

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”
КАФЕДРА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КУРСОВА РОБОТА

з дисципліни «Об’єктно-орієнтоване програмування»

На тему:

“єКвиток”

Студента групи ПЗ-22

спеціальності 6.121

“Інженерія програмного забезпечення”

Ясногородського Н.В.

Керівник: доцент кафедри ПЗ,

к.т.н., доцент Коротєєва Т.О.

Національна шкала _____

Кількість балів___Оцінка ECTS___

Члени комісії

_____	_____
_____	_____
_____	_____

Львів 2022

Зміст

Технічне завдання

Алгоритм роботи програми

Діаграми UML

Код програми

Протокол роботи програми

Опис та обробка виняткових ситуацій

Інструкція користувача

Висновки

Технічне завдання

Розробити програму засобами ООП згідно вказаного варіанту (Варіант №30(ПЗ-22)). Передбачити віконний режим роботи програми та інтерфейс користувача. Передбачити ввід даних у двох режимах:

- З клавіатури;
- З файлу;

Передбачити у програмі виняткові ситуації. Продемонструвати викладачу роботу розробленої програми. Сформувавати звіт курсової роботи обсягом не менше 20 сторінок.

Завдання варіанту №30:

30.Створити таблицю у візуальному середовищі

№ | Прізвище | Ім'я | Телефон | Тип білету | Строк дії білету

За заданим номером визначити строк дії білету (врахувати наявність в таблиці однакових прізвищ)

- 1) За прізвищем визначити телефон, тип білету та його строк дії
- 2) За телефоном визначити прізвище та ім'я
- 3) Знайти імена та прізвища людей, у яких однаковий тип білету.
- 4) Визначити тип білету, який має найбільшу кількість записів (прізвищ).
- 5) Вивести всі прізвища, які мають однакове значення «Строк дії білету».

Для класу створити: 1) Конструктор за замовчуванням; 2) Конструктор з параметрами; 3) конструктор копій; 4) перевизначити операції >>, << для зчитування та запису у файл. Для демонстрації роботи програми використати засоби візуального середовища програмування.

№ з/п	Зміст завдання	Дата
1	Здійснити аналітичний огляд літератури за заданою темою та обґрунтувати вибір інструментальних засобів реалізації.	26.10.22
2	Побудова UML діаграм	10.11.22
3	Розробка алгоритмів реалізації	12.11.22
4	Реалізація завдання (кодування)	14.11.22
5	Формування інструкції користувача	16.11.22
6	<p>Оформлення звіту до курсової роботи згідно з вимогами Міжнародних стандартів, дотримуючись такої структури:</p> <ul style="list-style-type: none"> - зміст; - алгоритм розв'язку задачі у покроковому представленні; - діаграми UML класів, прецедентів, послідовності виконання; - код розробленої програми з коментарями; - протокол роботи програми для кожного пункту завдання - інструкція користувача та системні вимоги; - опис виняткових ситуацій; - структура файлу вхідних даних; - висновки; - список використаних джерел. 	17.11.22

Завдання прийнято до виконання:



(Ясногородський Н.В.)

22.08.22 (підпис студента)

Керівник роботи: _____ / Коротєєва Т .О./

1. Загальні положення:

Найменування: YeKvutok.

Умовне позначення: YeKvutok.

Замовник: кафедра ПЗ

Розробник: Ясногородський Н.В.

Початок робіт: 26.10.2022 р.

Закінчення робіт: 17.11.2022 р.

2. Призначення системи:

Метою створення програми є збереження та зчитування інформації про квитки. Програма вирішує проблему підтримування письмового реєстру квитків, а також пошуку квитків, що підходять заданим критеріям. Інформація може зберігатися на носій, щоб використовуватися в майбутньому, або передаватися мережою.

3. Об'єкти даних:

Програма опрацьовує дані про власників квитків, а саме: ім'я та прізвище, телефон, тип квитку, строк дії, номер квитку. Ці дані зберігаються в оперативній пам'яті, а після експорту даних – на фізичному носії.

4. Вимоги до програмного забезпечення:

4.1. На пристрої має бути встановлений браузер та активне підключення до мережі Інтернет. Програма використовує браузерне середовище, остання оновлення якого відбулося не більше ніж 5 років тому.

4.2. *Функціональні вимоги:*

R1. Відображення існуючих квитків.

R2. Створення нових квитків.

R3. Імпорт квитків.

R4. Експорт поточних квитків.

R5. Валідація при імпорті квитків в систему

R6. Розширені можливості фільтрування інформації про квитки.

Вимоги до апаратного та програмного забезпечення:

Операційна система: MacOS/Windows xp+/Android/IOS/Linux.

Мінімальний обсяг ОЗП: 0.5Gb.

Мінімально необхідний простір на диску: 32 Мб.

Процесор: 32-розрядний з мінімальною тактовою частотою 1,5 ГГц.

Монітор: мінімальна роздільна здатність 600x400..

Периферійні пристрої: клавіатура та миша.

4.3. *Інші вимоги:*

Вимоги до надійності:

- Повідомлення при введенні неправильних даних у формі.
- Повідомлення, якщо імпортовані дані пошкоджені або мають неправильний формат.
- Повідомлення, якщо експорт даних не можливий, з можливими вказаними причинами.

5. **Стадії розробки:**

Аналіз та специфікація вимог — збір та аналіз вимог замовника до ПЗ та планування якості продукту.

Проектування — визначення структури й поведінки системи та інтерфейсу користувача.

Кодування — розроблення вихідного коду.

Тестування — перевірка програми на наявність помилок та відповідність вимогам, зазначеним у цьому технічному завданні.

Експлуатація — використання програми користувачами.

Супровід — виправлення помилок в роботі програми.

6. Вимоги до технічної документації:

- Діаграми класів UML.
- Технічне завдання.
- Текст програми.
- Інструкція з експлуатації.

Алгоритми роботи програми

Алгоритм FTBN - Find Ticket by Number

Знайти запис про квиток за його номером

FTBN1 Створення пустого вихідного масиву.

FTBN2 Ітерація по вхідному масиву.

FTBN3 Додавання до вихідного масиву елементу з номером який співпадає з шуканим

FTBN4 Виведення вихідного масиву з одного елементу в таблицю

Алгоритм FTBS - Find Tickets by Surname

Знайти записи про квитки за прізвищем власника

FTBS1 Створення пустого вихідного масиву.

