

Міністерство освіти і науки України  
Харківський національний університет радіоелектроніки

Кафедра програмної інженерії

КУРСОВА РОБОТА  
ПОЯСНЮВАЛЬНА ЗАПИСКА  
з дисципліни “Об’єктно-орієнтоване програмування”  
«Щоденник»

Керівник, ст. викл.

Черепанова Ю.Ю.

Студент гр. ПЗП-23-2

Колодіюк Н.С.

Комісія:

Проф.

\_\_\_\_\_ Бондарєв В.М.

Ст. викл.

\_\_\_\_\_ Черепанова Ю.Ю.

Ст. викл.

\_\_\_\_\_ Ляпота В.М.

## ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Кафедра	<i>програмної інженерії</i>
Рівень вищої освіти	<i>перший (бакалаврський)</i>
Дисципліна	<i>Об'єктно-орієнтоване програмування</i>
Спеціальність	<i>121 Інженерія програмного забезпечення</i>
Освітня програма	<i>Програмна інженерія</i>

Курс 1Група ПЗПІ-23-2Семестр 2

## ЗАВДАННЯ

на курсовий проект студента

Колодіюк Надії Сергіївни

1 Тема проекту:

«Щоденник»2 Термін здачі студентом закінченого проекту: **“08” - червня - 2024 р.**

3 Вихідні дані до проекту:

Завдання на курсову роботу

4 Зміст розрахунково-пояснювальної записки:

Вступ, опис вимог, проектування програми, інструкція користувача,  
висновки

## КАЛЕНДАРНИЙ ПЛАН

<i>№</i>	<i>Назва етапу</i>	<i>Термін виконання</i>
1	Видача теми, узгодження і затвердження теми	13.02.2024 - 15.03.2024 р.
2	Формулювання вимог до програми	18.03.2024 – 19.03.2024 р.
3	Розробка підсистеми зберігання та пошуку даних	01.04.2024 – 10.04.2024 р.
4	Розробка функцій ...	15.04.2024 – 26.04.2024 р.
5	Розробка функцій зберігання та завантаження даних	27.04.2024 – 07.05.2024 р.
6	Тестування і доопрацювання розробленої програмної системи	10.05.2024 – 15.05.2024 р.
7	Оформлення пояснювальної записки, додатків, графічного матеріалу	15.05.2024 – 28.05.2024 р.
8	Захист	03.06.2024 – 08.06.2024 р.

Студент \_\_\_\_\_

Керівник \_\_\_\_\_

Черепанова Ю.Ю.

« 21 » лютого \_\_\_\_\_ 2024 р.

## РЕФЕРАТ

Пояснювальна записка до курсової роботи: 44 с., 20 рис., 1 дод., 5 джерел.

ЗАХОДИ, ЗВІТ, МОВА C#, ООП, СПРАВИ, ЩОДЕННИК, .NET

Метою роботи є створення програми «Щоденник», що надаватиме користувачеві можливість ефективно керувати своїм часом.

У результаті роботи було розроблено програму, що надає користувачеві можливість зручного зберігання записів про справи.

У результаті створена програма, що дозволяє зберігати списки справ. Справи мають наступні характеристики: назва, дата час, тривалість, місце проведення, короткий опис, категорія. Користувач може додавати, змінювати та видаляти існуючі справи. Програма надає можливість зручно переглядати справи та шукати їх за фільтрами, зберігати списки справ у текстовому форматі, в тому числі відфільтровані. Також програма надає можливість встановити автоматичні нагадування про справи і переглядати деяку статистику.

В процесі розробки були використані середовище розробки JetBrains Rider, фреймворки Avalonia та ReactiveUI, бібліотека libnotify, платформа .NET 8.0, мова програмування C#, система контролю версій Git.

## ЗМІСТ

Вступ.....	6
1 Опис вимог.....	7
1.1 Сценарії роботи програми.....	7
1.2 Функціонал програми.....	13
2 Проектування програми .....	24
2.1 Вибір архітектури проекту.....	24
2.2 Загальна структура програми .....	25
2.2.1 Моделі даних .....	26
2.2.2 Моделі вигляду.....	28
2.3 Формат зберігання даних .....	33
3 Інструкція користувача.....	34
3.1 Розгортання програми .....	34
3.2 Користування програмою.....	34
3.2.1 Навігація у програмі .....	34
3.2.2 Користування функціями, що надає головне вікно програми .....	35
3.2.3 Користування функціями створення, редагування і видалення.....	39
3.3 Видалення програми.....	41
Висновки.....	42
Перелік джерел посилання.....	43
Додаток А Код програми .....	44

## ВСТУП

У сучасному світі інформаційних технологій традиційні способи управління завданнями та справами поступово поступаються місцем електронним застосункам. Створення програмного забезпечення для організації особистих справ є актуальним завданням, оскільки воно забезпечує зручність, ефективність та багатофункціональність у веденні щоденних справ. Метою цієї курсової роботи є розробка функціонального та інтуїтивно зрозумілого програмного забезпечення, призначеного для збереження та управління запланованими заходами та справами, який би враховував потреби сучасних користувачів і пропонував зручні інструменти для керування їх розкладом.

Дане програмне забезпечення реалізує низку важливих сценаріїв використання, таких як додавання нових справ, перегляд збережених справ за датами та іншими фільтрами, редагування та видалення справ, налаштування автоматичних нагадувань, збереження списків у текстовому форматі та інше. Основною аудиторією програми є люди, які прагнуть організувати свій час та ефективно керувати своїм часом. Це можуть бути професіонали, студенти, звичайні домогосподарки та всі, хто потребує чіткого планування свого дня. Кожна з цих груп користувачів має свої специфічні потреби, які можна задовольнити за допомогою функціоналу програми.

Використання даної програми суттєво покращить досвід управління справами та заходами, пропонуючи зручний і ефективний інструмент для організації та контролю справ. Традиційні методи роботи з паперовими записами, мають безліч недоліків, в той же час зберігання нотаток про справи в електронному форматі дозволить користувачам швидко отримувати доступ до своїх даних, легко знаходити потрібну інформацію та зручно ділитися списками запланованих справ, що значно підвищить якість досвіду організації робочого та вільного часу, ефективність і продуктивність.

## 1 ОПИС ВИМОГ

### 1.1 Сценарії роботи програми

#### **Сценарій 1. Додавання нової справи**

##### *Передумова*

Користувач відкрив головне вікно програми.

##### *Основний сценарій:*

1. Користувач натискає кнопку «Add»;
2. Програма відкриває вікно для створення і додавання нової справи;
3. Користувач вводить дані про справу;
4. Програма перевіряє коректність введених даних, у разі успішної перевірки, створює і додає нову справу до списку справ.

##### *Додатковий сценарій:*

1. Користувач натискає кнопку «Add»;
2. Програма відкриває вікно для створення і додавання нової справи;
3. Користувач вводить дані про справу;
4. Програма виявляє некоректність введених даних, не додає справу до списку;
5. Якщо введені дані коректні, але справа перетинається у часі з іншою, програма відкриває вікно «Overlap resolution», що сповіщає про перетинання;
6. Користувач підтверджує або не підтверджує додавання нової справи, що перетинається з іншою, натискаючи кнопки «Yes» або «No»;
7. У разі підтвердження, програма додає нову справу.

#### **Сценарій 2. Перегляд усіх збережених справ за попередньо визначеними датами**

##### *Передумова*

Користувач відкрив головне вікно програми, бачить усі справи з

відкритого списку.

*Основний сценарій:*

1. Користувач натискає на комбобокс, обирає одну з наступних опцій: «All», «Today», «Tomorrow», «The day after tomorrow»;
2. Програма відображає відфільтровані події за обраною опцією.

### **Сценарій 3. Редагування справи**

*Передумова*

Користувач відкрив головне вікно програми й обрав зі списку певну справу натисканням лівої кнопки миші.

