Plagiarism Scan Report

0% Plagiarism	0% Exact Match 0% Partial Match
100% Unique	

Words	966
Characters	7454
Sentences	112
Paragraphs	80
Read Time	5 minute(s)
Speak Time	7 minute(s)

Content Checked For Plagiarism

4.13 Метод перевірки подій

check_reminders — метод, який перевіряє події на наближення до них (15 хвилин до події).

- Для кожної події перевіряється, чи вона відбудеться найближчим часом.
- Якщо так, викликається метод send_reminder для відправки нагадування.

Рис.4.13.1 – Метод check_reminders

4.14 Метод відправки нагадувань та показу візуальних повідомлень

send_reminder — метод для відправки нагадувань користувачу через консоль і повідомлення в інтерфейсі.

Рис.4.14.1 – Метод send_reminder

Метод show_custom_message: це метод класу, який відповідає за показ візуального повідомлення у новому вікні.

Рис.4.14.2 – Метод show_custom_message

Рис.4.14.3 – Екземпляр та підключення до бази даних

Код реалізує систему інформування користувачів через графічний інтерфейс, дозволяючи додавати користувачів, відправляти повідомлення, створювати події та отримувати нагадування. Основна логіка реалізована через методи класу EventNotifierGUI, які взаємодіють з базою даних через об'єкт db_manager.

4.15 Створення та запуск основної програми з підключенням до бази даних

Здійснюється ініціалізація та запуск основної програми з використанням графічного інтерфейсу та підключенням до бази даних.

Рис.4.15.1 – Графічний інтерфейс введення подій та вибору подій

1. Імпорт необхідних модулів:
   - Спочатку імпортуються два важливих компоненти:
     - DatabaseManager з модуля database, який буде відповідати за роботу з базою даних.
     - EventNotifierGUI з модуля gui, який реалізує графічний інтерфейс програми.
   - Також імпортується модуль tkinter, який використовується для створення графічного інтерфейсу.
2. Функція main():
   - В функції main() створюється об'єкт db_manager класу DatabaseManager, який відповідатиме за взаємодію з базою даних.
   - Далі ініціалізується головне вікно програми через tk.Tk(), яке є основним елементом графічного інтерфейсу.
   - Створюється об'єкт app класу EventNotifierGUI, передаючи йому в якості параметрів головне вікно root та об'єкт для роботи з базою даних db_manager.
   - Викликається метод root.mainloop(), який запускає цикл подій для обробки взаємодії користувача з графічним інтерфейсом.
3. Перевірка запуску програми:

- Останній блок коду if __name__ == "__main__": перевіряє, чи є цей файл основним при виконанні програми. Якщо так, то запускається функція main(), що забезпечує ініціалізацію і запуск додатка.

5. КЕРІВНИЦТВО КОРИСТУВАЧА

Процес використання програми для інформування користувачів про події. Програма складається з графічного інтерфейсу, що надає можливість користувачам взаємодіяти з додатком для перегляду, додавання та керування подіями. Також передбачена можливість відправлення повідомлень користувачам через систему сповіщень.

1. Запуск програми:

- Для запуску програми користувач повинен запустити файл програми user_manager.py, де ініціалізується графічний інтерфейс і здійснюється підключення до бази даних. Після цього відкриється головне вікно програми.

2. Головне вікно:

Головне вікно містить наступні елементи:

- Список користувачів: тут відображається список всіх користувачів, для яких можна надіслати повідомлення.
- Канали зв'язку: у цьому розділі можна вибрати канал зв'язку для сповіщення користувачів при додаванні користувача (наприклад, через email або інші засоби зв'язку).
- Кнопка для відправлення повідомлень: користувач може натискати на кнопку для відправлення повідомлень обраним користувачам.

Рис.5.1 – Інтерфейс відкритого застосунку

Рис.5.2 – Додавання нового користувача та вибір зв’язку з користувачем

Рис.5.3 – Вивід створених користувачів

3. Додавання нових подій:

- Користувач може додавати нові події за допомогою інтерфейсу, заповнивши відповідні поля (наприклад, дата, час, опис події).
- Після заповнення форми для додавання події, інформація зберігається в базі даних і стає доступною для перегляду та сповіщення користувачів.