FTBS2 Ітерація по вхідному масиву.

FTBS3 Додавання до вихідного масиву елементу з телефоном який співпадає з шуканим

FTBS4 Виведення вихідного масиву в таблицю

Алгоритм FTBP - Find Tickets by Phone

Знайти записи про квитки за телефоном власника

FTBP1 Створення пустого вихідного масиву.

FTBP2 Ітерація по вхідному масиву.

FTBP3 Додавання до вихідного масиву елементу з прізвищем який співпадає з шуканим

FTBP4 Виведення вихідного масиву в таблицю

Алгоритм FTBT - Find Tickets by Type

Знайти записи про квитки які мають однаковий тип квитку

FTBT1 Створення пустого вихідного масиву.

FTBT2 Ітерація по вхідному масиву.

FTBT3 Додавання до вихідного масиву елементу з типом який співпадає з шуканим

FTBT4 Виведення вихідного масиву в таблицю

Алгоритм FTBE - Find Tickets by Expiry

Знайти записи про квитки за датою закінчення

FTBE1 Створення пустого вихідного масиву.

FTBE2 Ітерація по вхідному масиву.

FTBE3 Додавання до вихідного масиву елементу з датою закінчення яка співпадає з шуканою

FTBE4 Виведення вихідного масиву в таблицю

Алгоритм GMUT - Get Most Used Ticket

Визначення тип білету який має найбільше записів

GMUT1 Створення пустої хеш мапи та змінною яка буде зберігати тип найбільш популярного квитка.

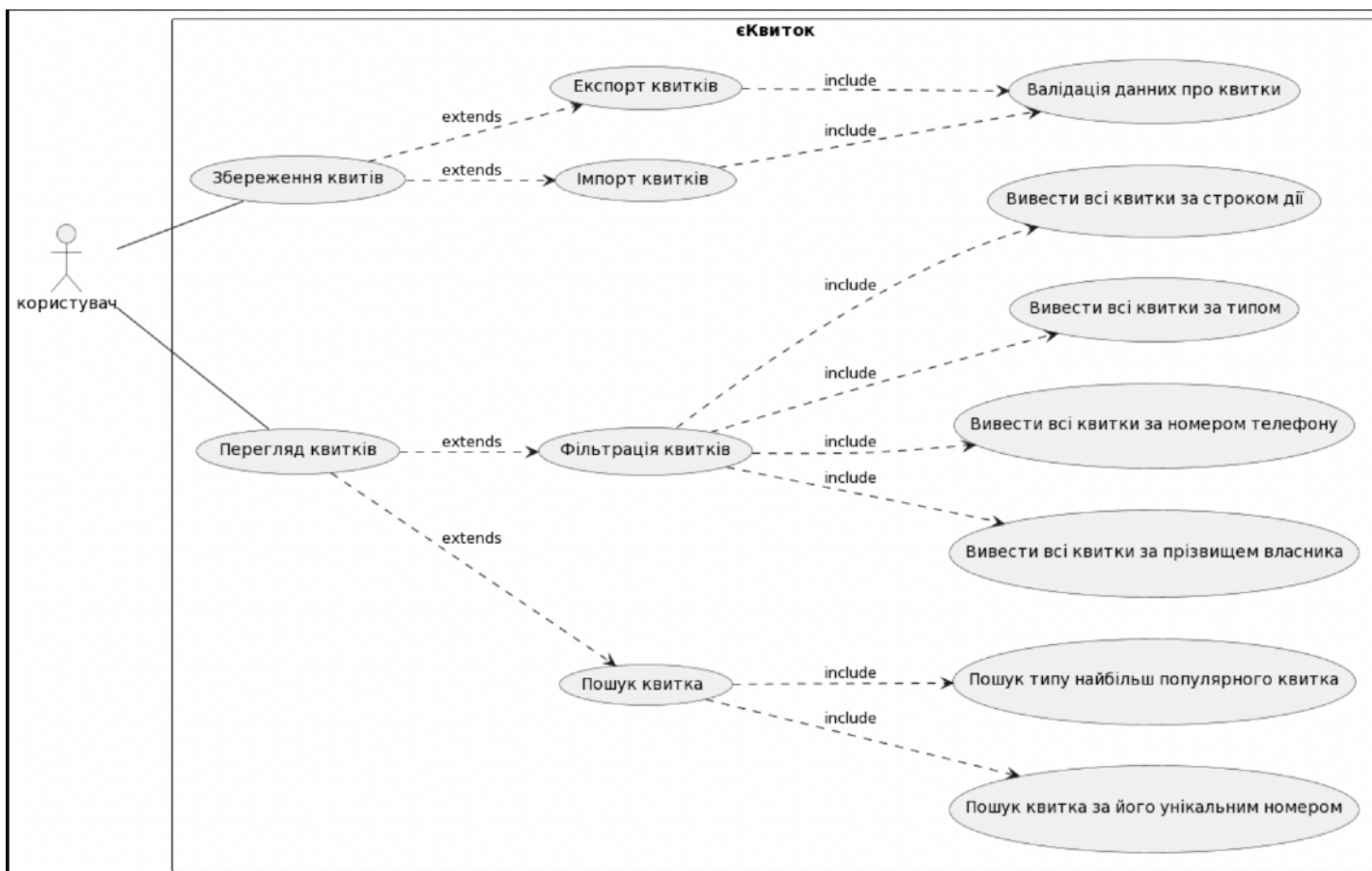
GMUT2 Ітерація по вхідному масиву квитків.

