МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Кафедра програмної інженерії

Звіт

З дисципліни «Аналіз та рефакторинг коду»

З лабораторної роботи №1

Виконавець:
ст. гр. ПЗПІ-22-2 Єременко. А.В

Перевірив:
доц.каф. ПІ Сокорчук І.П.

Харків 2024

Назва проекту: Програмна система для обліку використання жильцями внутрішньої інфраструктури житлового комплексу.

Посилання на відео: https://youtu.be/qeg9Hs2QV60

ХІД РОБОТИ

У рамках розробки програмної системи для обліку використання жильцями інфраструктури житлового комплексу виконано наступні етапи:

1. **Аналіз вимог та планування:** визначено ключовий функціонал, аналізовано ринок, сформульовано бачення і масштаб системи.
2. **Проєктування архітектури:** обрано мікросервісну архітектуру, визначено компоненти системи, обрано технології (Go, PostgreSQL, Kubernetes, React/Flutter).

ВИСНОВКИ

У результаті виконаної роботи була розроблена концепція програмної системи для обліку використання ресурсів в житлових комплексах. Система передбачає автоматизований збір даних з лічильників, розрахунок споживаних ресурсів та надання рекомендацій щодо оптимізації використання. Архітектура проєкту забезпечує масштабованість та ефективне управління даними, що дозволяє покращити контроль за споживанням енергоресурсів і забезпечити зручність для користувачів та адміністраторів. Наступним кроком є реалізація основних функціональних можливостей.

Додаток А

Vision and Scope Document

1. Передумови

1.1. Актуальність проблеми

У сучасних умовах урбанізації та розвитку багатоповерхових житлових комплексів, оптимізація використання ресурсів та внутрішньої інфраструктури стає важливим аспектом управління житловими будинками. Жильці стикаються з низкою викликів, зокрема:

- Нерозумінням реального споживання ресурсів (електроенергії, води, тепла) кожним домогосподарством.
- Складністю обліку витрат та прозорого розрахунку сум у платіжках.
- Відсутністю інструментів для аналізу та оптимізації використання ресурсів.
- Нездатністю ідентифікувати обладнання, яке споживає ресурси нераціонально.

Крім того, керуючі компанії стикаються з необхідністю автоматизації обліку даних, надання мешканцям зручних інструментів для перегляду статистики та генерації рекомендацій щодо оптимального використання ресурсів.

Розробка програмної системи для автоматизованого збору та аналізу даних дозволить знизити витрати, підвищити прозорість управління ресурсами, а також створити екологічно свідоме середовище.

1.2. Аналіз аналогічних рішень

На ринку вже існує декілька програмних рішень для управління житловими комплексами, проте більшість з них має вузьку спеціалізацію або не охоплює всі потреби сучасних житлових комплексів.

Розглянемо деякі з них:

1. **Landlord Studio**

Опис:

Landlord Studio розроблено для управління орендними об'єктами. Його простий інтерфейс надає основний набір інструментів, який підходить для невеликих орендодавців або початківців у сфері управління нерухомістю.

Ключові функції:

- Скринінг орендарів для перевірки платоспроможності.
- Відстеження оплат та управління фінансами.
- Базова звітність для аналізу доходів від оренди.

Недоліки:

- Вузька орієнтація лише на ринок оренди.
- Відсутність функцій для збору даних про використання ресурсів або їх аналізу.
- Немає інструментів для генерації рекомендацій щодо економії ресурсів чи управління внутрішньою інфраструктурою

Цільова аудиторія:
Маленькі приватні орендодавці.

2. **FRONTSTEPS**

Опис:

FRONTSTEPS — це спеціалізоване рішення для управління житловими об'єднаннями (HOA) та великими спільнотами. Його функціонал охоплює адміністрування житлових комплексів, але не включає можливостей аналізу споживання ресурсів.

Ключові функції:

- Управління спільними календарями для організації подій і зустрічей.
- Інструменти для комітетів: полегшують спілкування між членами спільноти.
- Управління робочими замовленнями, що підвищує ефективність виконання запитів мешканців.
- Інтеграція з платіжними системами для збору внесків HOA.

