ДОВІДКА

про перевірку курсової роботи (проекту) на наявність плагіату

Відділення: *Автоматизації та залізничного транспорту*

Спеціальність: *5.151.1 Обслуговування інтелектуальних інтегрованих систем*

Освітньо-професійна програма: *151 Автоматизація та комп'ютерно-інтегровані технології*

Дисципліна: *Програмування*

Автор роботи: *Федулов Ілля Юрійович*

Назва роботи: *Розробка програмного додатку для інформування користувачів.*

Керівник: *Васильєв Микола Васильович*

Обсяг матеріалів, сторінок *59*

Обсяг матеріалів, який перевірявся на оригінальність, сторінок *39*

Результати перевірки: робота перевірена програмою *https://www.duplichecker.com*, відсоток унікальності становить *99%*.

Студент Ілля ФЕДУЛОВ

Plagiarism Scan Report

2% Plagiarism

2% Exact Match

0% Partial Match

98% Unique

Words	997
Characters	7617
Sentences	81
Paragraphs	66
Read Time	5 minute(s)
Speak Time	7 minute(s)

Content Checked For Plagiarism

ЗМІСТ

1. ВСТУП

Останнім часом інформаційні технології стали невід'ємною частиною повсякденного життя, і їх застосування охоплює всі сфери діяльності. Усе більша кількість компаній і організацій використовує програмне забезпечення для

оптимізації комунікацій, підвищення ефективності взаємодії з клієнтами, співробітниками та партнерами. Одним із ключових аспектів є інформування користувачів про важливу інформацію, що дозволяє своєчасно донести необхідні дані та забезпечити належний рівень комунікації.

Метою цього проекту є розробка програмного додатку для інформування користувачів через різні канали зв'язку. Додаток буде надавати можливість зберігати дані про користувачів, визначати канали зв'язку з ними та надсилати повідомлення через ці канали. Програма забезпечить автоматичне відправлення повідомлень користувачам, що значно спростить процес комунікації та допоможе своєчасно доставляти важливу інформацію.

Актуальність цього проекту зумовлена потребою в ефективному управлінні комунікаціями, що особливо важливо для великих організацій, де інформація повинна бути доставлена до багатьох осіб швидко і без помилок. Використання бази даних для збереження інформації дозволить забезпечити надійний доступ до даних і зручне управління ними.

Метою даного проекту є створення зручного та ефективного інструменту для автоматизації процесу інформування користувачів, що допоможе покращити організацію комунікацій і забезпечить своєчасне доставлення важливих повідомлень.

2. АНАЛІЗ ЗАДАЧІ, ЗАСОБІВ ТА МЕТОДІВ ЇЇ ВИРІШЕННЯ

В умовах постійного розвитку інформаційних технологій одним із важливих аспектів у сучасному світі є забезпечення ефективної комунікації між користувачами. Зокрема, у великих організаціях, де кількість співробітників або користувачів постійно зростає, важливо забезпечити своєчасне інформування через різні канали зв'язку. Під час виконання даного проекту основною задачею є розробка програмного забезпечення, яке дозволить автоматизувати процес інформування користувачів через канали зв'язку.

Задача полягає у створенні програмного додатку, який буде зберігати інформацію про користувачів, їх канали зв'язку та надавати можливість відправлення повідомлень через ці канали. Оскільки ефективність комунікацій є важливою складовою в роботі організацій, створення такого інструменту дозволить оптимізувати процеси взаємодії між користувачами та забезпечить своєчасне та точне інформування.

Основними вимогами до програмного забезпечення є:

- Збереження та організація даних про користувачів.
- Підтримка кількох каналів зв'язку (електронна пошта, SMS, месенджери тощо).
- Можливість відправлення повідомлень на визначені канали зв'язку.
- Надійність і швидкість доступу до даних.
- Інтерфейс, що дозволяє зручно управляти користувачами та їх даними.