*Основний сценарій:*

1. Користувач натискає кнопку «Edit»;
2. Програма відкриває вікно редагування справи;
3. Користувач змінює дані в елементах керування користувацького інтерфейсу, підтверджує свій вибір;
4. Програма перевіряє коректність введених даних, у разі успіху перевірки, змінює справу у списку.

*Додатковий сценарій:*

1. Користувач натискає кнопку «Edit»;
2. Програма відкриває вікно редагування справи;
3. Користувач змінює дані в елементах керування користувацького інтерфейсу, підтверджує свій вибір;
4. Програма перевіряє коректність введених даних, не додає справу до списку;
5. Якщо введені дані коректні, але справа перетинається у часі з іншою, програма відкриває вікно, що сповіщає про перетинання;
6. Користувач підтверджує або не підтверджує додавання нової справи, що перетинається з іншою, натискаючи кнопки «Yes» або «No»;
7. У разі підтвердження, програма додає зміни.



#### **Сценарій 4. Видалення справи**

##### *Передумова*

Користувач відкрив головне вікно програми й обрав зі списку певну справу натисканням лівої кнопки миші.

##### *Основний сценарій:*

1. Користувач натискає кнопку «Edit»;
2. Користувач натискає кнопку «Delete»;
3. Програма видаляє справу зі списку.

#### **Сценарій 5. Перегляд справ, що відбуватимуться, починаючи з наступного дня**

##### *Передумова*

Користувач відкрив головне вікно програми.

##### *Основний сценарій:*

1. Користувач переходить до вкладки «Upcoming»;
2. Користувач може оновити список справ, натиснувши кнопку «Update», якщо він додавав справи у поточній сесії.

#### **Сценарій 6. Зміна налаштувань автоматичних нагадувань**

##### *Передумова*

Користувач відкрив головне вікно програми і перейшов до вкладки «Settings».

##### *Основний сценарій:*

1. Користувач за допомогою миші обирає текстове поле для встановлення часу, за який програма нагадає користувачу про заплановані справи;
2. Користувач вводить час, за який програма буде повідомляти користувача про справи, і натискає клавішу «Enter»;
3. Програма перевіряє коректність введених даних, у разі успішної перевірки, оновлює час за який буде надіслано повідомлення.

*Додатковий сценарій:*

1. Користувач за допомогою миші обирає текстове поле для встановлення часу, за який програма нагадає користувача про заплановані;
2. Користувач вводить час, за який програма буде повідомляти користувача про справи, і натискає клавішу «Enter»;
3. Програма виявляє некоректність введених даних і залишає встановлене значення.

**Сценарій 7. Створення нового списку справ та його збереження***Передумова*

Користувач відкрив головне вікно програми.

*Основний сценарій:*

1. Користувач натискає кнопку «File», розташовану у стрічці меню;
2. Користувач обирає «New» у відкритому меню. Відкривається діалогове вікно файлового менеджера, у якому користувач може обрати директорію, у якій буде створено новий файл, що зберігатиме список;
3. У разі успішного створення нового файлу, діалогове вікно закривається.

*Додатковий сценарій:*

1. Користувач натискає кнопку «File», розташовану у стрічці меню;
2. Користувач обирає «New». Відкривається діалогове вікно файлового менеджера, у якому користувач може обрати директорію у якій буде створено новий файл, що зберігатиме натискає кнопку «File», розташовану у стрічці меню;
3. У разі помилок з боку віконного, файлового менеджера, файлової системи, операційної системи, діалогове вікно закривається.

**Сценарій 8. Відкриття списку справ із файлу***Передумова*

Користувач відкрив головне вікно програми.

*Основний сценарій:*

1. Користувач натискає кнопку «File», розташовану у стрічці меню;
2. Користувач обирає «Open» у відкритому випадаючому меню. Відкривається діалогове вікно файлового менеджера, у якому користувач може обрати директорію, у якій знаходиться шуканий файл, і файл;
3. Користувач обирає файл, який програма буде використовувати для відкриття списку справ;
4. Програма відкриває файл.

*Додатковий сценарій:*

1. Користувач натискає кнопку «File», розташовану у стрічці меню;
2. Користувач обирає «Open» у відкритому випадаючому меню. Відкривається діалогове вікно файлового менеджера, у якому користувач може обрати директорію, у якій знаходиться шуканий файл, і файл;
3. Користувач обирає файл, що зберігає дані, що не є списками справ у контексті даної програми, або виникають помилки з боку віконного, файлового менеджера, файлової системи, операційної системи. У цих випадках діалогове вікно закривається.

**Сценарій 9. Збереження списку справ у текстовому форматі***Передумова*

Користувач відкрив головне вікно програми.

*Основний сценарій:*

1. Користувач натискає кнопку «File», розташовану у стрічці меню;
2. Користувач обирає «Save all» у відкритому випадаючому меню;
3. У відкритому діалоговому вікні файлового менеджера користувач обирає директорію, у якій буде збережено текстовий файл.
4. Програма зберігає файл.

*Додатковий сценарій:*

1. Користувач натискає кнопку «File», розташовану у стрічці меню;
2. Користувач обирає «Save all» у відкритому випадаючому меню;

5. У разі разі помилок з боку віконного, файлового менеджера, файлової системи, операційної системи, діалогове вікно закривається.

### **Сценарій 10. Збереження відфільтрованого списку справ у текстовому форматі**

#### *Передумова*

Користувач відкрив головне вікно програми та відфільтрував список справ.

#### *Основний сценарій:*

1. Користувач натискає кнопку «File», розташовану у стрічці меню;
2. Користувач обирає «Save filtered» у відкритому випадаючому меню;
3. У відкритому діалоговому вікні файлового менеджера користувач обирає директорію, у якій буде збережено текстовий файл;
4. Програма зберігає файл.

#### *Додатковий сценарій:*

5. Користувач натискає кнопку «File», розташовану у стрічці меню;
6. Користувач обирає «Save filtered» у відкритому випадаючому меню;
7. У відкритому діалоговому вікні файлового менеджера користувач обирає директорію, у якій буде збережено текстовий файл;
8. У разі разі помилок з боку віконного, файлового менеджера, файлової системи, операційної системи, діалогове вікно закривається.

### **Сценарій 11. Пошук справ у певному списку з застосуванням фільтрів**

#### *Передумова*

Користувач відкрив головне вікно програми.

#### *Основний сценарій:*

1. Користувач вводить фільтри на бічній панелі і натискає кнопку «Search»;
2. Програма відображає відфільтровані записи у елементі керування

DataGrid збоку від бічної панелі.

## **Сценарій 12. Запит статистики**

### *Передумова*

Користувач відкрив головне вікно програми та перейшов до вкладки «Statistics».

### *Основний сценарій:*

1. Програма відображає користувачу статистику за поточним списком.

## **1.2 Функціонал програми**

### **Функція 1. Додавання нової справи**

Вікно додавання нової справи (див. рис. 1.1) відкривається по натисканню на кнопку «Add» на головному вікні. На ньому розташовані текстові поля:

- 1) Name;
- 2) Duration;
- 3) Location;
- 4) Category;
- 5) Description.

Також на ньому є елементи керування обрання дати і часу:

- 1) Date;
- 2) Time.

Поле «Name» є обов'язковим для заповнення, об'єкт, що представляє захід, не буде створений, якщо не введена дата, але при цьому введений час.

The 'Create' dialog box contains the following fields and controls:

- Name:** A single-line text input field.
- Date:** Three separate input fields labeled 'day', 'month', and 'year'.
- Time:** Two separate input fields for time.
- Duration:** A single-line text input field.
- Location:** A single-line text input field.
- Category:** A single-line text input field.
- Description:** A single-line text input field.
- Create:** A button at the bottom right of the dialog.

Рисунок 1.1 – Вікно додавання нової справи

У випадку співпадіння у часі справи, що створюється, та однієї зі збережених, програма сповістить про таке перетинання користувача відкритим вікном підтвердження (див. рис. 1.2) і запитає чи бажає він створити справу, що перетинається з іншою. У разі натискання кнопки «Yes» нова справа буде додана, кнопки «No» — не буде додана.

