища

Промислові підприємства в процесі експлуатації споживають певну кількість чистої води, а також скидають очищені або неочищені стічні води в навколишнє середовище, що призводить до забруднення гідрографічної мережі і території району його розміщення. Основними джерелами забруднення поверхневих вод є:

- неочищені або недостатньо очищені виробничі і побутові стічні води;
- поверхневий стік з території підприємства;
- фільтраційні витoki шкідливих речовин з ємностей, трубопроводів та інших споруд;
- аварійні скиди стічних вод.

З метою раціонального використання та охорони поверхневих вод підприємство повинно забезпечити:

- економне і раціональне використання водних ресурсів;
- наявність ліцензії та договору на користування водним об'єктом та дотримання їх умов;
- запобігання та усунення забруднення поверхневих вод;
- утримання в справному стані очисних, гідротехнічних та інших водогосподарських споруд і технічних пристроїв;
- наявність контрольно-вимірювальної апаратури з визначення якості забирається і скидається у водний об'єкт води і дотримання строків її державної атестації;
- організацію обліку забирають, використовуваних і скидаються вод, кількості забруднюючих речовин в них, а також систематичні спостереження за водними об'єктами та їх водоохоронними зонами;
- дотримання встановлених лімітів забору води та скидання стічних вод;
- дотримання встановлених лімітів забору води та скидання стічних вод;
- розробку інженерних заходів щодо запобігання аварійних скидів неочищених або недостатньо очищених стічних вод, щодо

забезпечення екологічно безпечної експлуатації водозабірних споруд та водних об'єктів;

- дотримання встановленого режиму використання водоохоронних зон;
- запобігання потрапляння продуктів виробництва і супутніх йому забруднюючих речовин на територію виробничого майданчика промислового об'єкту і безпосередньо у водні об'єкти;
- розробку плану заходів на випадок можливого екстремального забруднення водного об'єкта.

У процесі господарської діяльності забороняється скидати у водні об'єкти стічні (поворотні) води які містять речовини або продукти трансформації речовин у воді, для яких не встановлені ГДК або ОДР, а також речовини, для яких відсутні методи аналітичного контролю, за винятком тих речовин, що містяться у воді водного об'єкта, які з урахуванням їх складу та місцевих умов можуть бути спрямовані в системи оборотного водопостачання для повторного використання або для інших цілей. Води які надають токсичну дію, за результатами біотестування, на живі організми, дощові і талі води, що відводяться з територій промислових майданчиків, які не пройшли очистку до встановлених вимог. Забороняється скидання вод в межах першого і другого поясів зон санітарної охорони джерел господарсько-питного водопостачання, округів санітарної охорони курортів, у водні об'єкти, що використовуються для лікувальних цілей, а також в місцях масового скупчення риб. Також забороняється скидання вод які містять збудників інфекційних захворювань, а також містять речовини, концентрації яких перевищують ГДК та їх фонові значення у водному об'єкті, якщо для них не встановлені норми гранично допустимого скиду (ГДС), зазначені в дозволі на скид стічних вод.

Підприємства повинні забезпечувати санітарний стан підвідомчої території і не допускати винесення через дощову каналізаційну мережу сміття і відходів виробництва. Не допускається проводити у водних об'єктах та на їх берегах миття транспортних засобів, інших механізмів, а також

проведення будь-яких робіт, що можуть з'явитися джерелом забруднення вод. З метою запобігання забруднення, засмічення, замулення і виснаження водних об'єктів створюються водоохоронні зони. В їх межах встановлюються прибережні захисні смуги, на територіях яких вводяться додаткові обмеження природокористування.

Для оцінювання природних вод існують певні показники, головними серед яких є гігієнічні ГДК (гранично допустимі концентрації) їх дотримання забезпечує нормальний стан здоров'я населення і сприятливі умови для санітарно-побутового використання. Вони також є критеріями ефективності заходів з охорони водойм від забруднення, а також стимулами прогресу в галузі промислової технології. Ці гігієнічні нормативи використовуються також для оцінки комплексного забруднення поверхневих вод. Вони визначалися з урахуванням запаху, кількості завислих речовин, прозорості, кольору, окислюваності, вмісту розчинного кисню, біологічної потреби кисню (БПК), щільного залишку, кількості солей, хлоридів, фенолів, нафтопродуктів, жорсткості тощо.

Аналіз ситуації показав, що малі річки України забруднені більше, ніж великі. Це пояснюється не тільки їхньою малою водністю, але й недостатньою охороною. Найбільш забруднені Південний Буг, річки Донецької і Луганської областей, Чорноморського узбережжя півдня України. Щороку до водоймищ України потрапляє 5 млн тонн солей і значна частина стоків від тваринницьких комплексів. Майже половина мінеральних добрив і отрутохімікатів змивається з полів у ріки. В табл. 5.1 проілюстровано обсяги скидання забруднених зворотних вод у природні поверхневі об'єкти за регіонами.