Засоби вирішення задачі:

Для вирішення поставленої задачі будуть використані сучасні програмні засоби та технології:

- Мова програмування Python: вона є зручним інструментом для розробки таких додатків завдяки великій кількості бібліотек, зокрема для роботи з базами даних, управління користувачами та канали зв'язку.
- Tkinter: бібліотека для створення графічного інтерфейсу користувача (GUI), що забезпечить зручне взаємодія користувача з додатком.
- База даних SQL Server: для зберігання інформації про користувачів, їх канали зв'язку та відправлені повідомлення. Використання реляційної бази даних забезпечить надійність та швидкість доступу до даних.
- pyodbc: бібліотека для роботи з SQL Server через Python, що дозволить зручно виконувати операції з базою даних.

Методи вирішення задачі:

Для реалізації програмного забезпечення будуть використані наступні методи:

1. Модульне програмування: розробка окремих модулів для управління базою даних, інтерфейсом користувача та процесом відправлення повідомлень.
2. Інтерфейс користувача: створення графічного інтерфейсу, що забезпечить простоту взаємодії з додатком та зручне управління даними.
3. Автоматизація процесу інформування: реалізація механізму автоматичного відправлення повідомлень користувачам через вибрані канали зв'язку.
4. Оптимізація доступу до даних: використання реляційної бази даних дозволить швидко здійснювати запити та маніпулювати інформацією про користувачів і повідомлення.

2.1 Python

Python є однією з найпопулярніших мов програмування на сьогоднішній день. Вона відома своєю простотою та зручністю для початківців, а також потужністю та гнучкістю, що дозволяє її використовувати для різноманітних завдань. Python активно застосовується у розробці програмного забезпечення, аналізі даних, автоматизації

процесів, створенні веб-додатків, а також в інших сферах, таких як машинне навчання, штучний інтелект та багато іншого.

Однією з основних причин вибору Python для цього проекту є його зручність у розробці та великий набір бібліотек, які значно спрощують виконання завдань, таких як робота з базами даних, побудова графічного інтерфейсу та інтеграція з іншими технологіями.

Основні переваги Python для реалізації проекту:

1. Простота синтаксису: python має дуже чистий і зрозумілий синтаксис, що дозволяє швидко освоїти мову і зосередитись на вирішенні завдань, а не на складних деталях реалізації. Це особливо важливо під час розробки програм, де важлива швидка і ефективна реалізація ітерацій.
2. Широкий набір бібліотек: python має багатий набір бібліотек, які дозволяють легко вирішувати конкретні завдання без необхідності розробляти складні алгоритми з нуля. Наприклад, для роботи з базами даних буде використано бібліотеку pyodbc, що спрощує процес взаємодії з SQL Server. Для створення графічного інтерфейсу користувача буде використана бібліотека Tkinter.
3. Міжплатформність: python є кросплатформною мовою, що означає, що програми, написані на Python, можуть працювати на різних операційних системах (Windows, Linux, macOS) без значних змін у коді. Це дозволяє забезпечити широку сумісність програми, що є важливим для подальшого використання програмного продукту на різних пристроях.
4. Швидка розробка: завдяки своїй простоті та багатим бібліотекам Python дозволяє скоротити час розробки програмного забезпечення, що особливо важливо в умовах обмежених термінів. У поєднанні з можливістю швидко тестувати та відлагоджувати програму, Python є оптимальним вибором для розробки.
5. Гнучкість та потужність: python є достатньо потужним для вирішення складних завдань. Зокрема, за допомогою бібліотек для роботи з базами даних (наприклад, pyodbc) можна ефективно взаємодіяти з SQL Server, а бібліотеки для роботи з мережевими протоколами дозволяють

Matched Source

Similarity 9%

Title:Що таке Python і чому він такий популярний?

Nov 1, 2024 · Python є однією з найпопулярніших мов програмування на сьогоднішній день, і це пояснюється його великою універсальністю та легкістю вивчення.

https://informatecdigital.com/uk/%D0%A9%D0%BE-%D1%82%D0%B0%D0%BA%D0%B5-Python-%D1%96-%D1%87%D0%BE%D0%BC%D1%83-%D0%B2%D1%96%D0%BD-%D1%82%D0%B0%D0%BA%D0%B8%D0%B9-%D0%BF%D0%BE%D0%BF%D1%83%D0%BB%D1%8F%D1%80%D0%BD%D0%B8%D0%B9%3F

реалізувати функціонал для відправки повідомлень через різні канали зв'язку (SMS, електронна пошта, месенджери).