The confirmation dialog box displays the message: "Events overlap. Do you want to save changes?"

At the bottom, there are two buttons: "Yes" and "No".

Рисунок 1.2 – Вікно підтвердження додавання справ, що перетинаються

В обох випадках діалогове вікно закриється.

Користувач може закрити вікно додавання нових справ або додати ще одну. Нова справа буде відображена в елементі керування DataGrid головного вікна.

## **Функція 2. Перегляд усіх збережених справ за різними наперед визначеними датами**

На головному вікні програми розташований комбобокс (див. рис. 1.3). При натисканні на нього випадає меню із наперед визначеними датами та можливістю відкрити усі справи зі списку:

- 1) All;
- 2) Today;
- 3) Tomorrow;
- 4) The day after tomorrow.

The screenshot shows the main application window. It has a top menu bar with 'File' and a main toolbar with 'Main', 'Upcoming', 'Past', 'Statistics', and 'Settings'. The 'Main' tab is active. On the left is a sidebar with search filters: Name (text input), From (date picker with day, month, year), To (date picker with day, month, year), Time (time input), Duration (text input), Location (text input), Category (text input), Description (text input), and a checkbox for 'Is done:'. Below these is a 'Search' button. The main area has a dropdown menu labeled 'All' with a caret icon. Below the dropdown is a list of items, each represented by a horizontal line. At the bottom of the main area are two buttons: 'Create' and 'Edit'.

Рисунок 1.3 – Головне вікно програми

При натисканні на одну з опцій програма відфільтровує записи, що відповідають заданому критерію і відображає їх в елементі керування DataGrid на головному вікні.

### **Функція 3. Редагування справи**

Для редагування справи користувач має виділити справу, що хоче відредагувати в елементі керування DataGrid головного вікна програми. Вікно редагування справи (див. рис. 1.4) відкривається по натисканню на кнопку «Edit» на головному вікні. На ньому розташовані текстові поля:

- 1) Name;
- 2) Duration;
- 3) Location;
- 4) Category;
- 5) Description.

Також на ньому є елементи керування обрання дати і часу:

- 1) Date;
- 2) Time.

Також на ньому розташований чекбокс «Is done».

При відкритті вікна в елементах керування відображаються дані обраної справи. Поле «Name» є обов'язковим для заповнення, об'єкт, що представляє захід, не буде створений, якщо не введена дата, але при цьому введений час. Поле «Category» при вводі відображає підказки зі списку наперед визначених категорій. Для збереження змін користувач натискає кнопку «Save changes».



The image shows a software window titled "Edit" with a close button in the top right corner. Inside the window, there are several form fields for editing an event:

- Name:** A single-line text input field.
- Date:** Three separate input fields for "day", "month", and "year".
- Time:** Two input fields for the time, with "AM" selected in the second field.
- Duration:** A single-line text input field.
- Location:** A single-line text input field.
- Category:** A single-line text input field.
- Description:** A single-line text input field.
- Is finished:** A checkbox.

At the bottom of the window, there are two buttons: "Save changes" and "Delete event".

Рисунок 1.4 – Вікно редагування справи

У випадку співпадіння у часі справи, що редагується, та однієї зі створених, програма сповістить про таке перетинання користувача відкритим вікном підтвердження (див. рис. 1.2) і запитає чи бажає він змінити справу. У разі натискання кнопки «Yes» справа буде відредагована, кнопки «No» — не буде відредагована. В обох випадках після вибору користувача вікно підтвердження закриється і користувач зможе знову відредагувати справу. DataGrid оновиться.

#### Функція 4. Видалення справи

Для видалення справи необхідно виділити в елементі керування DataGrid одну справу і натиснути на кнопку «Edit» на головному вікні. Далі відкриється вікно редагування справи (див. рис. 1.4). На ньому розташовані текстові поля «Name», «Duration», «Location», «Category», «Description», елементи керування обрання дати «From» і «To» та часу «Time», а також чекбокс «Is done».

Для того, щоб видалити справу потрібно натиснути кнопку «Delete».

Datagrid автоматично оновиться.

**Функція 5. Перегляд справ, що відбуватимуться, починаючи з наступного дня**

На головному вікні програми потрібно перейти до вкладки «Upcoming» (див. рис. 1.5).



Рисунок 1.5 – Вкладка «Upcoming»

Користувач за необхідності натискає кнопку «Update» для оновлення. Це може знадобитись, якщо користувач додавав нові справи, що відбуватимуться, починаючи з наступного дня, у поточній сесії користування програмою.

**Функція 6. Зміна налаштувань автоматичних нагадувань та їх показ за поточним часом**

На головному вікні користувач обирає вкладку «Settings» (див. рис. 1.6).



Рисунок 1.6 – Вкладка «Settings» головного вікна програми

Для зміни часу нагадування потрібно ввести у текстове поле кількість хвилин, за яку буде показано сповіщення і натиснути клавішу «Enter».

### Функція 7. Створення нового списку справ та його збереження

На головному вікні програми користувач натискає на кнопку «File», розташовану в стрічковому меню (див. рис. 1.7). У випадаючому списку він обирає кнопку «New». Програма відкриває діалогове вікно файлового менеджера (див. рис. 1.8). В ньому користувач обирає директорію, в яку він хоче зберегти файл (за замовчуванням, це директорія, що використовується програмою для збереження списку). Користувач може змінити назву файлу.

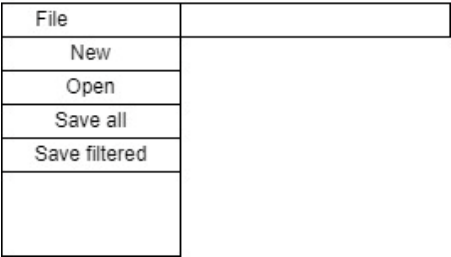


Рисунок 1.7 – Меню «File»

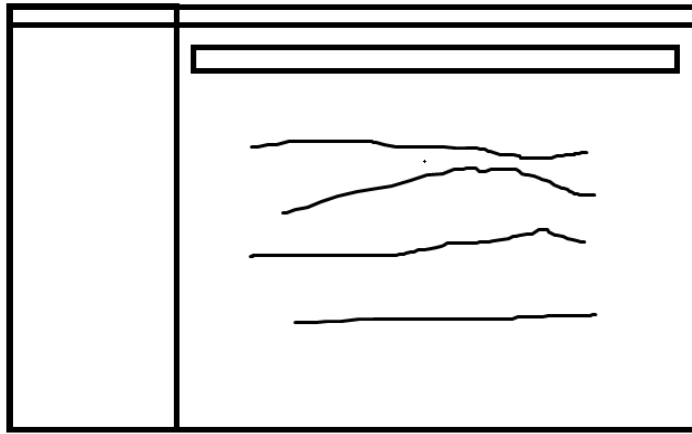


Рисунок 1.8 – Вікно файлового менеджера

Далі користувач натискає кнопку, що зберігає файл. Діалогове вікно закривається.

### **Функція 8. Відкриття списку справ із файлу**

На головному вікні програми користувач натискає на кнопку «File», розташовану в стрічковому меню (див. рис. 1.7). У випадаючому списку він обирає кнопку «Open». Програма відкриває діалогове вікно файлового менеджера (див. рис. 1.8). В ньому користувач обирає директорію, в якій знаходиться файл зі списком справ. Користувач підтверджує свій вибір. Діалогове вікно закривається.

### **Функція 9. Зберігання у текстовому файлі списку справ**

На головному вікні програми користувач натискає на кнопку «File», розташовану в стрічковому меню (див. рис. 1.7). У випадаючому списку він обирає кнопку «Save all». Програма відкриває діалогове вікно файлового менеджера (див. рис. 1.8). В ньому користувач обирає директорію, в яку він хоче зберегти файл (за замовчуванням, це директорія, що використовується програмою для збереження списку). Користувач може змінити назву файлу.