Рівень очищення води надзвичайно низький. Існуючі очисні споруди навіть при біологічному очищенні вилучають лише 10—40 % неорганічних речовин (40 % азоту, 30 % фосфору, 20 % калію) і практично не вилучають солі важких металів.

У басейні Дніпра — найбільшої водної артерії країни — крім Чорнобильської атомної електростанції діють також Запорізька, Південно-Українська, Хмельницька, Рівненська. У Дніпро щорічно скидається 370 млн

кубометрів забруднених стоків, або 14 % від їх обсягу по країні. Значна частина річного стоку Дніпра використовується промисловими підприємствами при гранично допустимих 20 %. Це дуже знижує якість води, а також рибопродуктивність і може призвести до втрати Дніпра як постачальника питної води.

36 мільйонів жителів України, що споживають води Дніпра, можуть залишитися без питної води вже в XXI столітті. Наслідки забруднення водного середовища можуть бути дуже різноманітними для здоров'я людини. Шкоди можуть завдати такі поширені забруднювачі як фторо-, хлоро- і фосфороорганічні забруднювачі, нітрати, нітроти, нітросполуки, пестициди, гербіциди тощо. Ці та інші негативні явища відбуваються на тлі низьких запасів води в Україні, які складають 97,3 куб. км (у маловодні роки — 66 куб. км). Дефіцит води в Україні вже зараз складає 4 млрд кубометрів.

Деградація, висихання малих річок невідворотно призведе до деградації великих рік, тому проблема їхнього збереження й оздоровлення є однією з найгостріших для нашої молоді держави.

У країні проводиться значна робота з охорони вод від забруднення[^] Розробляються схеми комплексного використання і охорони вод, згідно з цими схемами здійснюється вибір ділянок під будівництво об'єктів, кожен проект будівництва і реконструкції промислових та інших об'єктів проходить екологічну експертизу.

Таблиця 6.1 Скидання забруднених зворотних вод у природні поверхневі об'єкти за регіонами у 2011р.

Адміністративне утворення	Всього	У тому числі		Частка у загальному обсязі скидання зворотних вод, відсотків
		без очищення	недостатньо очищених	
Україна	2948	804	2144	31
Автономна	30	10	20	7

Республіка Крим				
Області				
Вінницька	51	6	45	55
Волинська	4	0	4	7
Дніпропетровська	664	200	464	44
Донецька	745	130	615	44
Житомирська	28	3	25	43
Закарпатська	В	1	12	25
Запорізька	403	240	163	36
Івано-Франківська	41	4	37	48
Київська	32	20	12	4
Кіровоградська	46	0	46	58
Луганська	280	39	241	66
Львівська	83	3	80	28
Миколаївська	79	36	43	73
Одеська	210	53	157	77
Полтавська	12	1	11	6
Рівненська	Зі	2	29	32
Сумська	13	0	13	18
Тернопільська	4	1	3	6
Харківська	18	10	8	5
Херсонська	28	1	27	25
Хмельницька	8	0	8	11
Черкаська	27	8	19	12
Чернівецька	10	6	4	21
Чернігівська	35	—	35	27
м. Київ	26	26	—	3

Деградація, висихання малих річок невідворотно призведе до деградації великих рік, тому проблема їхнього збереження й оздоровлення є однією з найгостріших для нашої молоді держави.

У країні проводиться значна робота з охорони вод від забруднення[^]. Розробляються схеми комплексного використання і охорони вод, згідно з цими схемами здійснюється вибір ділянок під будівництво об'єктів, кожен проект будівництва і реконструкції промислових та інших об'єктів проходить екологічну експертизу.

Що стосується очищення стічних вод, то в Україні діє понад 2,8 тис. очисних споруд з самостійним випусканням стічних вод у водні об'єкти. Серед них споруд біологічного очищення — 60 %, механічного — 36 % і фізико-хімічного — 5 %. Понад 300 міст мають споруди повного біологічного очищення. З метою охорони вод від забруднення потрібно пришвидшити введення нового порядку лімітування скидів, плати за скиди забруднюючих речовин.

6.2. Розробка заходів щодо зменшення забруднення поверхневих вод

Сучасний розвиток промисловості, сільського господарства, транспорту, а також ріст міст супроводжується великим скидом забруднених вод. При відсутності належних заходів із зниження забруднення води, природне розбавлення і самоочищення стає недостатньою. Великі концентрації шкідливих домішок перешкоджають самоочищенню води і її забруднення інтенсивно прогресує.