GMUT3 Оновлення значень в хеш мапі, де ключ це тип квитка а значення к-сть використань, перевірка чи більше значення поточного ключа зі змінною

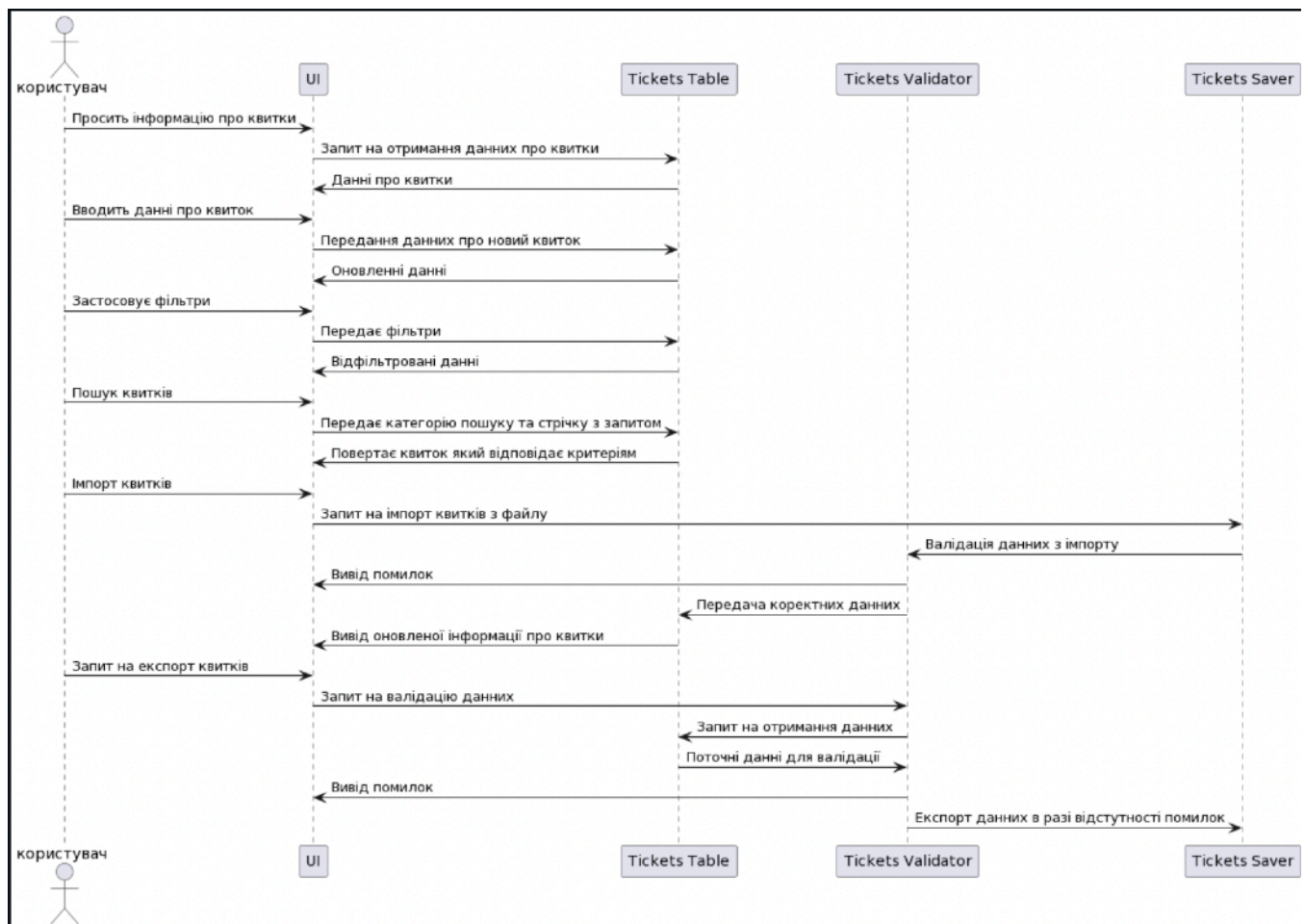
GMUT4 Виведення вмісту змінною в окрему область над таблицею білетів

Діаграми UML

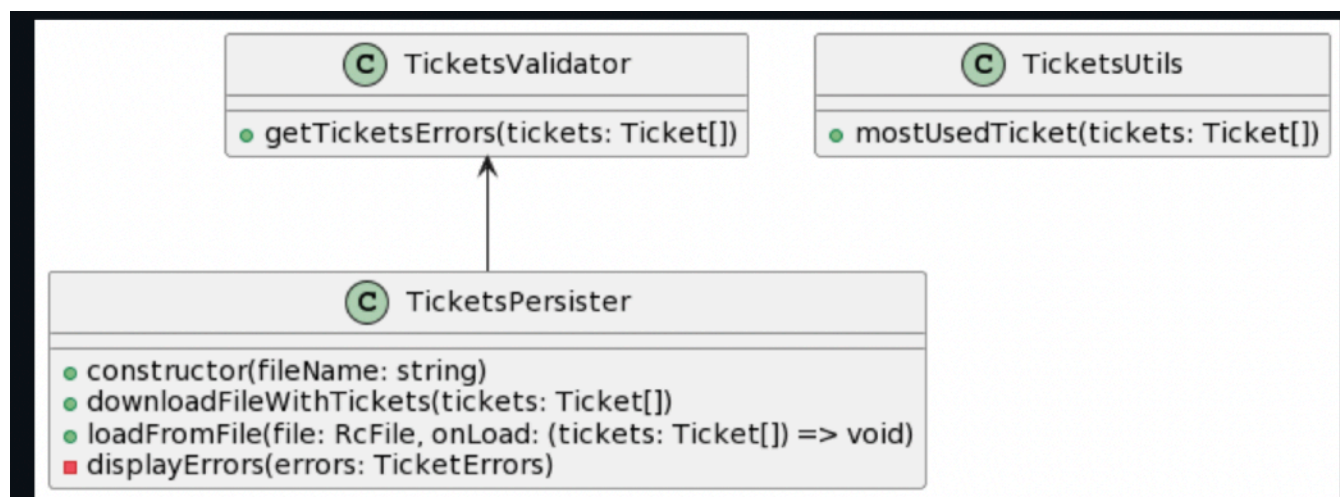
Діаграма прецедентів



Діаграма послідовностей



Діаграма класів



Код програми

Ініціалізатори:

- `_app.tsx`:

```
function MyApp({ Component, pageProps }: AppProps) {  
  return <Component {...pageProps} />;  
}  
  
export default MyApp;
```

- `index.tsx`:

```
const Home: NextPage = () => {  
  useEffect(() => {  
    // prompt before closing the tab  
    window.onbeforeunload = () => ""  
  }, [])  
  
  return (  
    <div className="site-page-header-ghost-wrapper">  
      <Head>  
        <title>Students Info</title>  
        <meta name="description" content="Tool for working with tickets" />  
        <link rel="icon" href="/favicon.ico" />  
      </Head>  
  
      <PageHeader ghost={false} title="Tickets Panel">  
        <main>  
          <TicketsTable />  
        </main>  
      </PageHeader>  
    </div>  
  )  
}
```

