Національний університет “Одеська політехніка”

Кафедра комп’ютеризовані системи та програмні технології

**КУРСОВА РОБОТА**

з дисципліни «Сучасні інтернет-технології»

Варіант 13

Студента 1 курсу, групи АТ–212

спеціальності «Автоматизація та комп'ютерно-інтегровані технології»

Виходець Р. С.

Керівник: доц. Сперанський В.О.

Національна шкала:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оцінка: ECTS \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

     Члени комісії  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   \_\_\_\_\_Сперанський В.О.\_\_\_\_\_

(підпис)          (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_    \_\_\_\_\_Ілуца А.С.\_\_\_\_\_\_\_\_\_\_\_

(підпис)          (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_     \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис)          (прізвище та ініціали)

м. Одеса – 2022 рік

1. Спроектувати структуру інформаційної бази свого варіанта завдання (Додаток А).
2. Створити основну таблицю інформаційної бази як динамічного списку об'єктів.
3. Передбачити функції контролю коректності всіх даних, що вводяться (наприклад, невід'ємний вік, дата продажу не раніше, ніж 100 років тому тощо).
4. Заповнити основну таблицю щонайменше 20 записами.
5. Забезпечити збереження та завантаження даних у зовнішній файл/з файлу.
6. Реалізувати інтерфейс користувача, який забезпечуватиме наступні функції:
7. створення інформаційної бази тієї предметної галузі, яка визначена постановкою завдання;
8. додавання нових записів до бази даних (з контролем коректності значень, що вводяться) в окремому вікні (модальному);
9. пошук записів по ключовому полю;
10. редагування записів бази даних у окремому вікні (модальному);
11. видалення непотрібних записів із підтвердженням дії;
12. збереження даних у файлі;
13. видачу звітів за запитом із можливістю сортування;
14. реалізувати систему нотифікацій в інтерфейсі, яка повідомлятиме про успішність/провальність дій в інтерфейсі (додавання запису, видалення тощо
15. Виконати оригінальну обробку, вказану у постановці конкретного завдання для вашого варіанту (Додаток А).
16. Організувати введення пароля під час операцій змін даних.
17. Додати довідкову систему користувача за створеною програмою.
18. Підготувати пояснювальну записку за розробленим проектом.

**ДОДАТОК (А)**

Розробити інформаційну базу автоматизації роботи багажного відділення аеропорту.

Кожен запис містить:

- прізвище, ім'я та по батькові пасажира;

- номер рейсу;

- Номер багажної квитанції;

- кількість місць багажу;

- Сумарна вага багажу пасажира;

- час розміщення багажу;

- термін розміщення.

*Передбачити:*

а) видачу відомостей про пасажирів, вага багажу яких перевищує 30 кг;

б) видачу відомостей про сумарну вагу багажу всіх пасажирів рейсу;

в) видачу відомостей про місцезнаходження багажу (номер рейсу) за заданим номером багажної квитанції.

г) видалення записів про пасажирів із заданим прізвищем.

**ЕТАПИ ПРОЕКТУВАННЯ**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Найменування | Зміст | Дата закінчення |
| 1. | Початок | Вивчення постановочного завдання, предметної галузі | 10.05.2022 |
| 2. | Теоретична підготовка | Вивчення літератури з дисципліни | 15.05.2022 |
| 3. | Підготовка базових класів | Створення класів для додавання, видалення в список, створення методів для коректного вводу, та генерації об’єктів | 21.05.2022 |
| 4. | Створення  бази даних | Створення класів з методами серіалізації/десеріалізації, заповнення файлу, та створення окремого файлу, який буде передаватися на локальний сервер | 25.05.2022 |
| 5. | Верстка користувацького інтерфейсу | Створення інтерфейсу для  виводу даних, сортування та  Адаптація для різних екранів | 11.06.2022 |
| 6. | Запуск локального сервера | Створення та налаштування  класів для сервера. Створення  функцій для запитів до сервера | 13.06.2022 |
| 7. | Реалізація зв’язку користувацького інтерфейсу з сервером | Ввід та вивід даних, видача  довідок, створення перевірок  даних, вводу пароля | 15.06.2022 |