Тому, для збереження чистоти водойм, необхідно:

- забезпечити повну очистку комунально-побутових і промислових стоків;
- вдосконалювати і змінювати технологію промислового виробництва;
- розроблювати і впроваджувати маловодну і безводну технології;
- широко впроваджувати оборотне водопостачання, розширювати

повторне використання очищених стічних вод;

- застосовувати раціональні способи і прийоми використання добрив і пестицидів;

- розробляти і здійснювати державні плани водоохоронних заходів в масштабах басейнів річок і водойм з урахуванням перспективного розташування продуктивних сил і засобів виробництва.

Наразі існують такі способи очищення стічних вод: механічна, фізико-хімічна, хімічна і біохімічна.

Механічна очистка служить для відокремлення нерозчинних речовин шляхом проціджування, відстоювання, фільтрування і центрифугування. Застосовують її, як попередню перед іншими способами очистки, або у випадках, коли стічні води, які пройшли через згадані пристрої, використовують з метою виробництва чи, якщо можливо, їх скидають у водойму. Воду, яка пройшла механічну очистку, як правило хлорують.

Хімічні і фізико-хімічні способи застосовують для очистки виробничих стічних вод від колоїдних і розчинних речовин. Для цього, у відповідності з характером забруднення, у воду вводять спеціальні реагенти, пропускають повітря чи пару, використовують електроліз та іонообмінні матеріали.

Біохімічна очистка основана на властивості деяких організмів використовувати для свого розвитку органічні речовини стічних вод. Цей спосіб використовують після того, як стічна вода очищена від мінеральних і нерозчинних органічних речовин. Він дозволяє майже повністю видалити забруднення органічного походження. Біохімічну очистку проводять в природних умовах – на полях зрошення, а також в штучних умовах – в біологічних фільтрах.

Охорона води від забруднення добривами і пестицидами. Забруднення водойм не є обов'язковим супутником інтенсифікації сільськогосподарського виробництва. Більш того, при правильне використання мінеральних добрив покращує структуру ґрунту і підвищує його стійкість до водної і вітрової ерозії.

Для запобігання попадання добрив у водойми необхідно:

- дотримувати відповідність норм кількості добрив потребі рослин;

- встановлювати оптимальні терміни внесення добрив;
- вносити добрива в подрібленому вигляді в період вегетації рослин;
- вносити добрива разом із зрошувальною водою, що зменшує їх дозу.

Для обмеження попадання пестицидів у водні об'єкти необхідно:

- вдосконалювати систему їх застосування;
- застосовувати стрічкову чи крайову обробку замість суцільної;
- ширше застосовувати біологічні методи захисту рослин;
- розробляти менш шкідливі види пестицидів;
- забороняти хімічну обробку за допомогою авіації.

Використання стоків тваринницьких комплексів. Задача боротьби з пагубною дією тваринницьких стоків на водні об'єкти довгий час рахувалась важкою із-за великої їх кількості, складності утилізації і вивозу, труднощі забезпечення санітарного стану гноєсховищ. Ця проблема не вирішена і зараз для більшості невеликих і неспеціалізованих господарств.

Захист води від забруднення синьо-зеленими. "Цвітіння" води в результаті інтенсивного розвитку синьо-зелених водоростей явище закономірне. Воно виникає в результаті діяльності людини з формування біоценозів, створення водосховищ і т.п.

До заходів, які регулюють розвиток синьо-зелених можна віднести такі:

- різке зменшення притоку харчових продуктів у водойму за рахунок поверхневих змивів і стічних вод;
- видалення водоростей з подальшим їх використанням в господарських цілях;
- локальне видалення мулових відкладень, які акумулюють значні запаси біогенних елементів;
- підвищення ступеня кисневого насичення природних шарів води за рахунок додаткової аерації.

Висновки

У результаті виконання даної дипломної роботи було розроблено проект системи розумного будинку з віддаленим доступом.

Для досягнення цієї мети були вирішені наступні задачі:

- сформульовані вимоги до автоматизованої системи управління;
- розроблені концепції системи, та проведено вибір оптимального варіанту;
- розроблені проектні рішення по системі з різних видів забезпечення, передбачених етапами ескізного та технічного проектів, зокрема загальносистемні рішення, рішення з організаційного забезпечення, рішення з інформаційного забезпечення, рішення з технічного забезпечення, рішення з програмного забезпечення.

- розроблена робоча документація по системі;
- створено програмно-апаратну реалізацію прототипа системи

Отримані результати дозволяють зробити висновок, що у разі успішної реалізації та впровадження системи управління, підвищиться ефективність управління мікрокліматом та освітленням, що призведе до покращення умов проживання, що призведе загального підвищення якості життя.