Рис.5.4 – Створення нової події

Рис.5.5 – Повідомлення до користувача

4. Виведення повідомлень:

- Програма надає можливість відправлення кастомізованих повідомлень через спливаючі вікна, що з'являються на екрані користувача. Повідомлення можуть бути різних кольорів, щоб привертати увагу користувачів.
- У кожному вікні є кнопка для закриття сповіщення.

Рис.5.6 – Повідомлення про додавання користувача

Рис.5.7 – Повідомлення про додавання події

Рис.5.8 – Повідомлення про відправлене повідомлення

Рис.5.9 – Повідомлення про видалення користувача

Рис.5.10 – Повідомлення про необхідність заповнити всі поля

Рис.5.11 – Повідомлення про необхідність обрати користувача зі списку

Рис.5.12 – Повідомлення про необхідність правильно написати формат часу

Рис.5.13 – Повідомлення про необхідність ввести повідомлення

5. Перегляд інформації про події:

- Користувач може переглядати список подій, що зберігаються в базі даних.
- Інтерфейс дозволяє фільтрувати події за різними критеріями (наприклад, по даті або типу події).

Рис.5.14 – Перегляд айді, користувача айді, події, дня, часу та опису події

Рис.5.15– Перегляд айді, користувача айді та повідомлення

Рис.5.16– Перегляд айді, ім’я користувача та метод зв’язку

6. Закриття програми:

- Для закриття програми користувач може натискати кнопку "Закрити" в головному вікні або закрити вікно через стандартні засоби операційної системи.

7. Помилки та обробка виключень:

- Якщо при роботі з базою даних або при додаванні нових подій виникають помилки, програма відображає повідомлення про помилку у вигляді спливаючих вікон, що містять опис помилки.
- Програма обробляє основні помилки при підключенні до бази даних і забезпечує користувача необхідною інформацією для вирішення проблем.

8. Особливості інтерфейсу:

- Інтерфейс користувача простий та зрозумілий, з усіма основними функціями, доступними на головному екрані.
- Кожна функціональність має чітке і зрозуміле позначення, що дозволяє користувачеві швидко освоїтися в програмі.

Програма забезпечує зручну систему для управління подіями та сповіщеннями, дозволяючи користувачам ефективно управляти інформацією та взаємодіяти з іншими користувачами через зручний графічний інтерфейс.

6. ВИСНОВКИ

Метою заданого курсового проекту було створення програмного додатку для інформування користувачів. Роботу було розділено на наступні етапи для чіткого розуміння поставленої задачі:

- Аналіз задачі та постановка цілей завдання;
- Проектування архітектури додатку та бази даних;
- Розробка програмного алгоритму роботи додатку;
- Створення графічного інтерфейсу та реалізація функціоналу;

- Тестування та налагодження програми.

Робота розпочалася з аналізу основних вимог до додатку та визначення функцій, які необхідно реалізувати, таких як управління користувачами, подіями та каналами зв'язку. Було обрано мову програмування Python, оскільки вона забезпечує високу гнучкість і швидкість розробки, а також підтримує численні бібліотеки для роботи з графічними інтерфейсами та базами даних.

Програма включає функцію відправки повідомлень користувачам, що робить її корисною для компаній, що займаються розсилкою інформації. Крім того, було реалізовано кастомізовані повідомлення, які дозволяють виводити різні типи інформаційних вікон для покращення взаємодії з користувачем.

Таким чином, в результаті виконання курсової роботи, я навчився проектувати програмні додатки, працювати з базами даних, створювати графічні інтерфейси та інтегрувати різні компоненти програми. Результатом виконаної роботи став функціональний додаток для інформування користувачів, який може бути використаний в різних сферах для полегшення управління інформацією та комунікацією.

Matched Source

No plagiarism found