6. Підтримка об'єктно-орієнтованого програмування (ООП): python підтримує принципи об'єктно-орієнтованого програмування, що дозволяє структурувати код і зробити його більш зручним для подальшого розширення та супроводження. Це особливо важливо при розробці програмного забезпечення, яке повинно бути масштабованим і підтримувати додаткові функціональні можливості в майбутньому.

Рис.2.1.1 – Логотип Python

2.2 Історія створення мови програмування Python

Мова програмування Python була створена у кінці 1980-х років голландським програмістом Гвідо ван Россумом (Guido van Rossum). Офіційний реліз першої версії відбувся 20 лютого 1991 року. Основною метою розробки було створення простої, читабельної та гнучкої мови програмування, яка б забезпечувала високу продуктивність і зручність використання.

Гвідо ван Россум працював у Центрі математики та інформатики (CWI) в Нідерландах і був залучений до розробки мови програмування ABC. Ця мова мала простий синтаксис, але мала певні обмеження, які заважали її широкому поширенню. Надихнувшись її ідеями та бажаючи створити більш потужний та універсальний інструмент, ван Россум розпочав роботу над новою мовою.

Python отримав свою назву не на честь змії, а завдяки британському комедійному шоу "Monty Python's Flying Circus", яке було популярним у той час. Ван Россум хотів, щоб його мова була простою, зрозумілою та, водночас, приносила задоволення від програмування.

Основні етапи розвитку Python:

1. Python 1.0 (1991 рік)
- Перша офіційна версія Python 1.0 була випущена у 1991 році.
- Вона вже містила основні особливості, які відрізняли її від інших мов:
  динамічну типізацію,
  автоматичне керування пам'яттю (збирання сміття),
  модульність (підтримку бібліотек),
  простий та читабельний синтаксис.

2. Python 2.x (2000 рік)
- У 2000 році вийшла версія Python 2.0, яка принесла важливі нововведення:
  підтримку списокових виразів (list comprehensions),
  покращене збирання сміття,
  нові можливості роботи з Unicode.
- Python 2.x став дуже популярним, і хоча його офіційна підтримка завершилася у 2020 році, багато компаній та програмістів використовували його протягом двох десятиліть.

3. Python 3.x (2008 – сьогодні)
- У 2008 році була випущена версія Python 3.0, яка внесла значні зміни:
  покращена підтримка Unicode,
  новий підхід до розділення цілочисельного та дійсного ділення (/ та //),
  оновлення стандартної бібліотеки.
- Python 3.x продовжує активно розвиватися, додаючи нові можливості та покращуючи продуктивність.

Рис.2.3.1 – Середовище розробки Visual Studio

2.4 База даних SQL Server

SQL Server — це реляційна система управління базами даних (СУБД), розроблена компанією Microsoft. Вона використовується для зберігання, управління та обробки великих обсягів даних у різних додатках, від корпоративних систем до веб-додатків та мобільних сервісів.

SQL Server був створений у 1989 році компаніями Microsoft, Sybase і Ashton-Tate. Перші версії були розроблені для OS/2, а згодом Microsoft почала розвивати його як продукт для Windows.

Офіційно Microsoft SQL Server 6.0 (1995 р.) став першою повністю самостійною версією компанії Microsoft. Відтоді SQL Server постійно оновлювався, отримуючи нові функції, покращену продуктивність та безпеку.

Сучасні версії, такі як SQL Server 2019 та SQL Server 2022, підтримують роботу з великими даними, штучним інтелектом та хмарними технологіями.

Microsoft SQL Server є потужною, надійною та масштабованою базою даних, яка підходить для реалізації різних інформаційних систем. Використання SQL Server у нашому проекті забезпечить ефективне управління даними користувачів, повідомленнями та каналами зв'язку.