Використані інструменти:

*Java, Html, Css, Js, Gson, HttpServer*

**ДЕТАЛЬНИЙ ОПИС ПРОЕКТУ**

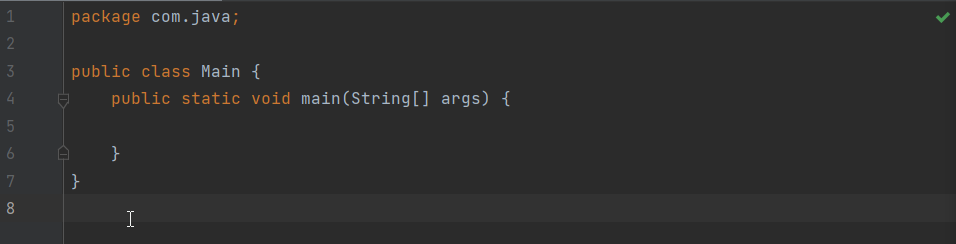
**BACK-END**

1.Створюємо поділи на папки(Рис. 1.1)



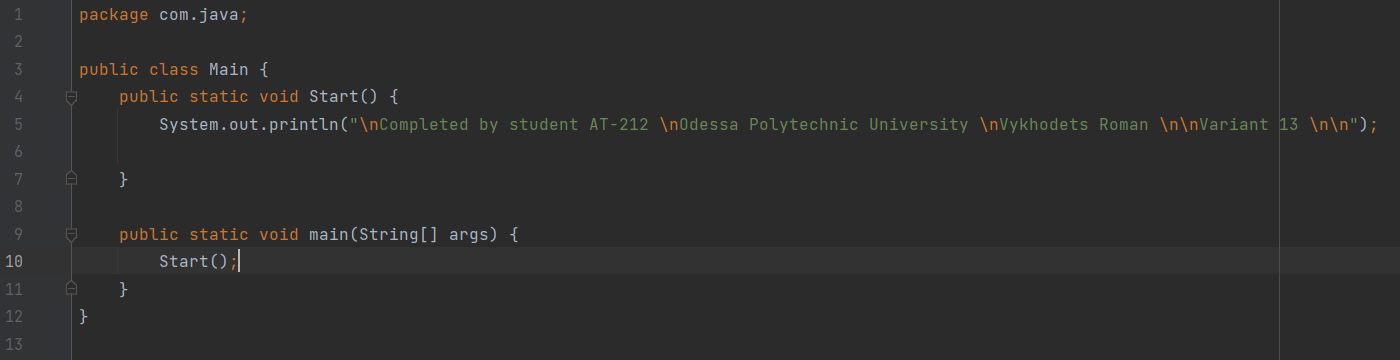
(Рис. 1.1)

В папці ***java*** створюємо наш основний файл ***Main.java*** та відкриваємо його (Рис.1.2)