- - -

Моделі інформації:

- schema.ts:

```
export const ticketTypes = ["Student", "Child", "Adult"] as const

const ticketSchema = {
  elements: {
    properties: {
      key: { type: "int32" },
      name: { type: "string" },
      surname: { type: "string" },
      phone: { type: "int32" },
      ticketType: { enum: ticketTypes },
      ticketExpire: {
        type: "timestamp",
      },
    },
    optionalProperties: {},
  },
} as const

export type Ticket = JTDDatatype<typeof ticketSchema>[number]

export type TicketsValidatorReturnType = Array<{
  message: string
  meta: string
}>

const ajv = new Ajv({ allErrors: true })
export const ticketsSerializer = ajv.compileSerializer<Ticket[]>(ticketSchema)
export const ticketsValidator = ajv.compile<Ticket[]>(ticketSchema)

export const getTicketsValidationErrors = (
  tickets: Ticket[]
): TicketsValidatorReturnType => {
  if (!ticketsValidator(tickets)) {
```

```

return (
  ticketsValidator.errors?.map((e) => ({
    message: e.message
    ? `${capitalize(e.keyword)} error: ${capitalize(e.message)}`
    : "Error",
    meta: `at ${e.instancePath}`,
  })) || []
)
}
return []
}

```

Класи-утиліти:

- utils.ts:

```

export class TicketsUtils {
  static mostUsedTicket(tickets: Ticket[]) {
    const stats = {}
    let mostUsed = null
    for (let ticket of tickets) {
      stats[ticket.ticketType] ??= 0
      stats[ticket.ticketType] += 1

      if (mostUsed == null) {
        mostUsed = ticket.ticketType
      } else {
        mostUsed =
          stats[ticket.ticketType] > stats[mostUsed]
            ? ticket.ticketType
            : mostUsed
      }
    }

    if (mostUsed == null) {

```

```

        return null
    }

    return {
        ticketType: mostUsed,
        count: stats[mostUsed],
    } as const
}
}

class TicketsPersister {
    fileName: string

    constructor(fileName?: string) {
        this.fileName = fileName || "tickets"
    }

    downloadFileWithTickets(tickets: Ticket[]) {
        if (!tickets.length) return

        const errors = getTicketsValidationErrors(tickets)

        if (errors.length) {
            this.displayErrors(errors)
            return
        }

        const raw = ticketsSerializer(tickets)
        const encodedObj = URL.createObjectURL(
            new Blob([raw], {
                type: "application/json",
            })
        )
        // create "a" HTML element with href to file
        const link = document.createElement("a")
        link.href = encodedObj
    }
}

```



```

link.download = `${this.fileName}.json`
document.body.appendChild(link)
link.click()

// clean up "a" element & remove ObjectURL
document.body.removeChild(link)
URL.revokeObjectURL(encodedObj)
}

loadFromFile(file: RcFile, onLoad: (tickets: Ticket[]) => void) {
  const reader = new FileReader()
  reader.readAsText(file)
  reader.onload = () => {
    const raw = reader.result?.toString() || "[]"
    let tickets: Ticket[] = []
    let errors: TicketsValidatorReturnTypes = []

    try {
      tickets = JSON.parse(raw)
    } catch (e) {
      if (e instanceof Error) {
        errors.push({
          message: `File is corrupted (wrong json)`,
          meta: e.message,
        })
      }
    }

    errors.push(...getTicketsValidationErrors(tickets))
    this.displayErrors(errors)

    if (tickets.length) {
      onLoad(tickets)
      message.success(`Loaded tickets from ${file.name}`)
    } else {
      message.info(`File ${file.name} was empty`)
    }
  }
}

```

```

    }
  }

  private displayErrors(errors: TicketsValidatorReturnType) {
    if (!errors?.length) return

    errors.forEach((e) =>
      notification.error({
        message: `${e.message}`,
        description: `Please fix the detected error: ${e.meta}`,
        placement: "bottomLeft",
        duration: 30,
      })
    )
  }
}

export const ticketsPersistor = new TicketsPersister()

```

Графічного інтерфейс таблиці:

- tickets_table.tsx:

```

export type TicketsTableEditProps = {
  edit?: {
    component?: (save: () => void, ref: Ref<any>) => React.ReactNode
    serialize?: (value: any) => any
    deserialize?: (value: any) => any
    rules?: Rule[]
  }
  dataIndex: keyof Ticket
}

const columns: Array<ColumnType<Ticket>[number] & TicketsTableEditProps> = [
  {
    title: "No",