Список літератури

1. Методичні вказівки до виконання дипломного проектування зі спеціальності 7.080401 «Інформаційні управляючі системи та технології». – К.В. Кошкін, С.Б. Приходько, С.В. Суслов, В.І. Тимофєєв. – Миколаїв: УДМТУ, 2003. – 52 с.
2. Кріс Шефер, Кларенс Хо, Роб Харроп. Spring 4 для професіоналів, APress, 2016
3. Роберт Лафоре, Структури даних та алгоритми Java, SAMS, 2016
4. Бретт Маклафлін. Об'єктно-орієнтований аналіз та проектування, O'REILLY Media, 2013
5. Сейед Тахагхогхі, Х'ю Є. Вільямс. Посібник з MySQL, O'REILLY Media, 2007
6. Герберт Шилдт. Java8. Полное руководство, 9-е изд., М.:Вильямс, 2015
7. Внедрение SQL-кода. Википедия — свободная энциклопедии. [Електроний ресурс]. – Режим доступу URL: http://ru.wikipedia.org/wiki/SQL_injection.
8. Межсайтовый скриптинг. Википедия — свободная энциклопедии. [Електроний ресурс]. – Режим доступу URL: <http://ru.wikipedia.org/wiki/Xss>.
9. Лановий О.Ф., Кобзєв С.В., Калякін С.В., Системи управління контентом і безпека WEB-сайтів. Системи обробки інформації. Харків; ХНУВС, 2010
10. Безопасность жизнедеятельности. Под редакцией Белова Н.А. – М.: Знание, 2000.– 364с.
11. ГОСТ 34.603 – 92. Информационная технология. Виды испытаний автоматизированных систем.
12. Новиков Ю.В. Экология, окружающая среда и человек: Учебное пособие для вузов. – М.: Агентство —ФАИР, 1998. – 295с.
13. Самгин Э.Б. Освещение рабочих мест. – М.: МИРЭА, 1989. – 186с.

14. Борьба с шумом на производстве: Справочник / Юдина Э.Я., Борисова Л.А. под редакцией Юдина Э.Я. – М.: Машиностроение, 1985. – 400с.

Додаток А – Технічне завдання на розробку

1 Вступ

1.1 Назва – Підсистема web-доступу в системі «Розумний дім»

1.2 Призначення програми: автоматизація роботи системи «Розумний дім» засобами web-технологій.

1.3 Область застосування: домашнє використання.

2 Призначення розробки

Призначена для автоматизації роботи кліматорегулюючих засобів, задля підтримання необхідних умов у житлових приміщеннях.

3 Вимоги до системи

3.1 Вимоги до функціональних характеристик

Система буде містити наступні функціональні характеристики :

- Збір інформації про кліматичні умови
- Виведення інформації про користувачів приміщення
- Автоматичне управління кліматичними системами
- Додання інформації в систему.

4.2 Вимоги до надійності

4.2.1 Вимоги до забезпечення надійності функціонування програми

Надійне (стійке) функціонування системи має бути забезпечено надійним (стійким) функціонуванням бездротової мережі і мережі електропостачання в офісному центрі.

4.2.2 Відмова виконання програмного продукту через некоректні дії користувача.

- Перевірка формату вводу даних для коректної роботи.
- Функція автоматичного backup кожен тиждень.

4.3 Умови експлуатації

Данною програмою будуть користуватися орендатори офісного центру, які є просунутими користувачами, та не потребують додаткового навчання по даному ПО.

4.5 Вимоги до інформаційної та програмної сумісності

4.5.1 Вимоги до інформаційних структур і методів розв'язання

Вимоги до інформаційних структур і методів розв'язання відсутні.

4.5.2 Вимоги до вихідних кодів і мов програмування

Додаткові вимоги не пред'являються.

4.5.3 Вимоги до програмних засобів , які використовуються програмою
Програмні засоби що використовуються мають бути вільно поширювані

4.6 Вимоги до маркування та упаковки

4.6.1 Вимоги до маркування :

- Читабельність тексту, чіткість ілюстрацій;
- Достовірність інформації про систему
- Нанесення на Install CD позначок про дане ПО.

4.6.2 Вимоги до упаковки:

Вимоги до упаковки відсутні.

4.7 Вимоги до транспортування та зберігання

Вимоги до транспортування програми відсутня.

4.8 Спеціальні вимоги

Спеціальні вимоги відсутні.

5 Вимоги до програмної документації

5.1 Попередній склад програмної документації

Попередній склад програмної документації повинен включати в себе:

- технічне завдання ;
- устав проекту .

5.2 Склад програмної документації

Програмна документація складається з наступних програмних документів:

- Устав проекту;
- Технічне завдання ;
- Опис системи ;
- Код програми ;
- Керівництво користувача.

Сертифікація містить список всіх програмних документів .

Технічне завдання містить опис завдання роботи , склад функціональних характеристик програми , опис вимог для розробки.

Опис програми містить опис логічної структури програми , форматів вхідних та вихідних даних , використовуваних технічних засобів.

Код програми містить опис блоків програми і запис блоків в програмному коді.