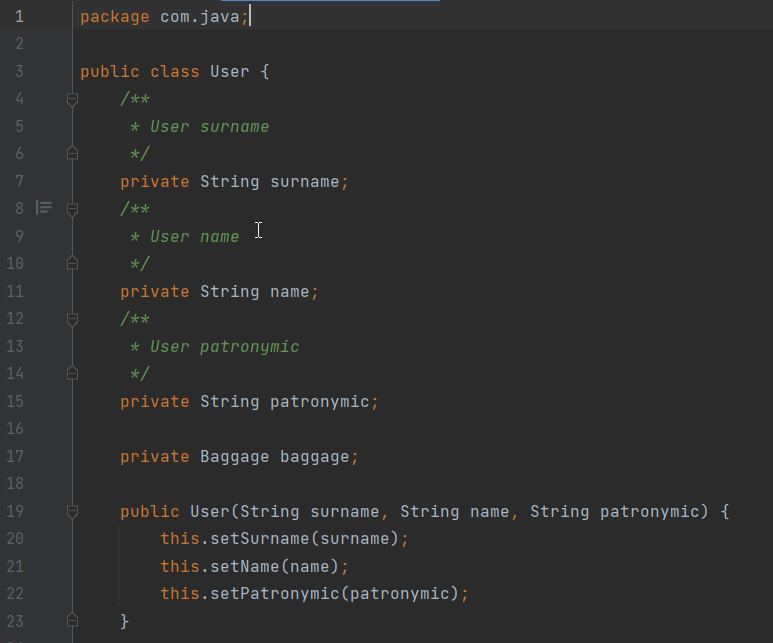
(Рис.1.2)

Додаємо функцію ***Start()*** в якій буде знаходитись інформація про виконавця роботи, та одразу її визиваємо у ***Main***(Рис. 1.3)



(Рис. 1.3)

Створення класу «User», з полями ім’я, прізвище, по батькові, додаємо їм Set і Get, формуємо toString, а також робимо конструктор(Рис. 1.4 та Рис.1.5)