```

Протокол роботи програми

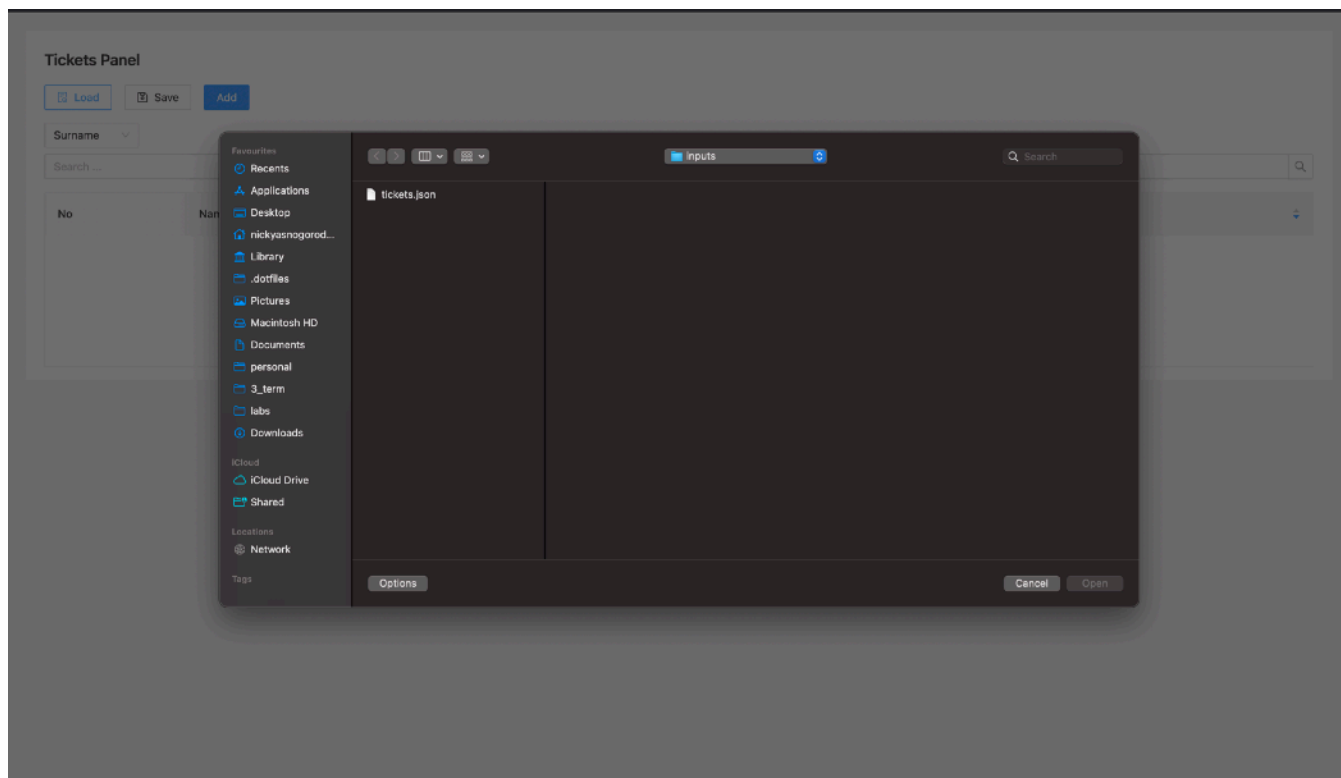


Рис. 1 – інтерфейс програми, користувач вибирає опцію імпорту

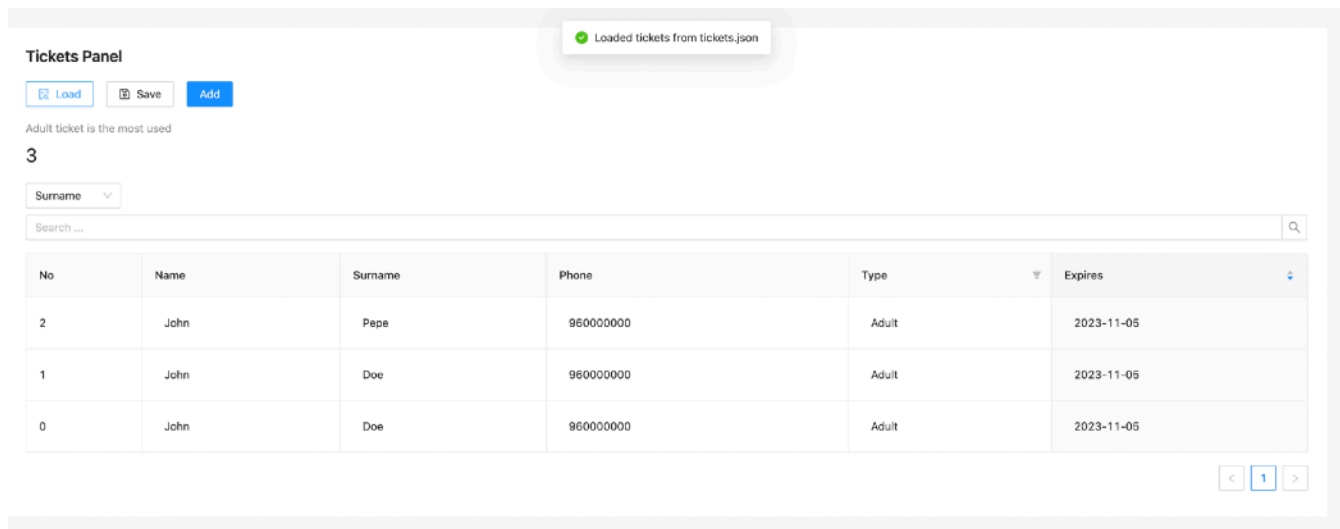


Рис. 2 – успішний імпорт даних користувачем. Сформована таблиця квитків.