Керівництво користувача містить інформацію , необхідну для ознайомлення користувача з програмним продуктом.

6 Техніко-економічні показники

6.1 Економічні переваги розробки

Орієнтовна економічна ефективність не розраховується.

7 Стадії та етапи розробки програмного забезпечення

7.1 Стадії розробки

Розробка повинна бути проведена в 5 стадій:

1. Технічне завдання
2. Ескізний проект
3. Технічний проект
4. Робочий проект
5. Впровадження

Додаток Б - Програма і методика випробувань

Об'єкт випробувань

Назва об'єкта випробувань: «Login.java» , надалі називається «програма».
Програма дозволяє користувачеві пройти аутентифікацію та отримати доступ до свого інтерфейсу.

Ціль випробувань

Перевірка придатності програми до використання за призначенням, її відповідність заданим вимогам і програмним документам. Перевірка реально досягнутих характеристик програми.

Вимоги до програми

Програма повинна задовольняти наступним функціональним характеристикам:

- 1) перевірка правильності імені та паролю користувача;
- 2) надання користувачеві доступу до свого інтерфейсу.

Користувач повинен вказати ім'я та пароль.

Вимоги до програмної документації

У комплект програмної документації повинні входити наступні документи:

- Загальний опис системи.
- Посібник користувача – власника товарів.
- Посібник користувача – дилера.
- Посібник користувача – покупця.
- Інструкція з формування і ведення бази даних.
- Посібник адміністратора з установки програмного забезпечення на серверах.
- Програма і методика випробувань.
- Текст програми.

Засоби і порядок випробувань

Вимоги до складу і параметрів технічних засобів:

Сервер на базі комп'ютера с архітектурою x86 або ARM

Вимоги до складу і параметрів програмних засобів :

- 1) операційна система Debian Jessie;
- 2) встановлене з'єднання з сервером баз даних MariaDB 10.1

Група людей, що будуть проводити випробування, повинна складатися з автора програми (розробника) і представника і/або самого замовника. При наявності відповідної можливості, ця група може бути розширена програмістом, спеціалістом-тестувальником і фахівцем-проектувальником, що не брали участь у розробці програми.

Попередньо (не пізніше 3 днів) до початку випробувань уточнюється кінцевий склад групи, що буде проводити випробування. Кожен член групи одержує необхідну документацію до програми (список вимог до програми, опис програми, керівництво користувача, програму і методику випробувань).

Порядок проведення випробувань:

- послідовно виконуються контрольні тести; результати заносяться у відповідний журнал;
- виробляється аналіз отриманих результатів і встановлюється відповідність програми заданим вимогам і програмним документам.

Методи випробувань

Робота програми залежить тільки від двох параметрів, що вводяться користувачем: від імені та паролю користувача.

Іспити будуть проводитися методом «чорної шухляди».

Тести будуть створюватися на основі класів еквівалентності.

Класи еквівалентності виділяються шляхом вибору кожної вхідної умови і розбивкою його на двох або більш груп. Розрізняють два типи класів еквівалентності: правильні класи еквівалентності, що представляють правильні вхідні дані програми, і неправильні класи еквівалентності, що представляють всі інші можливі стани умов (помилкові вхідні значення). При цьому існує ряд правил:

- один тест може покривати тільки один неправильний клас еквівалентності;
- один тест повинний покривати як можна більше число непокритих правильних класів еквівалентності.

Виділені класи еквівалентності для складання тестів мають унікальні номери і приводяться в таблиці Д.1.

Таблиця Б.1 – Класи еквівалентності

Операція, що тестується	Вхідні умови	Правильні класи еквівалентності	Неправильні класи еквівалентності
Перевірка імені та паролю користувача	Ім'я правильне	Так (1)	Ні (2)
	Пароль правильний	Так (3)	Ні (4)
Надання користувачеві доступу до свого інтерфейсу	Належність користувача до інтерфейсу встановлена	Так (5)	Ні (6)

Тести

Тести складені на основі класів еквівалентності таблиці Д.1. Тести використовують наступні початкові умови:

- 1) користувач – user1, його пароль – pass1 (власник товару);
- 2) користувач – admin, його пароль – adMpaSSw0rd (адміністратор системи).

Тест 1

Класи еквівалентності:

1, 3, 5

Вхідні дані:

Ім'я користувача: user1

Пароль користувача: pass1

Результат:

Ім'я та пароль користувача пізнані. Належність встановлена. Наданий доступ до інтерфейсу жителя.

Тест 2

Класи еквівалентності:

2, 3, 5

Вхідні дані:

Ім'я користувача: user2

Пароль користувача: pass1

Результати:

Ім'я та пароль користувача не пізнані. Належність до користувачів системи не встановлена. Видане повідомлення про помилку введення імені чи паролю. Доступу до системи не надано.