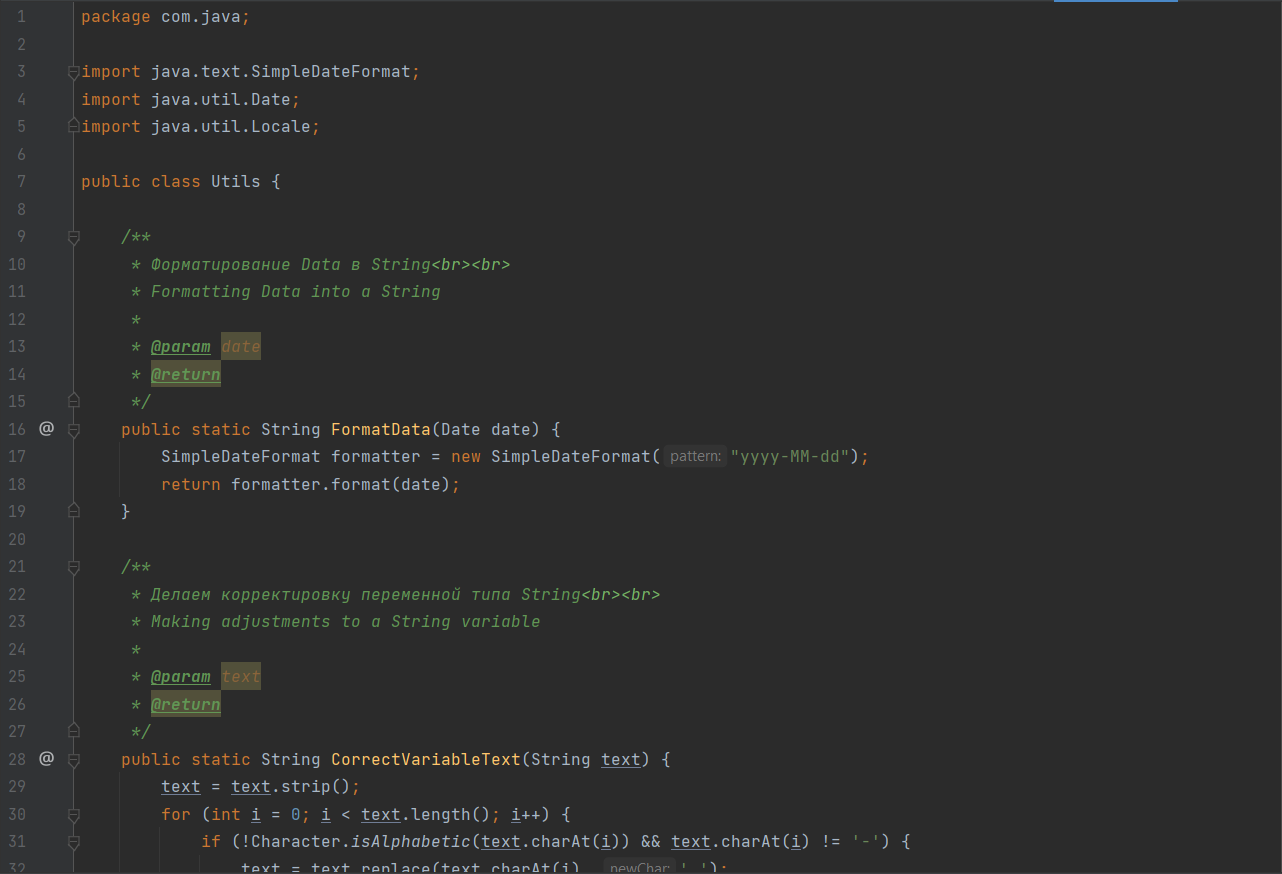


(Рис. 1.4)

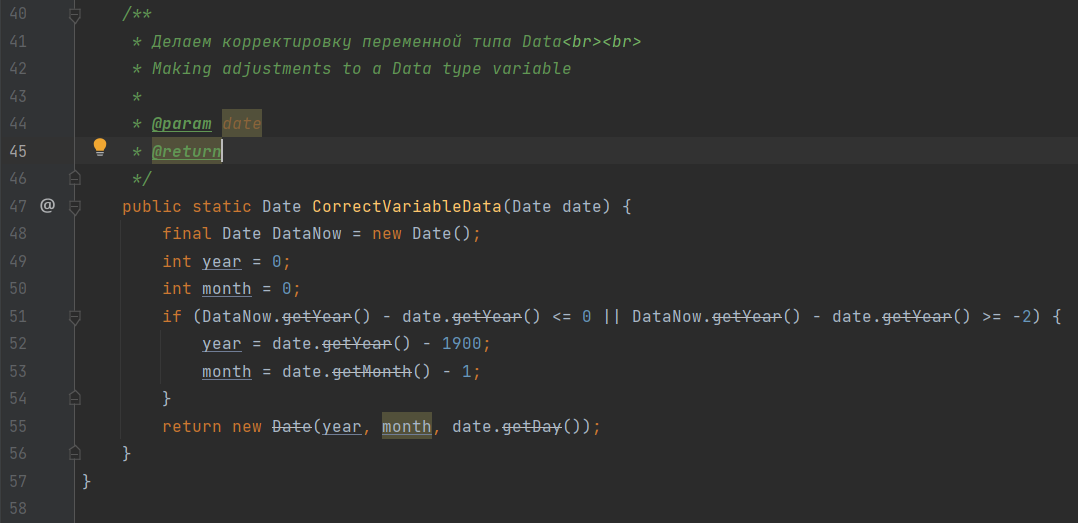


(Рис. 1.5)