Далі користувач натискає кнопку, що зберігає файл. Діалогове вікно закривається.

### **Функція 10. Зберігання у текстовому файлі відфільтрованого списку справ**

На головному вікні програми користувач натискає на кнопку «File», розташовану в стрічковому меню (див. рис. 1.7). У випадаючому списку він обирає кнопку «Save filtered». Програма відкриває діалогове вікно файлового менеджера (див. рис. 1.8). В ньому користувач обирає директорію, в яку він хоче зберегти файл (за замовчуванням, це директорія, що використовується програмою для збереження списку). Користувач може змінити назву файлу.

Далі користувач натискає кнопку, що зберігає файл. Діалогове вікно закривається.

### **Функція 11. Пошук справ у певному списку із застосуванням фільтрів**

На бічній панелі (див. рис. 1.9) вкладки «Main» головного вікна програми розташовані текстові поля:

- 1) Name;
- 2) Duration;
- 3) Location;
- 4) Category;
- 5) Description.

Також на ньому є елементи керування обрання дати і часу:

- 1) From;
- 2) To;
- 3) Time.

Також на ньому розташований чекбокс «Is done».

Відсутність даних у полі або іншому елементі означає, що відповідний фільтр не буде застосований. Значення, введені в текстові поля «Name», «Duration», «Location», «Category», «Description», будуть розглядатися як можливі підрядки та використовуватимуться для часткового збігу. Поле «Category» при вводі відображає підказки зі списку наперед визначених

категорій.

Name:

From:  

day	month	year
-----	-------	------

To:  

day	month	year
-----	-------	------

Time:  

<input type="text"/>	<input type="text"/>
----------------------	----------------------

Duration:

Location:

Category:

Description:

Is done:  
☐

Рисунок 1.9 – Бічна панель головного вікна

Результат пошуку буде відображатись у елементі керування DataGrid з правого боку від бічної панелі, тобто в основній DataGrid даних програми. Якщо жодний запис не задовільняє умовам фільтрування, у DataGrid не будуть відображатися справи.

### Функція 12. Запит статистики

На головному вікні програми користувач обирає вкладку «Statistics» (див. рис. 1.10).

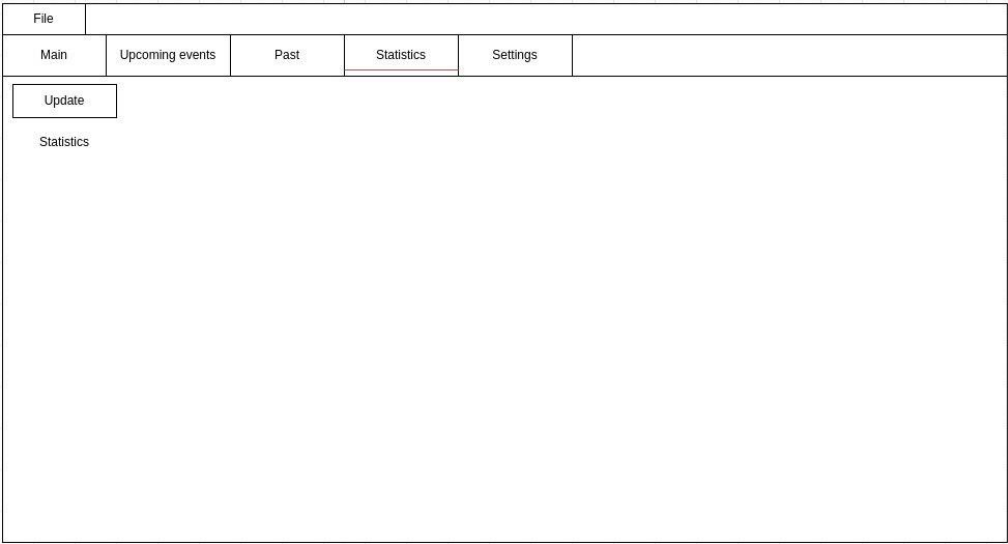


Рисунок 1.10 – Вкладка «Statistics»

Користувач бачить дані (загальна кількість справ, розподіл справ за певним критерієм тощо), що можуть бути цікавими для нього.

## 2 ПРОЕКТУВАННЯ ПРОГРАМИ

### 2.1 Архітектура програми

Для розробки користувацького інтерфейсу даної програми був обраний фреймворк Avalonia [1]. Його перевагою як фреймворка є те, що він є кросплатформним, що дозволяє створювати застосунки, які можуть працювати на різних операційних системах, таких як Windows, macOS та Linux. Також слід зазначити, що Avalonia використовує ReactiveUI [2]. ReactiveUI — це бібліотека для побудови користувацьких інтерфейсів, що базується на реактивному програмуванні. Вона забезпечує ефективне управління станом застосунку та дозволяє легко створювати динамічні та інтерактивні інтерфейси.

Avalonia і ReactiveUI надають можливість легко реалізувати архітектурний патерн MVVM, що є основним в архітектурі даного програмного забезпечення. Цей патерн розділяє логіку застосунку на три компоненти: Model (модель), View (вигляд) та ViewModel (модель вигляду). Model представляє дані та бізнес-логіку програми. View відповідає лише за відображення інтерфейсу користувача. ViewModel забезпечує зв'язок між Model і View, керує станом вигляду та обробляє команди від користувача.

Так як програма має не дуже складний для реалізації функціонал, використовує лише дані, що зберігаються локально та розрахована на використання одним користувачем, частина бізнес-логіки була інкапсульована за допомогою патерну команда і реалізована у View-Models. Це сприяє зменшенню зв'язування між користувацьким інтерфейсом та логікою і забезпечує більш підтримуваний та придатний до тестування код.

Для того, щоб забезпечити можливість тестування з використанням об'єктів-макетів, частина залежностей (наприклад, `IEventsDataProvider`) передаються через параметри конструктора.



## 2.2 Загальна структура програми

У проєкті є розділення за директоріями моделей, моделей вигляду, виглядів та утиліт (див. рис. 2.1).

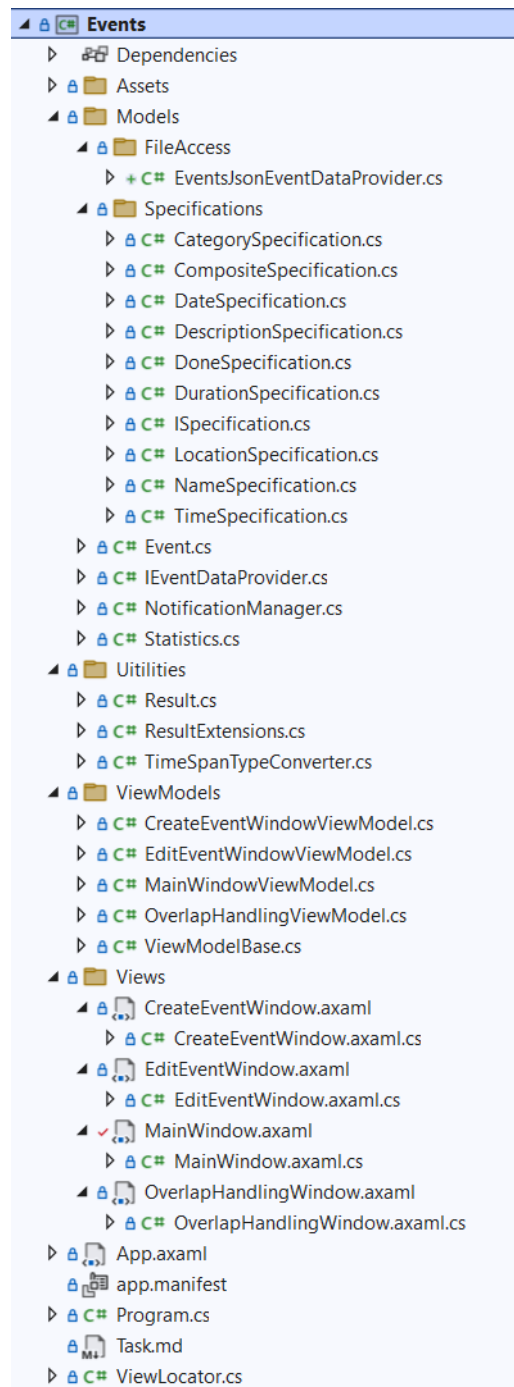


Рисунок 2.1 – Загальна структура

### 2.2.1 Моделі даних

Клас `Event` моделює заходи або справи. Клас реалізує інтерфейс `INotifyPropertyChanged`.