Тест 3

Класи еквівалентності :

1, 4, 5

Вхідні дані:

Ім'я користувача: user1

Пароль користувача: pass2

Результати:

Ім'я та пароль користувача не пізнані. Належність до користувачів системи не встановлена. Видане повідомлення про помилку введення імені чи паролю. Доступу до системи не надано.

Тест 4

Класи еквівалентності:

1, 3, 5

Вхідні дані:

Ім'я користувача: admin

Пароль користувача: adMpaSSw0rd

Результати:

Ім'я та пароль користувача впізнані. Наданий доступ до адміністрування системи.

Додаток В. - Інструкція користувача автоматизованої системи управління завданнями.

1. Призначення програми.

Програмне забезпечення інформаційної системи управління мікроклімату, яка призначена для автоматизації роботи кліматичних систем на будь-якому рівні. Функції, що автоматизуються системою, пов'язані з управлінням системою опалення, кондиціонування та вентилявання, а також моніторингом кліматичних умов.

Задачею інформаційної системи, є автоматизація наступних функцій користувача:

- Облік кліматичних умов.
- Моніторинг кліматичних умов.
- Формування звітів.

Для створення інформаційної системи необхідно розробити єдину базу даних для збереження інформації про кімнати та їх жителів, необхідні їм кліматичні умови. Також повинно бути передбачене програмне забезпечення для адміністрування системи ПЗ функціонує в мережі житлового будинку, обмін даними будується через єдину інформаційну базу.

2. Умови виконання програми.

Для виконання програми необхідна наявність на в будинку наступних апаратних і програмних засобів:

- робочі місця користувачів, що представляють собою персональні комп'ютери(ПК) з мінімальною конфігурацією Celeron +/- із встановленим веб оглядачем; мінімальний обсяг вільного місця на твердому диску 4Гб;
- локальної бездротової мережі для надання доступу мікроконтролерів до сервера з робочих місць користувачів доступу до даних, із пропускною здатністю 10/100Мбіт/с;

3. Виконання програми.

Виконання програми на кожному з робочих місць починається запуском веб оглядача та вводом адреси системи у мережі. Безпосередньо після цього Користувач системи повинний пройти авторизацію – ввести в діалоговому вікні, що з'явилося, користувальницьке ім'я і пароль. Пароль є унікальною комбінацією буквено-

цифрових символів, відомих тільки користувачу. Якщо ви забули пароль – звернетесь до системного адміністратора.

Після успішного входу в систему користувач бачить екран с інформацією про кліматичні умови а також які кліматичні системи вмикнені.

На цьому ж екрані користувач може змінити налаштування своєї кімнати

- на екран де може змінити свої налаштунки (кількість секунд для оновлення екрану завданнів, отримувати листа при отриммані завдання, звіту або повідомлення), після заповнення форми користувач повинен натиснути на кнопку „Save” щоб дані буди збереженні у базі;

- на екран статистики де відображується статистика за обраний період.

Після вибору початкового періоду та кінцевого користувач бачить умови що були в вказаний термін,

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

Користувач може вийти з системи натиском на відповідне посиляння з екрану статистики, він може завершити дію веб оглядача, але у цілях безпеки рекомендується виходити з системи.

4. Повідомлення користувачу.

При введенні некоректних даних над полями уведення виводиться діалогове вікно з описом типу і місця виникнення помилки.

При неправильному введенні чи пароля імені користувача виводиться повідомлення «Wrong login or password».

Додаток Г. - Тексти програмних модулів системи.

Occupant.java

```
package app.model;

import java.util.Objects;

public class Occupant {
    private int id;
    private String login;
    private String password;

    public Occupant(int id, String password, String login) {
        this.id = id;
        this.password = password;
        this.login = login;
    }

    @Override
    public int hashCode() {
        return Objects.hash(id, login, password);
    }

    @Override
    public boolean equals(Object obj) {
        return super.equals(obj);
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getLogin() {
        return login;
    }

    public void setLogin(String login) {
        this.login = login;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }
}
```

```
    }  
}
```

OccupantDAO.java

```
package app.model;  
  
public interface OccupantDAO {  
    public void createOccupant(String login, String password);  
    public Occupant authOccupant(String login, String password);  
}
```

Room.java

```
package app.model;  
  
public class Room {  
    private int id;  
    private String name;  
  
    public Room(int id, String name) {  
        this.id = id;  
        this.name = name;  
    }  
  
    public int getId() {  
        return id;  
    }  
  
    public void setId(int id) {  
        this.id = id;  
    }  
  
    public String getName() {  
        return name;  
    }  
  
    public void setName(String name) {  
        this.name = name;  
    }  
}
```

RoomDAO.java

```
package app.model;  
  
import java.util.List;  
  
public interface RoomDAO {  
    public List<Room> getAllRoom();  
}
```

```
        public List<Occupant> getRoomOccupants(int roomId);  
    }
```