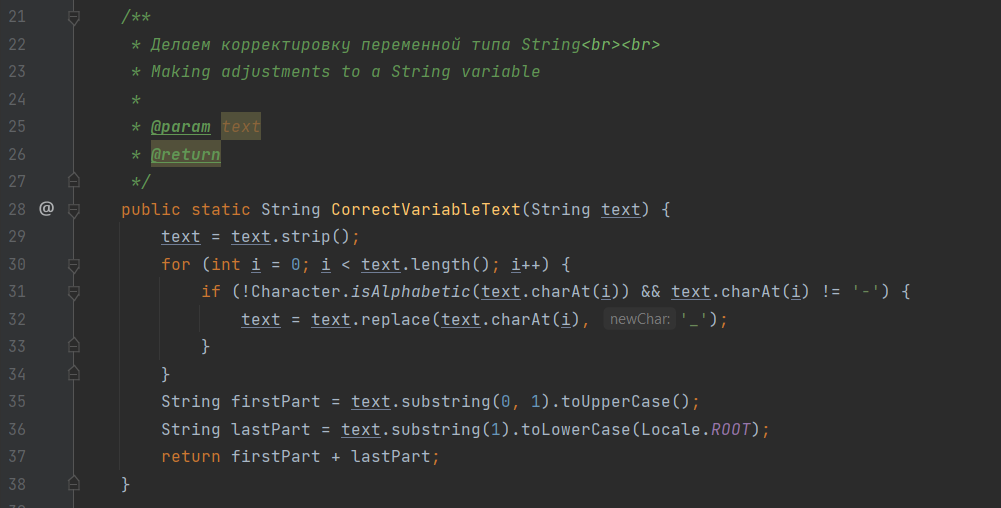
Створюємо клас ***Utils***, в якому будуть знаходитись різні доповнення, а саме корекція данніх(Рис. 1.6, Рис. 1.7, Рис. 1.8)



(Рис. 1.6)

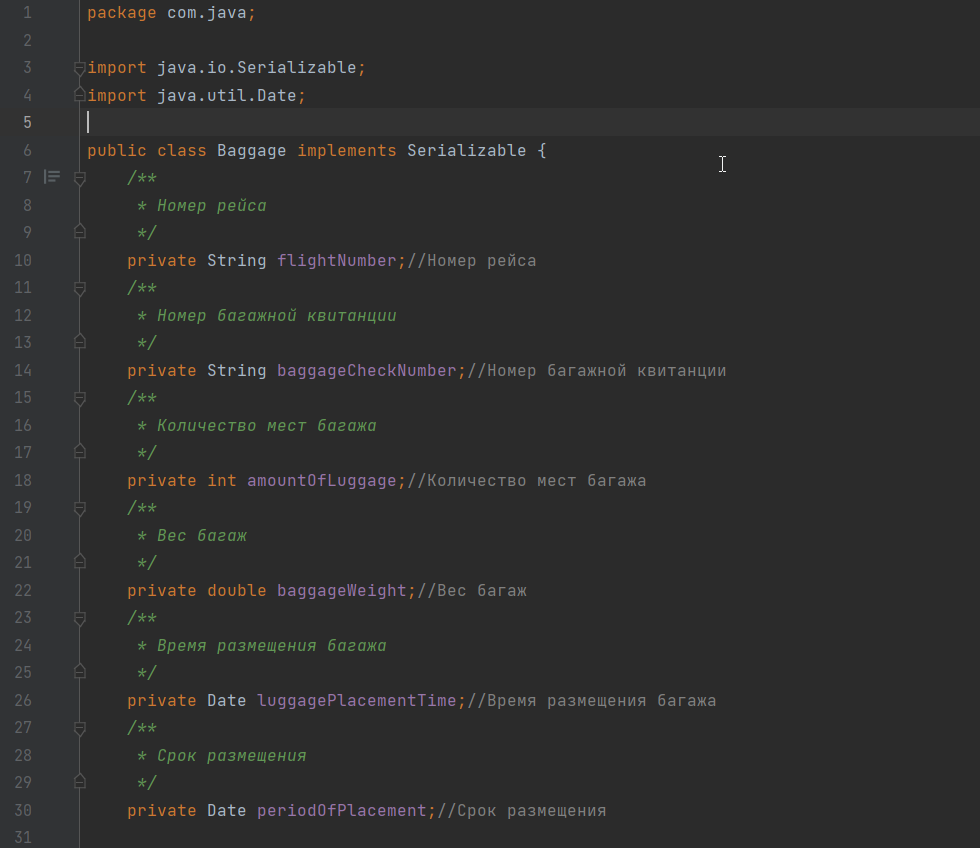


(Рис. 1.7)