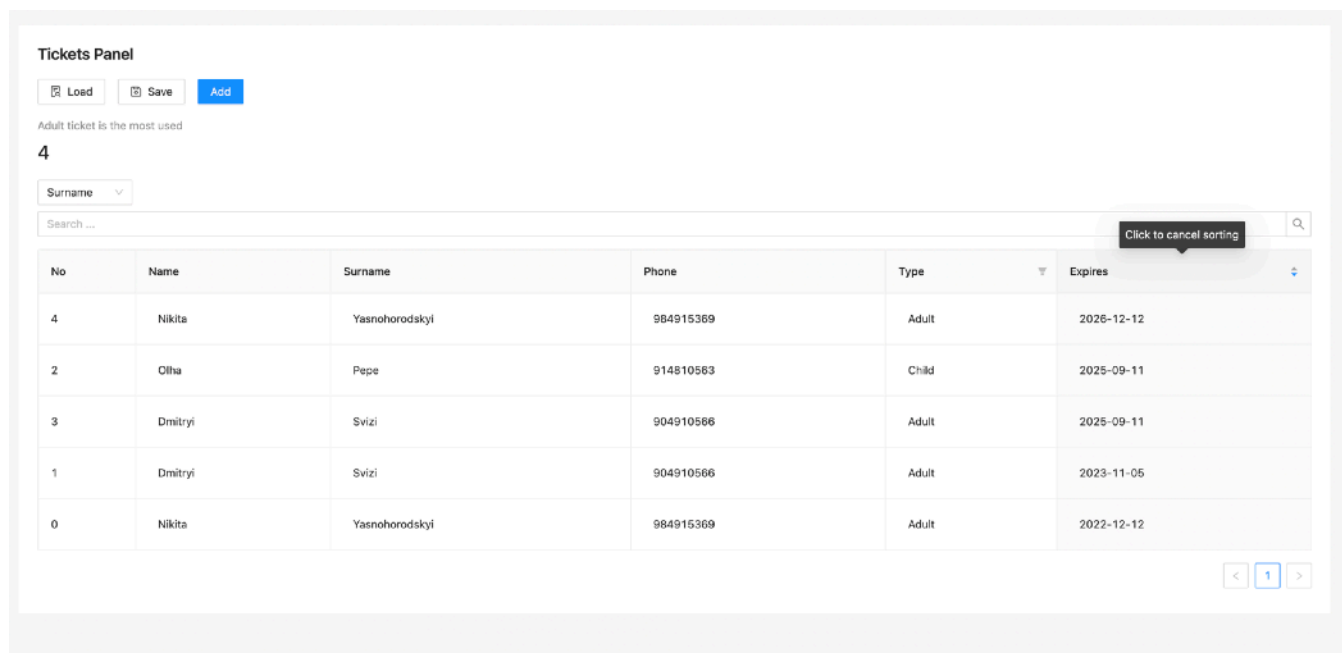


Рис.3 – користувач сортує за строком дії
квитка

Tickets Panel

Adult ticket is the most used

2

Surname ▾

Svi

No	Name	Surname	Phone	Type ▾	Expires ▴
3	Dmitryi	Svizi	904910566	Adult	2025-09-11
1	Dmitryi	Svizi	904910566	Adult	2023-11-05

< 1 >

Рис. 4 – фільтрація за прізвищем

Tickets Panel

Child ticket is the most used

1

Phone ▾

914810563

No	Name	Surname	Phone	Type ▾	Expires ▴
2	Olia	Pepe	914810563	Child	2025-09-11

< 1 >

Рис. 5 – Фільтрація за номером телефону

Tickets Panel

Load

Save

Add

Child ticket is the most used

1

Expires

2025-09-11

No	Name	Surname	Phone	Type	Expires
2	Oliha	Pepe	914810563	Child	2025-09-11
3	Dmitryi	Svizi	904910566	Adult	2025-09-11

<

1

>

Рис. 6 – Фільтрація за строком дії

Tickets Panel

Load

Save

Add

Adult ticket is the most used

4

No

Search ...

No	Name	Surname	Phone	Type	Expires
4	Nikita	Yasnohorodskiy	984915369	Adult	2026-12-12
3	Dmitryi	Svizi	904910566	Adult	2025-09-11
1	Dmitryi	Svizi	904910566	Adult	2023-11-05
0	Nikita	Yasnohorodskiy	984915369	Adult	2022-12-12

<

1

>

Рис. 7 – Фільтрація за типом квитка

Tickets Panel

Adult ticket is the most used

1

No

4

No	Name	Surname	Phone	Type <input type="button" value="v"/>	Expires <input type="button" value="v"/>
4	Nikita	Yasnohorodskiy	984915369	Adult	2026-12-12

Рис. 8 – Пошук за номером квитка

Adult ticket is the most used

4

*Рис. 9 – Пошук найбільш популярного типу
квитка*

Tickets Panel

Load

Save

Add

Adult ticket is the most used

4

No

Search ...

No	Name	Surname	Phone	Type	Expires
4	Nikita	Yasnohorodskiy	984915369	Adult	2026-12-12
2	Oliha	Pepe	914810563	Child	2025-09-11
3	Dmitryi	Svizl	904910566	Adult	2026-09-11
1	Dmitryi	Svizl	904910566	Adult	2023-11-05
0	Nikita	Yasnohorodskiy	984915369	Adult	2022-12-12

<

1

>

tickets (6).json

Show all

Рис. 10 – Експорт даних – успішно


```

5 [
4   {
3     "key": 0,
2     "name": "Nikita",
1     "surname": "Yasnohorodskyi",
    "phone": 984915369, Not Committed Yet
1     "ticketType": "Adult",
2     "ticketExpire": "2022-12-12T15:21:42.535Z"
3   },
4   {
5     "key": 1,
6     "name": "Dmitryi",
7     "surname": "Svizi",
8     "phone": 904910566,
9     "ticketType": "Adult",
0     "ticketExpire": "2023-11-05T15:21:58.432Z"
1   },
2   {
3     "key": 2,
4     "name": "Olha",
5     "surname": "Pepe",
6     "phone": 914810563,
7     "ticketType": "Child",
8     "ticketExpire": "2025-09-11T15:22:05.103Z"
9   },
0   {
1     "key": 3,
2     "name": "Dmitryi",
3     "surname": "Svizi",
4     "phone": 904910566,
5     "ticketType": "Adult",
6     "ticketExpire": "2025-09-11T15:22:05.103Z"
7   },
8   {
9     "key": 4,
0     "name": "Nikita",
1     "surname": "Yasnohorodskyi",
2     "phone": 984915369,
3     "ticketType": "Adult",
4     "ticketExpire": "2026-12-12T15:21:42.535Z"
5   }
6 ]

```

Рис.11 – Вигляд експортованих даних / даних для імпорту.

Демонстрація формату даних при роботі з програмою.

key - Номер квитку - ціле 32 бітне число

name - Стрічка з іменем власника квитка

surname - Стрічка з прізвищем власника квитка

phone - Номер телефону - ціле 32 бітне число

ticketType - Тип білету одна із стрічок (Дорослий, Дитячий, Студенський)

ticketExpire - ISO стрічка з датою про кінець строку дії квитка

Опис та обробка виняткових ситуацій

1. Неправильний формат номеру мобільного при імпорті даних

The screenshot shows a web interface titled "Tickets Panel". At the top, there are buttons for "Load", "Save", and "Add". Below these, a message states "Adult ticket is the most used" followed by the number "4". There is a dropdown menu labeled "No" and a search bar. The main part of the interface is a table with the following columns: "No", "Name", "Surname", "Phone", "Type", and "Expires". The table contains five rows of data. At the bottom right of the table, there are navigation buttons: "<", "1", and ">". Below the table, a red error message box is displayed, indicating a "Type error: Must be int32" and suggesting to fix the error at "/0/phone".