Рис.2.4.1 – База даних EventNotifierApp SQL Server

3. ПРОЕКТУВАННЯ ЗАГАЛЬНОГО АЛГОРИТМУ РОБОТИ ПРОГРАМИ

3.1 Загальна структура програми

Розроблений програмний застосунок призначений для інформування користувачів шляхом надсилання повідомлень через різні канали зв'язку. Його основна функціональність включає:

- збереження списку користувачів у базі даних,
- керування каналами зв'язку (електронна пошта, месенджери тощо),
- можливість відправлення повідомлень користувачам,
- перегляд історії повідомлень та їх статусів.

Програма складається з трьох основних модулів:

1. Модуль управління базою даних (database.py) – збереження користувачів, контактної інформації та історії повідомлень.
2. Модуль запуску програми (user_manager.py) – взаємодія з іншими модулями та запуск програми.
3. Модуль інтерфейсу (gui.py) – графічний інтерфейс для взаємодії з користувачем.

3.2 Алгоритм роботи програми

Робота застосунку базується на наступному алгоритмі:

Запуск програми:

- Ініціалізація підключення до бази даних SQL Server.
- Завантаження списку користувачів та їх каналів зв'язку.
- Відображення головного вікна інтерфейсу.

Робота з користувачами:

- Додавання нового користувача до бази даних.
- Редагування або видалення існуючих користувачів.
- Прив'язка каналів зв'язку (email, Telegram, SMS тощо).

Формування та відправлення повідомлення:

- Вибір користувача або групи користувачів.
- Визначення каналу зв'язку.
- Введення тексту повідомлення.
- Збереження статусу повідомлення у базі даних.

Перегляд історії повідомлень:

- Відображення списку надісланих повідомлень.

Завершення роботи програми:

- Закриття підключення до бази даних.
- Завершення виконання процесів.

Рис.3.2.1 – Алгоритм роботи застосунку від початку до дій користувача

Рис.3.2.2 – Алгоритм роботи завдання "Вибір користувача зі списку"

Рис.3.2.3 – Алгоритм роботи завдання "Додавання користувача"

Рис.3.2.4 – Алгоритм роботи завдання "Видалення користувача"

Рис.3.2.5 – Алгоритм роботи завдання "Створення події"

Рис.3.2.6 – Алгоритм роботи завдання "Перевірка нагадувань"

Проектування алгоритму є важливим етапом розробки програмного забезпечення, що дозволяє структурувати логіку

роботи та оптимізувати процеси. Запропонована схема забезпечує гнучкість, надійність та зручність у використанні, а також гарантує ефективне інформування користувачів через різні канали зв'язку.

4. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Розробка програмного забезпечення для інформування користувачів включає кілька основних етапів: створення бази даних, реалізацію основних модулів та розробку графічного інтерфейсу.

4.1 Створення бази даних

База даних є ключовим компонентом програмного додатку, оскільки вона відповідає за збереження інформації про користувачів, повідомлення та події. Для реалізації роботи з базою даних у даному проекті використовується SQL Server, а взаємодія з нею здійснюється за допомогою бібліотеки pyodbc.

Бібліотека pyodbc використовується для підключення Python-додатку до SQL Server та виконання SQL-запитів. Клас DatabaseManager відповідає за підключення до бази даних, створення необхідних таблиць та виконання SQL-запитів.

Даний клас інкапсулює основні операції роботи з базою даних, що забезпечує зручність її використання в програмному коді.

4.2 Метод з'єднання з базою даних

Призначення методу __init__:

- Встановлює з'єднання з SQL Server через ODBC Driver 17.
- Підключається до бази даних EventNotifierApp.
- Використовує захищене підключення (Trusted Connection).
- Якщо підключення успішне, створюється об'єкт cursor, який дозволяє виконувати SQL-запити.
- Викликає метод create_tables(), який перевіряє наявність необхідних таблиць та створює їх у разі потреби.
- Якщо підключення не вдалося, програма генерує виняток і повідомляє про помилку.