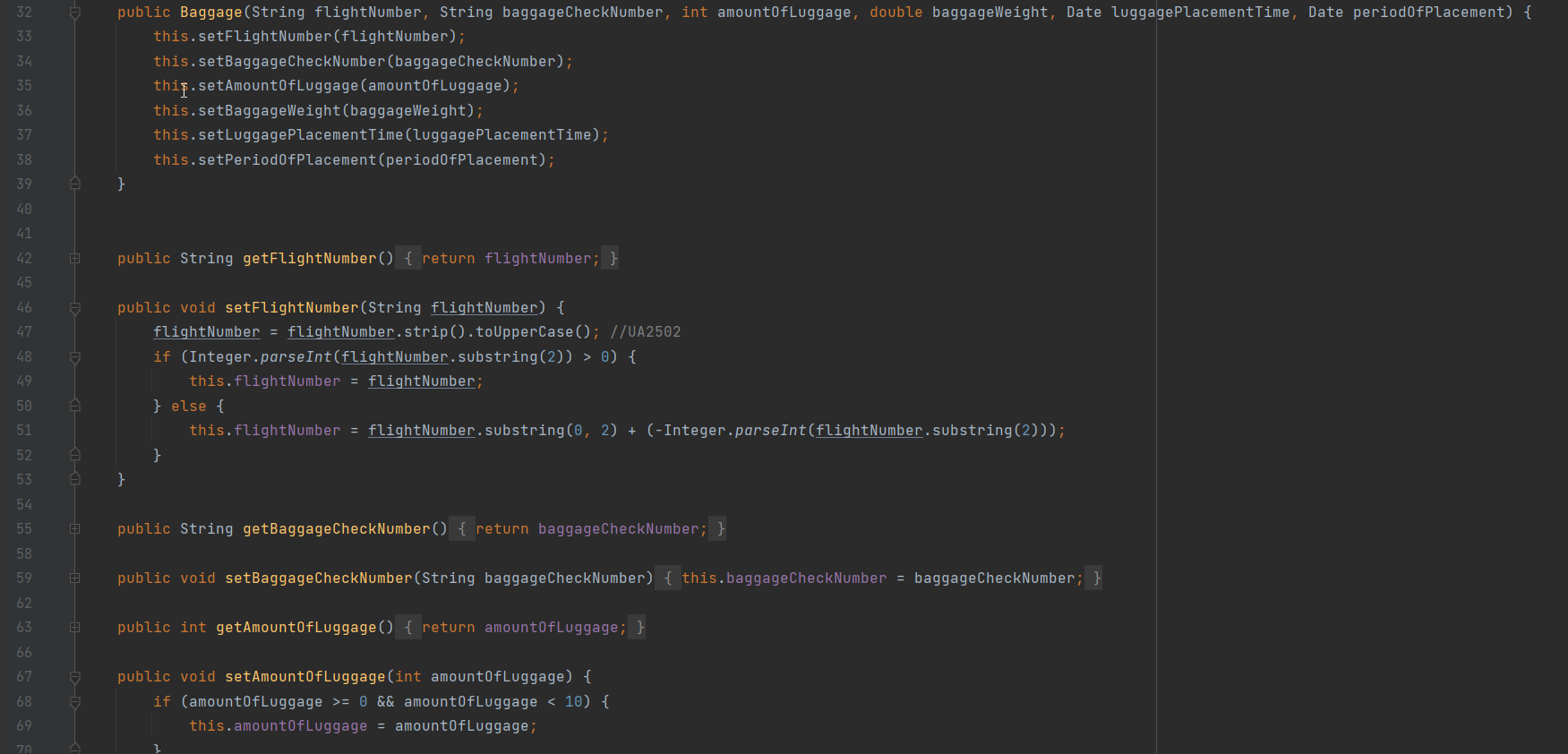


(Рис. 1.8)