No	Name	Surname	Phone	Type	Expires
4	Nikita	Yasnohorodskiy	984915369	Adult	2026-12-12
2	Olya	Pepe	914810563	Child	2025-09-11
3	Dmitryi	Svizi	904910566	Adult	2025-09-11
1	Dmitryi	Svizi	904910566	Adult	2023-11-05
0	Nikita	Yasnohorodskiy	984915369	Adult	2022-12-12

Рис. 12 – виняткова ситуація

2. Невірний тип білету при імпорті даних

The screenshot shows the same "Tickets Panel" interface as in Figure 12. The table contains the same five rows of data. However, the error message box at the bottom now indicates an "Enum error: Must be equal to one of the allowed values" and suggests to fix the error at "/1/ticketType".

No	Name	Surname	Phone	Type	Expires
4	Nikita	Yasnohorodskiy	984915369	Adult	2026-12-12
2	Olya	Pepe	914810563	Child	2025-09-11
3	Dmitryi	Svizi	904910566	Adult	2025-09-11
1	Dmitryi	Svizi	904910566	Adult	2023-11-05
0	Nikita	Yasnohorodskiy	984915369	Adult	2022-12-12

Рис. 13 – виняткова ситуація

3. Дублікат номеру квитка при імпорті даних

Tickets Panel

Adult ticket is the most used


4

No

Search ...

No	Name	Surname	Phone	Type	Expires
4	Nikita	Yasnohorodskiy	984915369	Adult	2026-12-12
1	Olya	Pepe	914810563	Child	2025-09-11
3	Dmitryi	Svizi	904910566	Adult	2025-09-11
1	Dmitryi	Svizi	904910566	Adult	2023-11-05
0	Nikita	Yasnohorodskiy	984915369	Adult	2022-12-12

< 1 >

 Ticket numbers are not unique

Please fix the detected error: Make sure tickets have different "key"


Рис. 14 – виняткова ситуація


4. Невірний формат дати при імпорті даних

Tickets Panel

Surname

Search ...

No	Name	Surname	Phone	Type	Expires
 No data					

 Type error: Must be timestamp

Please fix the detected error: at /?ticket=Expire

Рис. 15 – виняткова ситуація

5. Невалідний формат даних json при імпорті

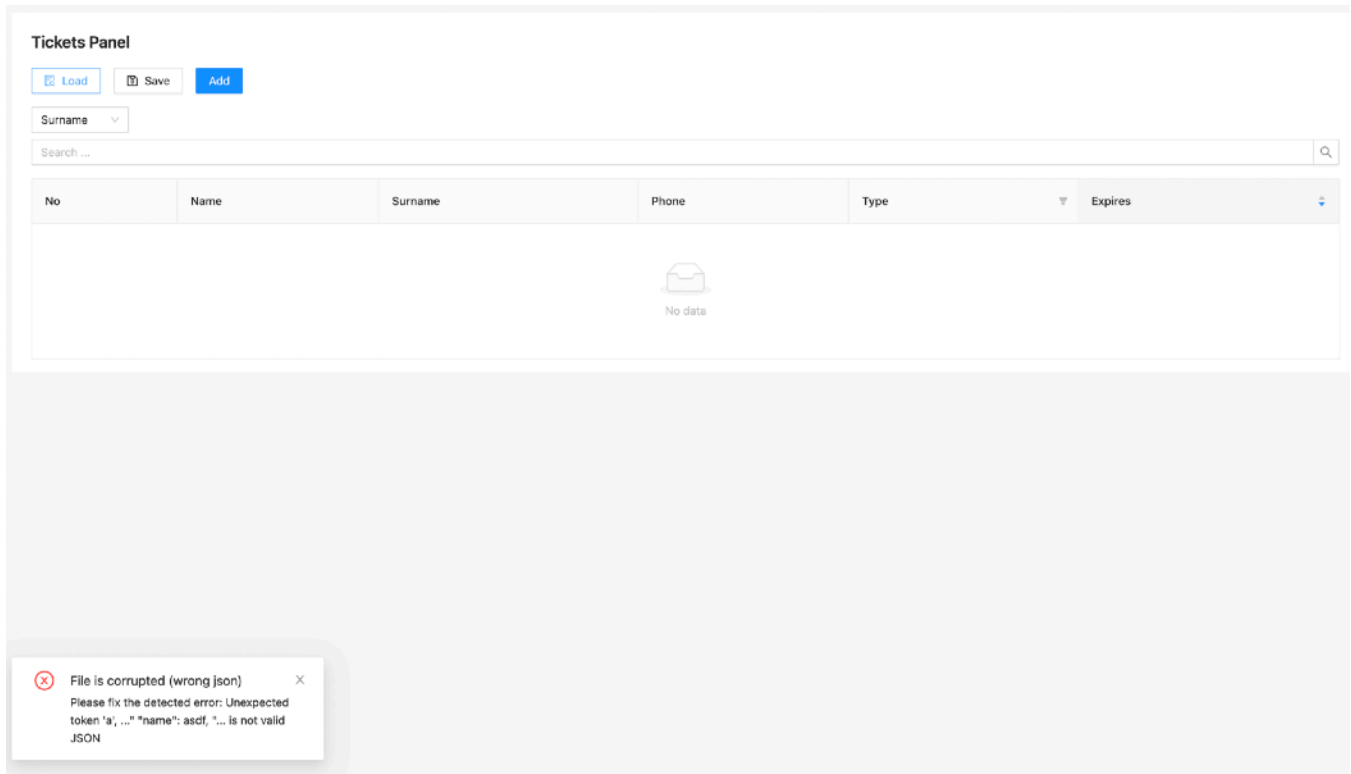
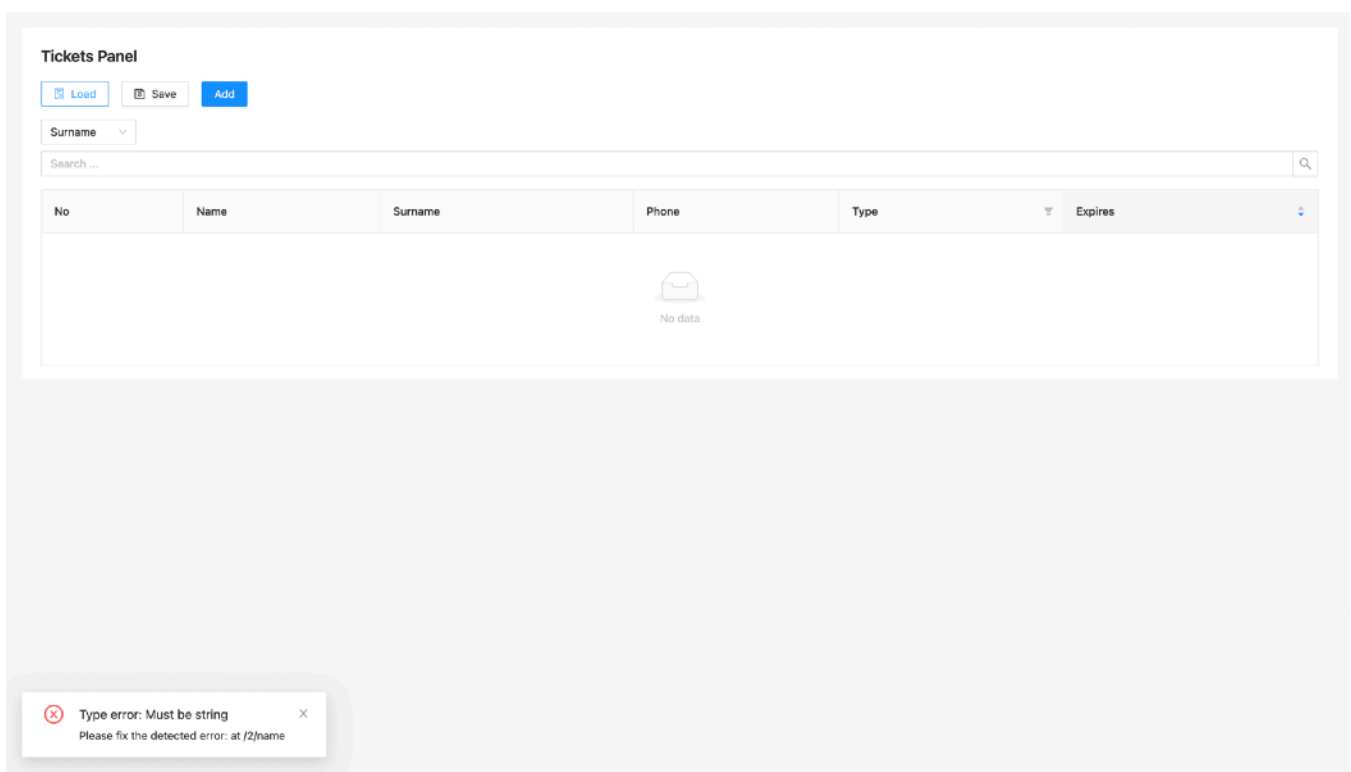