RoomLog.java

```
package app.model;  
  
public class RoomLog {  
    private int id;  
    private float t;  
    private float h;  
    private float co2;  
    private int timestamp;  
  
    public RoomLog(int id, int timestamp, float co2, float h,  
float t) {  
        this.id = id;  
        this.timestamp = timestamp;  
        this.co2 = co2;  
        this.h = h;  
        this.t = t;  
    }  
  
    public int getId() {  
        return id;  
    }  
  
    public void setId(int id) {  
        this.id = id;  
    }  
  
    public float getT() {  
        return t;  
    }  
  
    public void setT(float t) {  
        this.t = t;  
    }  
  
    public float getH() {  
        return h;  
    }  
  
    public void setH(float h) {  
        this.h = h;  
    }  
  
    public float getCo2() {  
        return co2;  
    }  
  
    public void setCo2(float co2) {
```

```

        this.co2 = co2;
    }

    public int getTimestamp() {
        return timestamp;
    }

    public void setTimestamp(int timestamp) {
        this.timestamp = timestamp;
    }
}

```

RoomLogDAO.java

```

package app.model;

import java.util.List;

public interface RoomLogDAO {
    public List<RoomLog> getRoomLog(int roomId);
    public void insertRoomLogEntry(int roomId, RoomLog entry);
}

```

RoomSettings.java

```

package app.model;

public class RoomSettings {
    private int id;
    private float min_t;
    private float max_t;
    private float min_h;
    private float max_h;
    private float max_co2;

    public RoomSettings(int id, float max_co2, float max_h, float
max_t, float min_t, float min_h) {
        this.id = id;
        this.max_co2 = max_co2;
        this.max_h = max_h;
        this.max_t = max_t;
        this.min_t = min_t;
        this.min_h = min_h;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }
}

```

```

    public float getMinT() {
        return min_t;
    }

    public void setMinT(float min_t) {
        this.min_t = min_t;
    }

    public float getMaxT() {
        return max_t;
    }

    public void setMaxT(float max_t) {
        this.max_t = max_t;
    }

    public float getMinH() {
        return min_h;
    }

    public void setMinH(float min_h) {
        this.min_h = min_h;
    }

    public float getMaxH() {
        return max_h;
    }

    public void setMaxH(float max_h) {
        this.max_h = max_h;
    }

    public float getMaxCo2() {
        return max_co2;
    }

    public void setMaxCo2(float max_co2) {
        this.max_co2 = max_co2;
    }
}

```

RoomSettingsDAO.java

```

package app.model;

public interface RoomSettingsDAO {
    public RoomSettings getRoomSettings(int roomId);
    public void setRoomSettings(RoomSettings settings);
}

```

Utils.java

```
package app;

import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStream;
import java.math.BigInteger;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.util.Properties;

public class Utils {

    private static Connection connection = null;

    public static String md5(String st) {
        MessageDigest messageDigest;
        byte[] digest = new byte[0];

        try {
            messageDigest = MessageDigest.getInstance("MD5");
            messageDigest.reset();
            messageDigest.update(st.getBytes());
            digest = messageDigest.digest();
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        }

        BigInteger bigInt = new BigInteger(1, digest);
        String md5Hex = bigInt.toString(16);

        while( md5Hex.length() < 32 ){
            md5Hex = "0" + md5Hex;
        }

        return md5Hex;
    }

    public static Connection getConnection() {
        if (connection != null)
            return connection;
        else {
            try {
                Properties prop = new Properties();
                InputStream inputStream =
Utils.class.getClassLoader().getResourceAsStream("/db.properties");
                prop.load(inputStream);
                String driver = prop.getProperty("driver");
                String url = prop.getProperty("url");
                String user = prop.getProperty("user");
                String password = prop.getProperty("password");
                Class.forName(driver);
            }
        }
    }
}
```



```

        connection = DriverManager.getConnection(url, user,
password);
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    } catch (SQLException e) {
        e.printStackTrace();
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }
    return connection;
}

}
}

```

Esp8266-sender-sketch.ino

```

#include <ArduinoJson.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <ESP8266mDNS.h>
#include <ESP8266HTTPClient.h>
#include <TaskScheduler.h>
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>

#include <MQ135.h>

const char *ssid = "wifi-ssid";
const char *password = "password";
const char *urlServer = "http://hostname:port/service";
const char *roomName = "room1";

void jsonTaskCallback();
void serverTaskCallback();

Scheduler runner;
ESP8266WebServer server ( 80 );
HTTPClient http;
Task jsonTask(TASK_SECOND, TASK_FOREVER, &jsonTaskCallback);
Task serverTask(TASK_IMMEDIATE, TASK_FOREVER,
&serverTaskCallback);

#define DHTPIN 2
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);

#define MQ135PIN A0