Робимо клас в якому будуть знаходитись усі данні про багаж ***Baggage***, робимо теж саме що і з класом ***User***, також у класі ***User*** додаємо Об’єкт типу ***Baggage***, для того, щоб отримувати данні багажа.(Рис. 1.9 – 1.14)



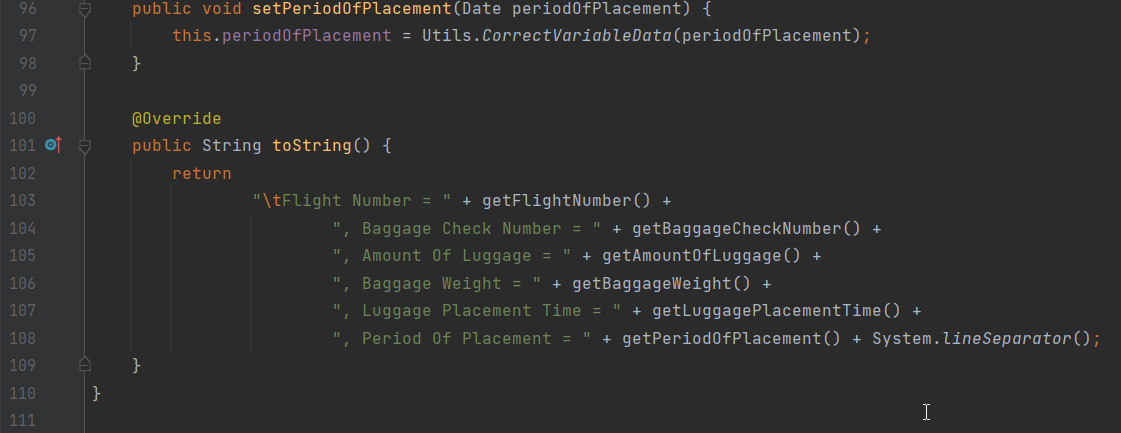
(Рис. 1.9 створення та заповнення змінними(Номер рейса, номер багажного квитка, кількість багажу, вага багажу, час розміщення багажа та срок розміщення))



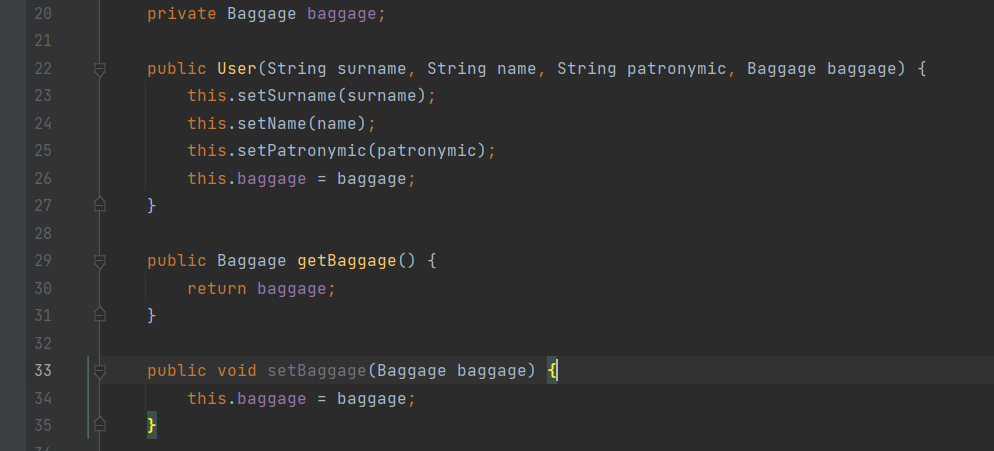
(Рис 1.10 створення конструктора, перевірки вхідних данних у ***Set*** та вихідних в ***Get***)