Рис. 16 – виняткова ситуація

6. Число замість стрічки для імені при імпорті



Інструкція для користувача

Призначення

Метою створення програми є відображення даних про квитки та їх власників. Це має полегшити облік квитків для транспорту та музеїв.

Компоненти ПЗ

Програму розроблено мовою Typescript в термінальному редакторі тексту vim. Програма розрахована на користувачів ОС на яку можна встановити сучасний браузер.

Вимоги до програмного та апаратного забезпечення:

Операційна система: MacOS/Windows xp+/Android/IOS/Linux.

Мінімальний обсяг ОЗП: 0.5Gb.

Мінімально необхідний простір на диску: 32 Мб.

Процесор: 32-розрядний з мінімальною тактовою частотою 1,5 ГГц.

Монітор: мінімальна роздільна здатність 600x400

Периферійні пристрої: клавіатура та миша

Інсталяція ПЗ

Перед використанням веб додатку, потрібно встановити сучасний веб браузер

Робота з програмою

Для запуску веб додатка, потрібно перейти за веб посиленням

<https://cursova-oop-keep-simple.vercel.app/>

(відкрити його у Браузері). Робота з програмою не вимагає досконалих навичок роботи з програмним забезпеченням та орієнтована на пересічного користувача. Нижче наведені підказки для роботи з даним ПЗ.

Tickets Panel

Load

Save

Add

Adult ticket is the most used

4

No

Search ...

No	Name	Surname	Phone	Type	Expires
4	Nikita	Yasnohorodskiy	984915369	Adult	2026-12-12
2	Olha	Pepe	914810563	Child	2025-09-11
3	Dmitryi	Svizi	904910566	Adult	2025-09-11
1	Dmitryi	Svizi	904910566	Adult	2023-11-05
0	Nikita	Yasnohorodskiy	<input type="text" value="984915369"/>	Adult	2022-12-12

<

1

>

Рис. 22 – Графічний інтерфейс програми

Для створення нового квитка, натисніть кнопку “**Add**” та заповніть новий рядок який з’явився в таблиці або імпортуйте файл (кнопка “**Load**”).

Для фільтрування або пошуку, скористайтесь іконками на назві стовпчиків таблиці та вводом для пошуку над таблицею.

Ви також можете експортувати дані з таблиці, натиснувши кнопку “Export”

Висновки

Під час виконання цієї курсової роботи я засвоїв навички розробки програмного забезпечення мовою програмування Typescript. Я закріпив знання створення графічних інтерфейсів за допомогою веб бібліотеки React.

Я також закріпив свої знання в об'єктно-орієнтованому/функціональному програмуванні та шаблонах проектування.

Я оновив знання створення UML-діаграм, як-от: діаграма прецедентів, діаграма послідовностей, діаграма класів.

Під час виконання курсової я використав декілька структур даних, як-от список та мапа.

Для розширення проєкта буде доцільно додати інтеграцію із реляційною базою даних.

Список літератури

- Об'єктно-орієнтоване програмування: методичні вказівки до виконання курсової роботи для студентів напряму 6.121 «Інженерія програмного забезпечення» / Укл. Коротєєва Т.О., Дяконюк Л.М.– Львів: Національний університет “Львівська політехніка ” кафедра програмного забезпечення, 2020. – 27с.
- Патерни проектування. Рефакторінг.Гуру. (n.d.). Цитовано: Жовтень 24, 2022, <https://refactoring.guru/design-patterns>
- Офіційна онлайн документація веб бібліотеки React - <https://reactjs.org/docs/react-api.html>
- Офіційна онлайн документація веб фреймворку Next.js - <https://nextjs.org/docs/getting-started>
- Офіційна онлайн документація бібліотеки веб компонентів Ant Design - <https://ant.design/docs/react/introduce>