Matched Source

Similarity 15%

Title:Microsoft SQL Server 2016 - Windows, Office, Server

Original-Lizenzen blitzschnell & sicher erhalten. Microsoft SQL Server 2016 Vollversion. Echte Lizenzen zum Sparpreis. Große A

https://www.bing.com/aclick?ld=e8MmaPPg_e6rgBTK7vEuYhuDVUCUwWMgef-PAD7Tg3E-NIJeiJuEEuOXwWBNUJ44FiVo-F5eX6elRGLvmlMIoolz7aqMefEpuiRe4yzwmjw37YCRqzKBsuuii5bo9rgTibPsRmaKsdUrLKJleGW69zKimyGGd0JhAepECo6hVLdT

Plagiarism Scan Report

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

Words	956
Characters	7123
Sentences	132
Paragraphs	96
Read Time	5 minute(s)
Speak Time	7 minute(s)

Content Checked For Plagiarism

Рис.4.2.1 – Метод __init__

4.3 Метод перевірки таблиць

Призначення методу create_tables:

- Метод перевіряє, чи існують таблиці users, messages, events, і створює їх, якщо вони відсутні.
- Використовується SQL-запит IF NOT EXISTS, щоб уникнути дублювання таблиць.
- users – містить дані про користувачів (ім'я, email, телефон, спосіб зв'язку).
- messages – зберігає повідомлення для користувачів. Має зовнішній ключ user_id, який посилається на користувача у таблиці users.
- events – містить записи про події, збережені користувачами. Також пов'язана з users через зовнішній ключ user_id.
- Усі запити виконуються у циклі, щоб кожна таблиця перевірялася та створювалася окремо.
- Викликається commit(), щоб зберегти зміни у базі.

Рис.4.3.1 – Метод create_tables

4.4 Метод виконання SQL-запитів

Призначення методу execute_query:

- Загальний метод для виконання SQL-запитів.
- query – SQL-запит, який потрібно виконати.
- params – параметри, які передаються у запит.
- fetch=False – якщо встановлено True, метод поверне результат виконання запиту (fetchall()).
- Використовується контекстний менеджер with, який автоматично закриває курсор після виконання операції.

Рис.4.4.1 – Метод execute_query

Розроблений модуль для роботи з базою даних забезпечує:

1. Безпечне підключення до SQL Server.
2. Автоматичне створення таблиць, якщо вони відсутні.
3. Гнучке виконання SQL-запитів через метод execute_query.
4. Інтеграцію з іншими модулями додатку для роботи з користувачами, повідомленнями та подіями.

Ця реалізація гарантує надійність та ефективність збереження та обробки даних у програмному додатку для інформування користувачів.

4.5 Розробка графічного інтерфейсу користувача та взаємодія з базою даних

1. Створення інтерфейсу програми: розробив візуальну частину програми, використовуючи бібліотеку Tkinter. Це включає:

- Головне вікно з кнопками для різних дій (наприклад, додавання, видалення користувачів, відправка повідомлень).
- Використання віджетів Tkinter (кнопки, мітки, текстові поля) для створення інтерфейсу, де користувач може взаємодіяти з програмою.

- Вбудований календар (за допомогою tkcalendar) для вибору дати події.

2. Завантаження та управління користувачами: налаштував механізм для завантаження даних користувачів з бази даних і відображення їх в інтерфейсі. Користувачі можуть бути додані або видалені через інтерфейс програми.
- Для цього використовувався Listbox для відображення списку користувачів.
- Для роботи з даними була використана база даних SQL Server, до якої ми підключалися через pyodbc.

3. Відправка повідомлень користувачеві: в програмі реалізована можливість відправки повідомлень конкретним користувачам через графічний інтерфейс:
- Користувач вибирає зі списку потрібного отримувача і вводить текст повідомлення.
- Повідомлення надсилається в базу даних і зберігається для обраного користувача.

4. Створення та управління подіями: реалізував функціонал для створення подій, які прив'язуються до користувачів:
- Користувач вибирає дату з календаря і створює подію з назвою, часом та описом.
- Ці події зберігаються в базі даних і можуть бути використані для подальшого інформування користувачів.