Недоліки:

- Сфокусоване лише на адміністративних функціях.
- Відсутність автоматизованого збору даних з лічильників або аналітики використання ресурсів.
- Обмежена персоналізація для індивідуальних домогосподарств.

Цільова аудиторія:
Великі житлові об'єднання та спільноти.

3. RealPage Commercial

Опис:

RealPage Commercial пропонує спеціалізовані рішення для управління комерційними об'єктами нерухомості. Його функціонал включає потужні інструменти автоматизації фінансів та адміністрування договорів оренди.

Ключові функції:

- Інструменти для автоматизації оренди, включаючи адміністрування договорів.
- Робочий портал для орендарів, що спрощує комунікацію.
- Розрахунки витрат на спільні площі та автоматизація платежів.
- Потужний бухгалтерський модуль для управління фінансами, включаючи облік штрафів, підвищення ставок та повторюваних платежів.

Недоліки:

- Фокус лише на комерційних об'єктах.
- Немає функціоналу для роботи з домогосподарствами або аналізу індивідуального споживання ресурсів.
- Відсутність рекомендацій щодо оптимізації використання ресурсів.

Цільова аудиторія:
Комерційна нерухомість (торгові центри, офіси тощо).

1.3. Способи монетизації

Для забезпечення фінансової ефективності проєкту запропоновано кілька способів монетизації програмної системи. Модель монетизації передбачає отримання доходу від різних сегментів користувачів і партнерів.

1. Підписка для користувачів (жильців):

Модель: Введення щомісячної або щорічної абонентської плати за доступ до розширених функцій.

Платні функції:

- Деталізовані звіти про споживання ресурсів.
- Інструменти для порівняння власного споживання з іншими домогосподарствами.
- Персоналізовані рекомендації щодо оптимізації споживання.

Ціновий діапазон: $2–$5 на місяць для індивідуальних домогосподарств.

2. Ліцензійні пакети для керуючих компаній:

Модель: Продаж ліцензій на використання системи для обслуговуючих компаній або об'єднань співвласників багатоквартирних будинків (ОСББ).

Особливості пакета:

- Кількість підключених домогосподарств.
- Функції аналітики для загального управління ресурсами комплексу.
- Інструменти для створення колективних звітів.

Ціновий діапазон: Від $50 до $500 на місяць залежно від кількості підключених об'єктів.

3. Модульна монетизація (додаткові платні функції):

Модель: Продаж окремих модулів, які користувачі або компанії можуть активувати за додаткову плату.

Приклади модулів:

- Автоматичні оповіщення про перевищення лімітів використання ресурсів.
- Інтеграція з "розумними" пристроями.
- API для підключення до сторонніх систем або платформ.
- Прогнозування витрат на основі історичних даних.

2. Vision of the Solution

2.1. Призначення системи

Програмна система для обліку використання жильцями внутрішньої інфраструктури житлового комплексу спрямована на створення інтегрованого рішення, яке забезпечить автоматизацію збору даних про споживання ресурсів (електроенергія, вода, тепло тощо), їх аналіз, формування звітів та надання рекомендацій для оптимізації витрат.

Основні цілі системи:

1. Прозорість: Надання жильцям та керуючим компаніям точних даних про використання ресурсів.
2. Ефективність: Забезпечення рекомендацій для раціонального використання ресурсів та зниження витрат.

3. Автоматизація: Мінімізація людського фактора в обліку ресурсів завдяки автоматизованому збору даних з лічильників.
4. Інтеграція:Підтримка роботи з існуючими системами житлових комплексів та "розумними" пристроями.
5. Екологічна відповідальність: Сприяння зменшенню енергоспоживання та впровадженню екологічно свідомої поведінки серед жильців.

Система орієнтована на дві основні групи користувачів:

- Жильці: Для аналізу особистих витрат на ресурси, перегляду звітів і отримання рекомендацій.
- Керуючі компанії: Для адміністрування житлового комплексу, контролю ресурсів та формування загальних звітів.