В ньому визначені наступні властивості:

- `Name (string)` – назва;
- `DateTime (Nullable<DateTime>)` – дата і час;
- `Duration (Nullable<TimeSpan>)` – тривалість;
- `Location (Nullable<string>)` – локація;
- `Category (Nullable<string>)` – категорія;
- `Description (Nullable<string>)` – опис;
- `Done (Nullable<bool>)` – чи завершена справа.

Методи:

- `void OnPropertyChanged()` – обробник подій;
- `string ToString()` – перевизначений метод `Object`.

Подія:

- `PropertyChanged` – реалізація інтерфейсу `INotifyPropertyChanged`.

Клас `EventsJsonDataProvider` відповідає за реалізацію CRUD операцій та зберігання в пам'яті списку справ. Він реалізує інтерфейс `IEventDataProvider`.

Поля:

- `_filePath (string)` – шлях до файлу зі списком справ;
- `_events (List<Event>)` – список справ;
- `_options(JsonSerializerOptions)` – опції серіалізації.

Методи:

- `Result<Event> AddEvent(Event @event)` – додавання справи;
- `Result<List<Event>> GetEventListByCondition(Func<Event, bool>)` – запит списку справ, що задовільняють певній умові;
- `Result<List<Event>> GetAllEvents()` – запит усіх справ;

- Result UpdateEvent(Event @event) – оновлення справи;
- Result DeleteEvent(Guid id) – видалення справи;
- Result<Event> GetEventById(Guid id) – запит справи за GUID;
- Result SubmitChanges() – збереження змін у файл.

Клас NotificationManager надає статичний метод із сигнатурою public static Result ShowNotification(string title, string message) для показу сповіщень.

Клас Statistics має наступні поля:

- \_events (List<Event>) – колекція подій;
- \_stringRepresentation (Nullable<string>) – строкове представлення.

Властивості:

- TotalNumberOfEvents (int) – загальна кількість справ;
- NumberOfCompletedEvents (int) – кількість завершених справ;
- NumberOfUpcomingEvents (int) – кількість майбутніх справ;
- NumberOfPastEvent (int) – кількість минулих справ;
- EventDistributionByCategory (Dictionary<string, int>) – розподіл справ за категоріями;
- EventsByLocation (Dictionary<string, int>) – розподіл справ за розташуванням;
- EventDistributionByDayOfWeek (Dictionary<DayOfWeek, int>) – розподіл справ за днями тижня;
- StringRepresentation (Nullable<string>) – строкове представлення.

Для реалізації пошуку за комбінованими критеріями реалізовано патерн специфікація. Визначений інтерфейс ISpecification та класи, що реалізують його: CategorySpecification, DataSpecification, DescriptionSpecification, DoneSpecification, DurationSpecification, LocationSpecification, NameSpecification, TimeSpecification. Для комбінації декількох критеріїв пошуку використовується клас CompositeSpecification.

Інтерфейс ISpecification:

- bool IsSatisfiedBy(Event @event).

### 2.2.2 Моделі вигляду

Кожне вікно програми (вигляд) пов'язане із відповідною моделлю вигляду. Усі моделі вигляду успадковуються від `ViewModelBase`, що реалізовує `INotifyPropertyChanged`.

Клас `MainWindowViewModel` є моделлю вигляду для головного вікна програми.

Поля:

- `_dataProvider (IDataProvider)` – провайдер даних;
- `_filepath (string)` – шлях до файлу зі списком подій;
- `_timer (Timer)` – таймер для перевірки майбутніх подій;
- `_notificationThresholdMinutes (int)` – поріг часу для сповіщень;
- `_notificationThresholdMinutesTemp (Nullable<int>)` – тимчасова змінна для порогу сповіщень;
- `_shownNotfications (List<Guid>)` – список ідентифікаторів подій, для яких вже були показані сповіщення;
- `_showNotifications (bool)` – флаг для визначення, чи показувати сповіщення.

Властивості:

- `FilteredEvents (ObservableCollection<Event>)` – відфільтрований список подій;
- `UpcomingEvents (ObservableCollection<Event>)` – список майбутніх подій;
- `PastEvents (ObservableCollection<Event>)` – список минулих подій;
- `SelectedEvent (Event)` – вибрана подія;
- `SelectedFilter (string)` – вибраний фільтр;
- `DateToFilterBy (Nullable<DateTimeOffset>)` – дата початку для фільтрації;

- `DateToFilterTo(Nullable<DateTimeOffset>)` – дата завершення для фільтрації;
- `DoneFilter (Nullable<bool>)` – фільтр за завершеними/незавершеними подіями;
- `NameFilter (string)` – фільтр за назвою події;
- `DescriptionFilter (Nullable<string>)` – фільтр за описом події;
- `LocationFilter (Nullable<string>)` – фільтр за місцем проведення події;
- `CategoryFilter (Nullable<string>)` – фільтр за категорією події;
- `DurationFilter (Nullable<TimeSpan>)` – фільтр за тривалістю події;
- `TimeFilter (Nullable<TimeSpan>)` – фільтр за часом події;
- `Statistics (Nullable<string>)` – рядок зі статистикою подій;
- `OpenCreateEventWindowCommand (ReactiveCommand<Unit, Unit>)` – команда для відкриття вікна створення події;
- `OpenEditEventWindowCommand (ReactiveCommand<Unit, Unit>)` – команда для відкриття вікна редагування події;
- `FilterEventsComboboxCommand (ReactiveCommand<string, Unit>)` – команда для фільтрації подій за значеннями комбо-боксу;
- `FilterEventSearchButtonCommand (ReactiveCommand<Unit, Unit>)` – команда для фільтрації подій за параметрами;
- `OpenFileCommand (ReactiveCommand<Unit, Unit>)` – команда для відкриття діалогу вибору файлу;
- `NewListCommand (ReactiveCommand<Unit, Unit>)` – команда для створення нового списку подій;
- `SaveAllEventsCommand (ReactiveCommand<Unit, Unit>)` – команда для збереження всіх подій у текстовий файл;
- `SaveFilteredEventsCommand (ReactiveCommand<Unit, Unit>)` – команда для збереження відфільтрованих подій у текстовий файл;
- `UpdatePastEventsCommand (ReactiveCommand<Unit, Unit>)` – команда для оновлення списку минулих подій;
- `UpdateUpcomingEventsCommand (ReactiveCommand<Unit, Unit>)` –

команда для оновлення списку майбутніх подій;

— `SaveReminderSettingsCommand (ReactiveCommand<Unit, Unit>)` –

команда для збереження налаштувань нагадувань;

— `UpdateStatsCommand (ReactiveCommand<Unit, Unit>)` – команда для оновлення статистики.