5. Потокова перевірка нагадувань: в окремому потоці реалізована перевірка подій на їх наближення (в межах 15 хвилин). Коли настає час події, користувачу надсилається нагадування.
- Це нагадування виводиться в консоль і відображається в графічному інтерфейсі як спливаюче вікно.

6. Використання багатозадачності: для того, щоб програма могла працювати з нагадуваннями без блокування інтерфейсу, використовував багатозадачність за допомогою бібліотеки threading, що дозволяє перевіряти нагадування у фоновому режимі, не заважаючи роботі користувача з інтерфейсом.

4.6 Імпорти та клас

Імпорти та клас:

1. tkinter — бібліотека для створення графічного інтерфейсу користувача (GUI) в Python.
- tk — стандартний модуль для роботи з GUI.
- messagebox — використовують для відображення стандартних діалогових вікон.
- simpledialog — для введення даних через прості діалогові вікна.

2. tkcalendar — використовується для створення календаря в інтерфейсі для вибору дати події.
3. threading — використовується для створення фонових потоків, щоб виконувати перевірку нагадувань у фоновому режимі.
4. time — використовується для встановлення затримок між перевірками подій (кожну хвилину).
5. datetime — для роботи з датами та часом, перевірка та обробка подій.
6. DatabaseManager — клас, який відповідає за взаємодію з базою даних (підключення, запити тощо).

Рис.4.6.1 – Імпорти та клас

4.7 Клас EventNotifierGUI

Клас EventNotifierGUI – цей клас визначає GUI для програми і реалізує основні функціональні можливості.

__init__ — конструктор класу:
- root — це основне вікно програми, створене через tk.Tk().
- db_manager — об'єкт для роботи з базою даних.
- Налаштування заголовку вікна та розміру.
- Виклик методів create_gui для створення інтерфейсу та load_users для завантаження даних про користувачів із бази.
- Створення потоку для перевірки нагадувань, який працює у фоновому режимі та викликає метод check_reminders.

Рис.4.7.1 – Ініціалізація та налаштування інтерфейсу

4.8 Метод всіх елементів інтерфейсу

create_gui — метод для створення всіх елементів інтерфейсу:
- Використовуються віджети, такі як Label, Listbox, Text, Button для різних компонентів (список користувачів, введення повідомлення, кнопки для додавання/видалення користувачів, створення подій).
- Календар, на якому користувач може вибрати дату події.
- Кожна кнопка викликає відповідний метод для додавання користувача, видалення користувача, або створення події.

Рис.4.8.1 – Створення графічного інтерфейсу

4.9 Метод завантаження даних користувачів

load_users — метод для завантаження даних користувачів з бази даних:

- Очистка списку користувачів в GUI.
- Виконується запит до бази даних для отримання користувачів.
- Дані кожного користувача виводяться в форматі: "Ім'я (метод зв'язку: email/телефон)".
- Збереження даних про користувачів у словнику self.user_data для подальшого використання (ідентифікація користувачів).

Рис.4.9.1 – Завантаження користувачів із бази даних

4.10 Метод відправки повідомлень

send_message — метод для відправки повідомлення обраному користувачу:

- Перевіряється, чи користувач вибраний в списку.
- Береться текст повідомлення з текстового поля.
- Якщо повідомлення не введено, виводиться повідомлення про помилку.
- Відправка повідомлення до бази даних для збереження.

Рис.4.10.1 – Відправка повідомлення

4.11 Методи додавання та видалення користувачів

Додавання та видалення користувачів

Методи add_user і delete_user відповідають за додавання та видалення користувачів з бази даних через відповідні вікна для введення даних або вибору користувача для видалення.

Рис.4.11.1 – Додавання користувача

Рис.4.11.2 – Видалення користувача

4.12 Метод створення, вспливаючого вікна та збереження події

create_event — метод для створення події:

- Перевірка на вибір користувача.
- Вибір дати з календаря.
- Перевірка правильності формату дати.

Рис.4.12.1 – Створення події

Рис.4.12.2 – Створення вікна для введення події, часу та опису події

Рис.4.12.3 – Збереження вікна для створення події

Matched Source

No plagiarism found

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 екран
- 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