```

```

MQ135 mq(MQ135PIN);

String getHttpData( String URI ) {
    http.begin( URI.c_str() );
    int code = http.GET();
    if(code > 0) {
        if( code == HTTP_CODE_OK ) {
            return http.getString();
        }
    }
    return "Error";
}

void sendPostData( String data ) {
    http.begin(urlServer);
    http.addHeader("Content-Type", "application/x-www-form-urlencoded");
    http.POST(data.c_str());
    http.writeToStream(&Serial);
    http.end();
}

void serverTaskCallback() {
    server.handleClient();
    yield();
}

void jsonTaskCallback() {
    Serial.println(millis());
    char tchar[6], hchar[6], co2char[6];
    char data[255];
    float h = dht.readHumidity();
    float t = dht.readTemperature();
    float co2ppm = mq.getPPM();
    dtostrf(h, 4, 2, hchar);
    dtostrf(t, 4, 2, tchar);
    dtostrf(co2ppm, 4, 2, co2char);

    StaticJsonBuffer<255> jsonBuffer;
    JsonObject& obj = jsonBuffer.createObject();
    obj["room"] = roomName;
    obj["humidity"] = hchar;
    obj["temperature"] = tchar;
    obj["co2"] = co2char;

    obj.printTo(data, sizeof(data));

    sendPostData(String(data) + String("\r\n\r\n"));
    yield();
}

void handleRoot() {

```

```

    char temp[700];
    char ipaddr[16];
    char clientaddr[16];
    IPAddress ip = WiFi.localIP();
    IPAddress ipcli = server.client().remoteIP();
    int sec = millis() / 1000;
    int min = sec / 60;
    int hr = min / 60;

    String ipJson = getHttpData( "http://api.ipify.org/?format=json"
);
    StaticJsonBuffer<200> jsonBuffer;
    JsonObject& root = jsonBuffer.parseObject( ipJson.c_str() );
    const char* extip = root[ "ip" ];

    snprintf( ipaddr, 16, "%d.%d.%d.%d", ip[0], ip[1], ip[2], ip[3]
);
    snprintf( clientaddr, 16, "%d.%d.%d.%d", ipcli[0], ipcli[1],
ipcli[2], ipcli[3] );

    snprintf ( temp, 600,
"<html>\
<head>\
  <meta http-equiv='refresh' content='5' />\
  <title>ESP8266 Demo</title>\
  <style>\
    body { background-color: #cccccc; font-family: Arial,
Helvetica, Sans-Serif; Color: #000088; }\
  </style>\
</head>\
<body>\
  <h1>Hello from ESP8266!</h1>\
  <p>Uptime: %02d:%02d:%02d</p>\
  <p>Your IP is: %s</p>\
  <p>Connected to: %s</p>\
  <p>IP Address: %s</p>\
  <p>External IP Address: %s</p>\
  <p>Signal strength: %d</p>\
</body>\
</html>",

        hr, min % 60, sec % 60, clientaddr, WiFi.SSID().c_str(),
ipaddr, extip, WiFi.RSSI()
    );
    server.send ( 200, "text/html", temp );
}

void handleNotFound() {
    String message = "File Not Found\n\n";
    message += "URI: ";
    message += server.uri();
    message += "\nMethod: ";

```

```

message += ( server.method() == HTTP_GET ) ? "GET" : "POST";
message += "\nArguments: ";
message += server.args();
message += "\n";

for ( uint8_t i = 0; i < server.args(); i++ ) {
    message += " " + server.argName ( i ) + ": " + server.arg ( i
) + "\n";
}

server.send ( 404, "text/plain", message );
}

void setup ( void ) {
    pinMode(D0, OUTPUT);
    digitalWrite(D0, HIGH);
    Serial.begin ( 115200 );
    WiFi.begin ( ssid, password );
    Serial.println ( "" );

    while ( WiFi.status() != WL_CONNECTED ) {
        delay ( 500 );
        Serial.print ( "." );
    }

    Serial.println ( "" );
    Serial.print ( "Connected to " );
    Serial.println ( ssid );
    Serial.print ( "IP address: " );
    Serial.println ( WiFi.localIP() );

    if ( MDNS.begin ( "esp8266" ) ) {
        Serial.println ( "MDNS responder started" );
    }

    server.on ( "/", handleRoot );
    server.onNotFound ( handleNotFound );
    server.begin();
    Serial.println ( "HTTP server started" );
    dht.begin();

    float rzero = mq.getRZero();
    delay(3000);
    Serial.print("MQ135 RZERO Calibration Value : ");
    Serial.println(rzero);

    runner.init();
    runner.addTask(jsonTask);
    runner.addTask(serverTask);
    serverTask.enable();
    jsonTask.enable();
}

```

```
void loop ( void ) {  
    runner.execute();  
}
```