Методи:

— `void Initialize()` - ініціалізація початкових даних;

— `void ReloadEvents()` - перезавантаження подій з джерела даних;

— `void CheckUpcomingEvents(object sender, ElapsedEventArgs e)` – перевірка майбутніх подій і показ сповіщень;

— `void UpdateStatistics()` - оновлення статистики подій;

— `void SaveReminderSettings()` - збереження налаштувань нагадувань;

— `void CombineSpecifications()` - комбінація специфікацій для фільтрації подій;

— `void FilterEvents()` - фільтрація подій за заданими параметрами;

— `void FilterEventsByComboBoxValues(string filter)` - фільтрація подій за значеннями комбо-боксу;

— `void ResetFilterProperties()` - скидання властивостей фільтра;

— `void UpdateUpcomingEvents()` – оновлення списку майбутніх подій;

— `void UpdatePastEvents()` – оновлення списку минулих подій;

— `void SaveChanges()` – збереження змін до джерела даних;

— `void ShowOpenFileDialog()` – показ діалогу вибору файлу;

— `void CreateNewList()` – створення нового списку подій;

— `void SaveAllEventsFromListToTxt()` – збереження всіх подій у текстовий файл;

— `void SaveFilteredEventsToTxt()` – збереження відфільтрованих подій у текстовий файл;

— `StringBuilder CreateStringBuilderFromEvents(IEnumerable<Event> events)` – створення `StringBuilder` з подій;

— `void ShowCreateFileDialog()` – показ діалогу створення файлу;

- `void OpenCreateEventWindow()` – відкриття вікна створення події;
- `void OpenEditEventWindow()` – відкриття вікна редагування події.

Клас `CreateEventWindowViewModel` відповідає за створення нових справ у додатку. Він реалізує інтерфейс `INotifyPropertyChanged`.

Властивості:

- `Name (string)` – назва справи. Вимагається для створення справи;
- `Date (Nullable<DateTimeOffset>)` – дата справи;
- `Time (Nullable<TimeSpan>)` – час справи;
- `Duration (Nullable<TimeSpan>)` – тривалість справи;
- `Location (Nullable<string>)` – місце проведення справи;
- `Category (Nullable<string>)` – категорія справи;
- `Description (Nullable<string>)` – опис справи;
- `Done (Nullable<bool>)` – статус завершення справи;
- `Suggestions (List<string>)` – список підказок для категорій;
- `CreateEventCommand (ReactiveCommand<Unit, Unit>)` —

команда для створення нової справи;

- `OpenOverlapHandlingWindowCommand (ReactiveCommand<Unit, Unit>)` — команда для відкриття вікна обробки перекриттів справ.

Методи:

- `void CreateNewEventWithOverlapCheck()` — створює нову справу з перевіркою на перекриття з існуючими справами;
- `void CreateEvent()` — додає нову справу до постачальника даних;
- `bool AreEventsOverlapping(DateTime? dateTime, TimeSpan? duration)` – перевіряє, чи перекривається нова справа з існуючими;
- `void OpenOverlapHandlingWindow()` — відкриває вікно для обробки перекриттів справ.

Подія:

- `EventCreated` — подія, що викликається після створення нової події.

### 2.3 Зберігання даних

Дані зберігаються програмою у форматі JSON. Вибір цього формату обумовлений наступними факторами. По-перше, обсяг даних, що потрібно зберігати, буде невеликим, тож зберігання в оперативній пам'яті усього списку із певного файлу не створить проблем із надлишковим використанням пам'яті. По-друге, програмі фактично необхідно зберігати об'єкти лише одного класу – Event.

Платформа .NET 8.0 [3] пропонує широкий набір інструментів для роботи з файлами у форматі JSON, зокрема, типи простору імен System.Text.Json [4]. Вони надають зручний API, який відповідає всім потребам даної програми. System.Text.Json дозволяє легко серіалізувати та десеріалізувати об'єкти, забезпечуючи високу продуктивність та гнучкість у роботі з JSON.

Для інкапсуляції доступу до даних був створений клас EventsDataProvider, який реалізує інтерфейс IEventDataProvider, що спрощує тестування.

Інтерфейс IEventDataProvider визначає наступні методи:

- AddEvent;
- GetEventListByCondition;
- UpdateEvent;
- DeleteEvent;
- GetAllEvents;
- GetEventById;
- SubmitChanges.



## 3 ІНСТРУКЦІЯ КОРИСТУВАЧА

### 3.1 Розгортання програми

Вимоги до програмного забезпечення, встановленого на комп'ютері:

- операційна система Linux;
- .NET 8.0;
- бібліотека libnotify та інструмент командного рядка notify-send [5];
- Git.

Для запуску програми необхідно виконати наступні кроки:

1. За допомогою терміналу завантажте репозиторій командою “git clone <https://github.com/kolodiiuk/EventsCoursework.git>” у певну директорію;
2. Відкрийте директорію “EventsCoursework”;
3. Відкрийте директорію “docs”;
4. Відкрийте директорію “executable”;
5. Запустіть програму “Events.exe”.

### 3.2 Користування програмою

#### 3.2.1 Навігація у програмі

Програма має головне вікно (див. рис. 3.1), у якому користувач може переходити по 5 вкладках, натиснувши правою кнопкою миші на відповідний елемент користувацького інтерфейсу.

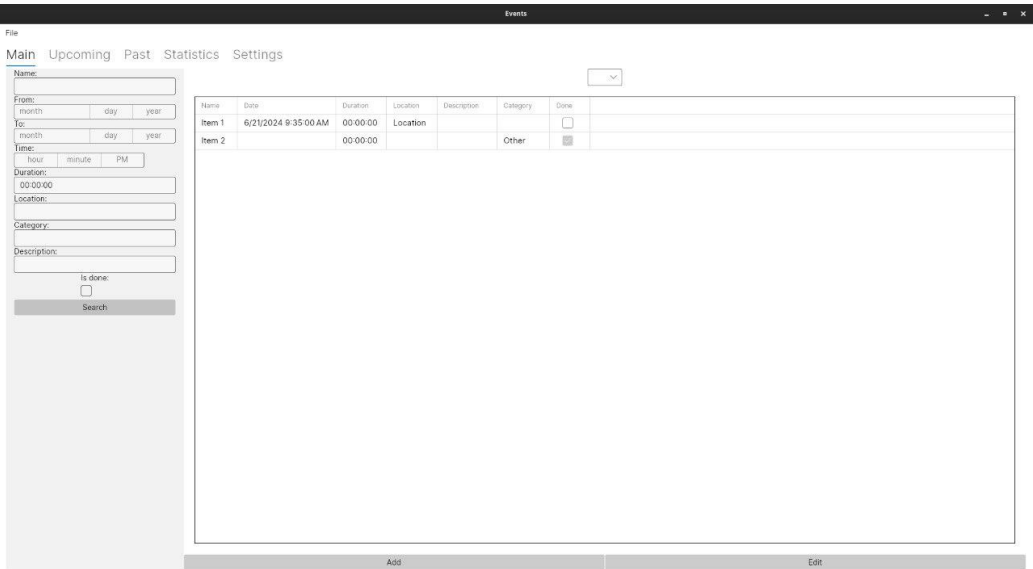


Рисунок 3.1 – Вигляд головного вікна програми

Також для додавання, редагування і видалення справ існують 3 допоміжних вікна.

3.2.2 Користування функціями, що надає головне вікно програми

На початку роботи програма відкриває список справ із файлу «Events». На вкладці «Main» користувач може переглядати список справ, представлений у вигляді таблиці. За допомогою елемента керування комбобокс користувач може відфільтрувати справи за датою, обираючи наперед визначені опції: «All», «Today», «Tomorrow», «The day after tomorrow».

Для більш точного пошуку справ за певними критеріями користувач має ввести дані в елементи бічної панелі і натиснути кнопку «Search». Після натискання цієї кнопки користувач бачить результати пошуку в таблиці. Для повернення до перегляду усіх справ у списку користувач натискає на комбобокс і обирає «All».

Для того, щоб зберегти увесь список користувач натискає кнопку

«File», далі «Save all». У відкритому діалоговому вікні (див. рис. 3.2) він обирає в яку директорію він хоче зберегти список справ і підтверджує збереження натисканням кнопки «Save».