2.2. Опис бізнес-логіки системи

Система функціонує за наступним алгоритмом:

1. **Збір даних:**
   - Інтеграція з "розумними" лічильниками для збору інформації про споживання ресурсів кожним домогосподарством.
   - Регулярне оновлення даних у реальному часі або за заданим розкладом.
2. **Обробка даних:**
   - Обчислення загального обсягу використаних ресурсів.
   - Визначення питомих витрат для кожного домогосподарства.
   - Аналіз споживання за типами ресурсів та пристроїв.
3. **Аналіз і прогнозування:**
   - Використання алгоритмів аналізу даних для визначення трендів у споживанні.
   - Прогнозування майбутніх витрат на основі історичних даних.
   - Виявлення неефективного використання ресурсів (наприклад, старі або несправні прилади).
4. **Рекомендації:**
   - Генерація персоналізованих рекомендацій для жильців щодо оптимізації споживання ресурсів (наприклад, заміна неефективного приладу).
   - Надання загальних рекомендацій для керуючих компаній (наприклад, встановлення енергоефективного обладнання).
5. **Візуалізація:**
   - Інтуїтивно зрозумілий інтерфейс для перегляду статистики споживання, включаючи:
   - Графіки використання ресурсів.
   - Порівняння з середніми показниками по комплексу.
   - Дашборди для керуючих компаній із консолідованими даними.
6. **Інтеграція з платіжними системами:**
   - Автоматизація розрахунку рахунків за комунальні послуги на основі точних даних про споживання.

- Надання користувачам можливості сплачувати рахунки безпосередньо через платформу.

7. **Безпека та доступ:**
   - Впровадження багаторівневого доступу:
   - Жильці: доступ до персональних даних.
   - Керуючі компанії: доступ до загальних та індивідуальних даних.
   - Захист персональних даних згідно з міжнародними стандартами (GDPR та ін.).

Система орієнтована на інтуїтивність використання, а також забезпечує масштабованість, дозволяючи додавати нові модулі та функції відповідно до потреб користувачів.

3. Scope and Limitations

3.1. Реалізація, опис серверної частини

Серверна частина системи побудована за сучасними принципами мікросервісної архітектури, забезпечуючи масштабованість, надійність та ефективність обробки даних. Для цього система планується бути повністю побудована як **Cloud Native Application.** Ось її основні компоненти:

1. **Сервер основних функцій:**

- Мова програмування: Серверна логіка реалізована на Go (Golang), що забезпечує високу продуктивність і низьке споживання ресурсів.
- Протокол зв'язку: Сервер приймає HTTPS-запити від підключених пристроїв для забезпечення безпечної передачі даних.
- Масштабованість: Сервер розгортається в Kubernetes-кластері, що дозволяє автоматично масштабувати кількість інстансів залежно від навантаження.

2. **База даних:**

- Технологія: PostgreSQL — реляційна база даних для зберігання структурованих даних, таких як дані споживання ресурсів, історичні записи та аналітична інформація.
- Інтеграція: База даних розгорнута як окремий под у Kubernetes і підтримує спільний стан між кількома подами для забезпечення стійкості та доступності.
- Резервування: Впровадження реплікаційних механізмів для забезпечення стійкості до збоїв.

3. **API-сервер:**

- Мова програмування: Написаний також на Go для уніфікації серверної екосистеми.
- Призначення: Забезпечення комунікації між клієнтськими додатками та серверною частиною.

• Виконання аутентифікації користувачів, обробки запитів і повернення даних у зручному форматі.

3.2. Реалізація, опис клієнтської частини

Клієнтська частина системи складається з веб-додатка, що забезпечує доступ до функцій системи для різних груп користувачів.

1. **Веб-інтерфейс:**

   Технології:
   - React для створення інтерактивного користувацького інтерфейсу.
   - Material UI для сучасного дизайну.

   Функціонал:
   - Відображення персональної статистики споживання.
   - Графіки та аналітика даних.
   - Інструменти для налаштування сповіщень та отримання рекомендацій.

2. **Комунікація з API:**
   - Використання REST API для швидкої та ефективної передачі даних.

3.3. Обмеження та виключення

1. **Технічні обмеження:**

   - Підключення до системи можливе лише за умови наявності сумісних "розумних" лічильників або інших IoT-пристроїв.
   - Система залежить від наявності стабільного інтернет-з'єднання для збору даних.

2. **Функціональні виключення:**

   - Немає підтримки для аналізу ручного введення даних.
   - Не передбачено управління фізичною інфраструктурою (наприклад, дистанційне включення/вимкнення пристроїв).