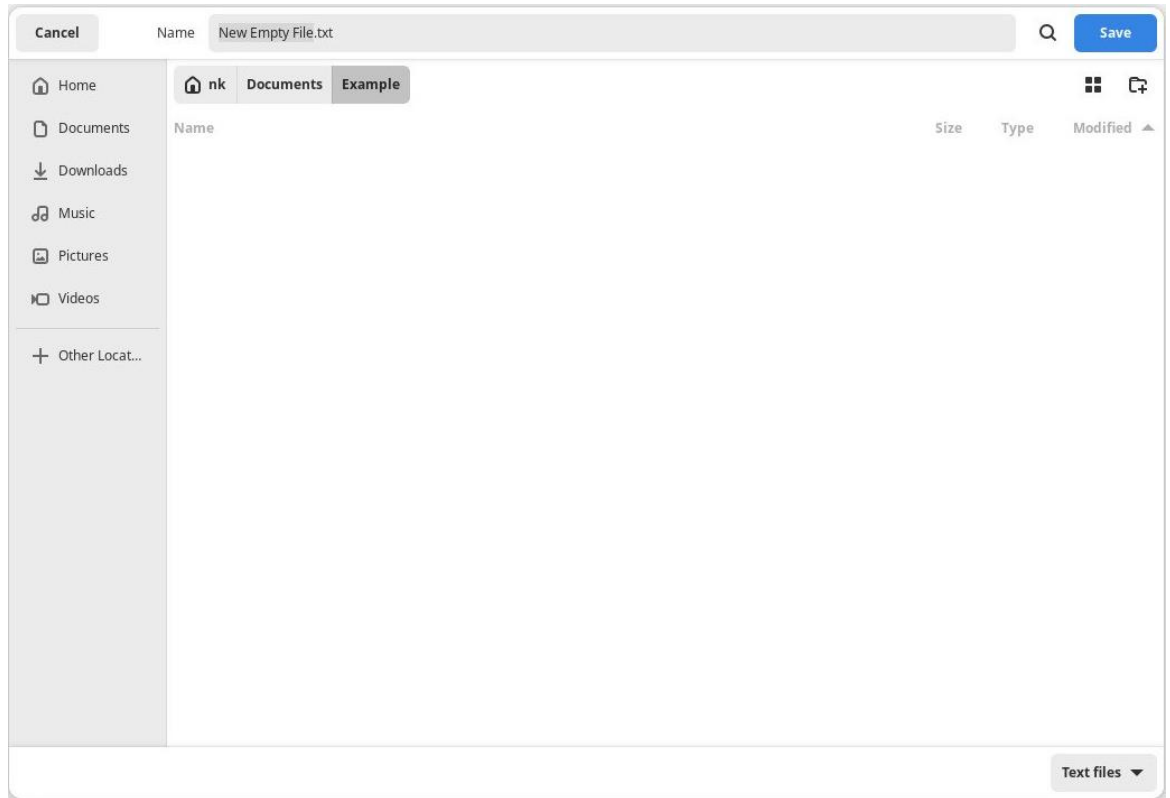


Рисунок 3.2 – Приклад діалогового вікна збереження файлу

Для збереження відфільтрованої частини списку користувач натискає кнопку «File», далі «Save filtered». У відкритому діалоговому вікні (див. рис. 3.2) він обирає в яку директорію він хоче зберегти список справ і підтверджує збереження натисканням кнопки «Save».

Користувач може створювати нові списки справ і зберігати їх у форматі JSON. За замовчуванням програма створює директорію «events-storage», у яку зберігається файл, що представляє список справ. Для того, щоб додати новий список користувач натискає кнопку «File», далі «New». У відкритому діалоговому вікні (див. рис. 3.3) він обирає в якій директорій він хоче створити новий список справ (рекомендовано зберігати списки справ у директорії за замовчуванням), змінює ім'я файлу і підтверджує збереження

натисканням кнопки «Save».

Для відкриття списку справ із файлу користувач натискає кнопку «File», далі «Open». У відкритому діалоговому вікні (див. рис. 3.3) він обирає з якої директорії він хоче відкрити список справ (рекомендовано зберігати списки справ у директорії за замовчуванням), змінює ім'я файлу і підтверджує відкриття натисканням кнопки «Select».

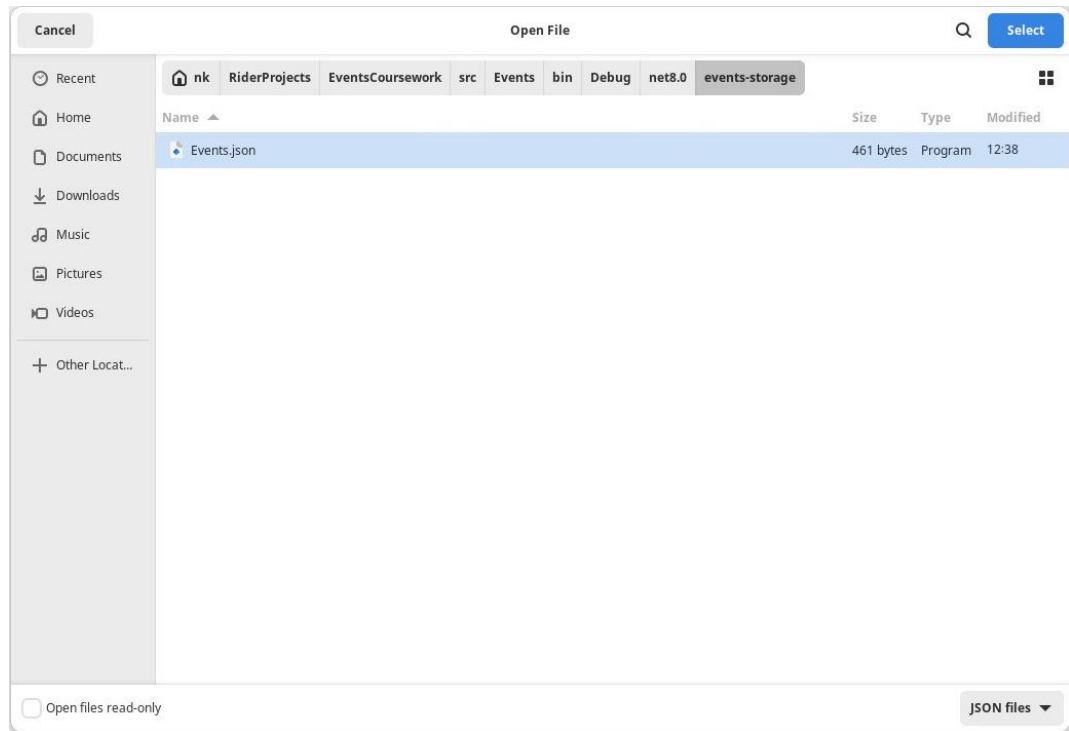


Рисунок 3.3 – Приклад діалогового вікна відкриття файлу

На вкладці «Upcoming» (див. рис. 3.4) користувач може побачити справи, що відбуватимуться, починаючи з наступного дня.

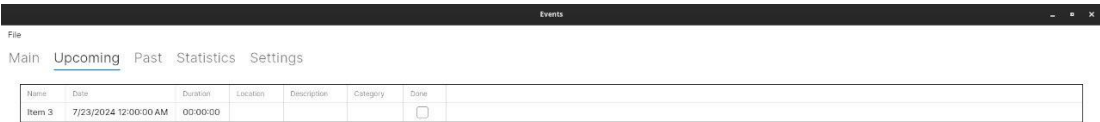


Рисунок 3.4 – Вигляд вкладки «Upcoming»

На вкладці «Past» (див. рис. 3.5) користувач бачить справи, що вже відбулися. Він може виділити одну з них та відредагувати її, натиснувши кнопку «Edit».



Рисунок 3.5 – Вигляд вкладки «Past»

На вкладці «Statistics» (див. рис. 3.6) користувач може переглянути статистику за поточним списком справ, натиснувши на кнопку «Update».

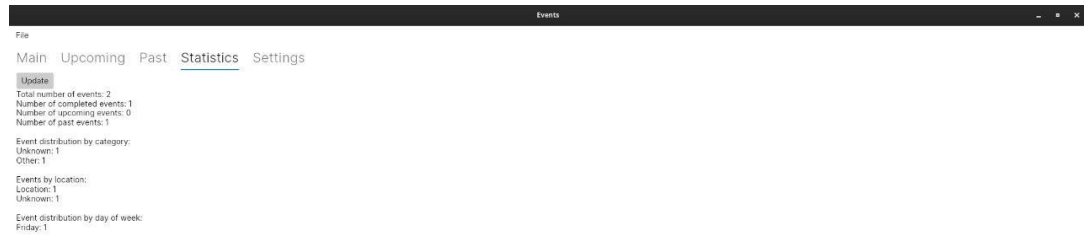


Рисунок 3.6 – Вигляд вкладки «Statistics»

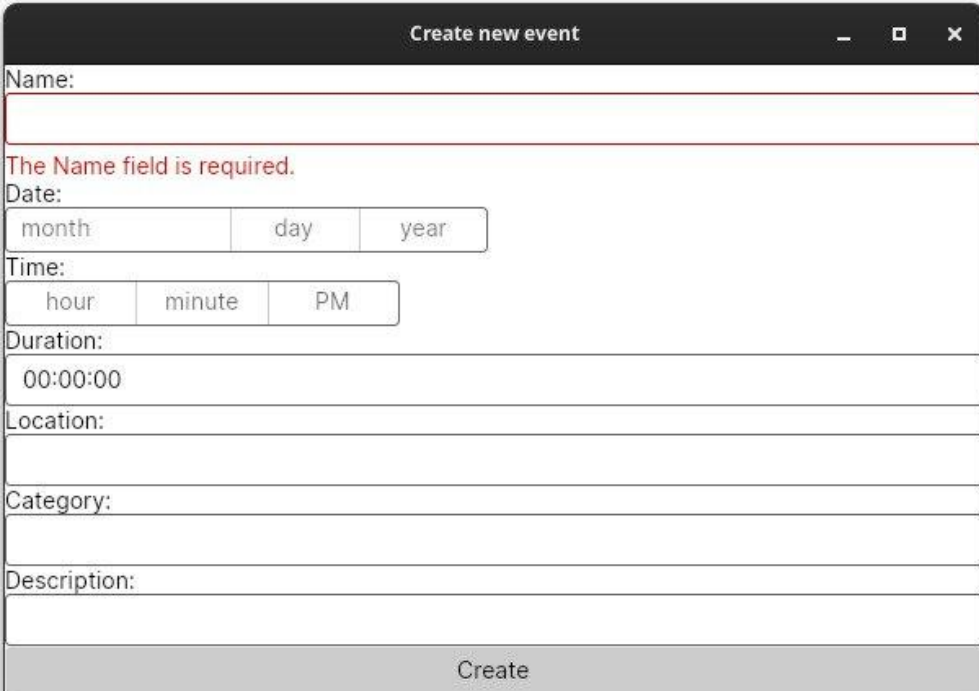
На вкладці «Settings» (див. рис. 3.7) користувач може встановити час за який буде показано нагадування про справу. Для цього він вводить кількість хвилин у текстове поле і натискає клавішу «Enter».



Рисунок 3.7 – Вигляд вкладки «Settings»

### 3.2.3 Користування функціями створення, редагування і видалення

Для того, щоб створити нову справу, користувач натискає кнопку «Add» на вкладці «Main» головного вікна програми. Відкривається вікно додавання нової справи (див. рис. 3.8), у якому користувач може задати усі потрібні йому характеристики справи. Обов'язковим для заповнення є поле «Name», крім того не можливо створити справу, що має час і не має дати. Для підтвердження створення користувач натискає кнопку «Save».



The screenshot shows a window titled "Create new event" with standard window controls (minimize, maximize, close). The form inside includes the following fields and elements:

- Name:** A text input field with a red border and a red error message below it: "The Name field is required."
- Date:** A section containing three input fields: "month", "day", and "year".
- Time:** A section containing three input fields: "hour", "minute", and "PM".
- Duration:** A time input field showing "00:00:00".
- Location:** A text input field.
- Category:** A text input field.
- Description:** A text input field.
- Create:** A button at the bottom of the form.

Рисунок 3.8 – Вікно «Create new event»


Програма проаналізує чи є перетинання у часі з іншою справою, буде показано вікно (див. рис. 3.9), що запитає чи хоче користувач додати нову справу. Якщо користувач натисне кнопку «Yes» справа буде змінена, в іншому — не буде. В обох випадках діалогове вікно закриється.



Рисунок 3.9 – Вікно «Overlap resolution»

Щоб відредагувати або видалити існуючу справу, користувач має виділити у таблиці справу і натиснути на кнопку «Edit», дана кнопка є на двох вкладках: «Main» і «Past». У вікні, що відкриється, (див. рис. 3.10) розташовані елементи керування у які виводяться усі характеристики справи. По закінченню редагування користувач натискає кнопку «Save changes».





The image shows a software window titled "Edit event". It contains the following fields and controls:

- Name:** A text box containing "Item 3".
- Date:** A date picker showing "July", "22", and "2024".
- Time:** A time picker showing "0" and "00".
- Duration:** A text box containing "00:00:00".
- Location:** An empty text box.
- Category:** An empty text box.
- Description:** An empty text box.
- Is finished:** A checkbox that is currently unchecked.
- Buttons:** Two buttons at the bottom: "Save changes" and "Delete".

Рисунок 3.10 – Вікно «Edit event»

Програма проаналізує чи є перетинання у часі з іншою справою, буде показано вікно (див. рис. 3.9), що запитає чи хоче користувач змінит справу. Якщо користувач натисне кнопку «Yes» справа буде змінена, в іншому — не буде. В обох випадках діалогове вікно закриється.

Щоб видалити справу потрібно натиснути кнопку «Delete».

### 3.3 Видалення програми

Для видалення програми потрібно видалити директорію «EventsCoursework».

## ВИСНОВКИ

Програма для організації особистих справ відповідає заявленим цілям, зазначеним у вступі. Програма забезпечує зручність, ефективність та багатофункціональність у веденні щоденних справ, що робить її актуальним інструментом для сучасних користувачів.

Програмне забезпечення досягло основної мети – створення функціонального та інтуїтивно зрозумілого інструменту для збереження та управління запланованими заходами та справами. Було реалізовано низку важливих сценаріїв використання, таких як додавання нових справ, перегляд збережених справ за датами та іншими фільтрами, редагування та видалення справ, налаштування автоматичних нагадувань, збереження списків у текстовому форматі.

На даному етапі розробка програми може вважатися завершеною, оскільки всі основні функції успішно реалізовані та протестовані. Незважаючи на досягнуті результати, існують можливості для подальшого вдосконалення та розвитку програмного забезпечення. Серед можливих напрямків вдосконалення можна виділити: інтеграція з іншими сервісами, розширення функціоналу, покращення користувацького інтерфейсу.

## ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

- 1) Avalonia Docs. URL: <https://docs.avaloniaui.net/> (дата звернення: 28.05.2024)
- 2) ReactiveUI. An advanced, composable, functional reactive model-view-framework for all .NET platforms. URL: <https://www.reactiveui.net/> (дата звернення: 28.05.2024)
- 3) .NET documentation. URL: [https://learn.microsoft.com/en-us/dotnet/?WT.mc\\_id=dotnet-35129-website](https://learn.microsoft.com/en-us/dotnet/?WT.mc_id=dotnet-35129-website) (дата звернення: 28.05.2024)
- 4) System.Text.Json Namespace. URL: <https://learn.microsoft.com/en-us/dotnet/api/system.text.json?view=net-8.0> (дата звернення: 28.05.2024)
- 5) notify-send Man Page. URL: <https://ss64.com/bash/notify-send.html> (дата звернення: 28.05.2024)

## ДОДАТОК А

### Код програми

Вихідний код програми на GitHub —  
<https://github.com/kolodiiuk/EventsCoursework>.