3.4. Опис основних процесів системи

Реєстрація та автентифікація користувачів

1. **Реєстрація:**

   - **Флоу:**

1. Користувач переходить на сторінку реєстрації.
2. Вводить дані: ім’я, електронну пошту, пароль, номер квартири, інші деталі (за необхідності).
3. Система перевіряє унікальність введених даних (електронна пошта, номер квартири).
4. Після успішної валідації користувач отримує підтвердження електронною поштою.
5. Користувач активує обліковий запис, натискаючи на посилання у листі.

- **Обробка на сервері:**

  Дані передаються через HTTPS-запит на API-сервер.

  API-сервер перевіряє коректність та унікальність даних і зберігає їх у PostgreSQL через захищений ORM.

2. **Автентифікація:**

- **Флоу:**

  1. Користувач вводить електронну пошту та пароль у формі входу.
  2. Система перевіряє відповідність введених даних із записами в базі даних.
  3. У разі успіху генерується JWT-токен, який передається користувачу для авторизації в наступних запитах.

- **Обробка на сервері:**

  Запит на API-сервер із параметрами авторизації.

  API-сервер шифрує пароль (SHA-256) та перевіряє його в базі.

  У разі успіху генерує та відправляє токен.

Збір даних від пристроїв

1. **Флоу:**

• Підключені “розумні” пристрої (лічильники води, електрики, тепла) періодично відправляють дані через HTTPS-запити на сервер збору даних.

• Дані містять ідентифікатор пристрою, тип ресурсу, значення показників та часову мітку.

2. **Обробка на сервері:**

Сервер збору даних приймає HTTPS-запити та передає дані на обробку.

Дані перевіряються на валідність:

- Ідентифікатор пристрою перевіряється у базі.
- Часова мітка порівнюється із попередніми записами для уникнення дублювання.

3. **Збереження:**

- Дані обробляються мікросервісом аналізу та зберігаються у PostgreSQL.
- Середні та агреговані значення зберігаються для оптимізації запитів.

Обробка даних

1. **Флоу:**

- Сервіс обробки періодично отримує дані з черги.
- Виконує наступні завдання:
  - Розрахунок щоденного, щотижневого та щомісячного споживання.
  - Визначення аномалій (наприклад, різке збільшення витрат).
  - Генерація рекомендацій на основі трендів (наприклад, заміна енерговитратного приладу).

2. **Збереження:**

- Оброблені дані зберігаються в PostgreSQL у спеціалізованих таблицях для звітів і рекомендацій.
- Дашборди користувачів отримують доступ до цих даних через API-сервер.

Візуалізація та доступ до даних

1. **Флоу:**

- Користувач відкриває веб-інтерфейс.
- Надсилається запит до API-сервера із JWT-токеном для отримання персональних даних.
- Сервер виконує запит до бази даних для отримання необхідної інформації (статистика, графіки, рекомендації).

2. **Обробка:**

Дані агрегуються API-сервером у форматі, зручному для фронтенду (JSON).

Інтерфейс візуалізує дані у вигляді графіків, таблиць та рекомендацій.

3. **Реальний час:**

   Дані оновлюються через REST API для забезпечення інтерактивності (наприклад, моніторинг витрат у реальному часі).

Ці процеси забезпечують комплексну роботу системи, починаючи з реєстрації користувачів і закінчуючи наданням аналітики у зручній формі. Система побудована так, щоб бути масштабованою, ефективною та зручною для користувачів.

4. Operating Environment

4.1. Використані технології

1. **Серверна частина:**
   - Go (Golang) — для основної логіки та обробки запитів.
   - Kubernetes — для оркестрації контейнерів і масштабування системи.
   - PostgreSQL — для управління даними.
2. **API-сервер:**
   - Go — для реалізації REST API та аутентифікації.
3. **Клієнтська частина:**
   - React — для веб-додатка.
4. **Інфраструктура:**
   - Docker Desktop — для розгортання серверів і баз даних.
   - Docker — для контейнеризації додатків.
5. **Безпека:**
   - HTTPS для захищеного зв’язку.
   - OAuth 2.0 для авторизації та аутентифікації.