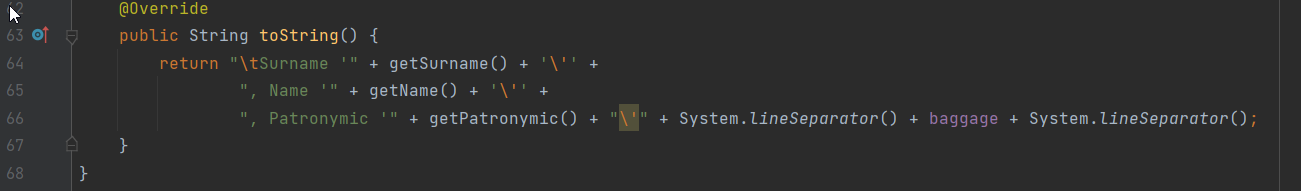
(Рис. 1.11)



(Рис. 1.12 Формуємо ***toString***)

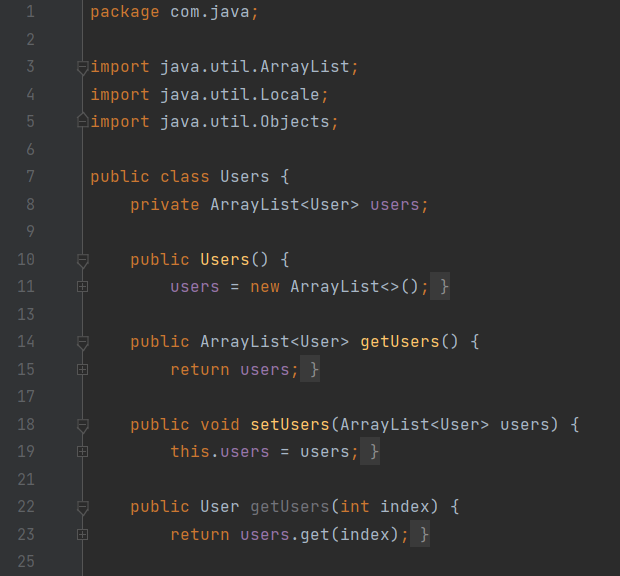


(Рис. 1.13 Зміни в класі ***User***)

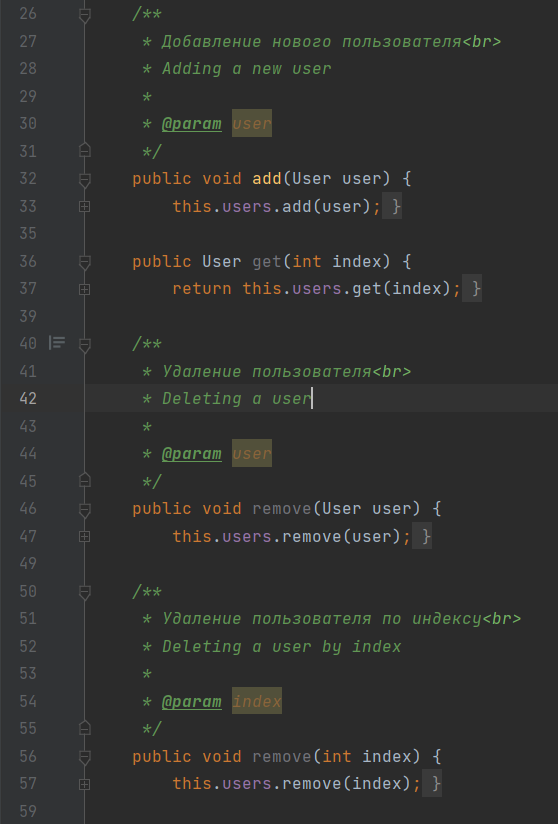


(Рис. 1.14 Зміни ***toString*** в класі ***User***)

2. Для того, щоб ми змогли створювати список, ми робимо окремий клас ***Users***, в якому додаємо ***ArrayList***, setUsers, getUsers, робимо дві функції (для додавання “***add”***, для віднімання “***remove”*** та для виводу даних по індексу ”***get***”). (Рис. 2.1, Рис.2.2)



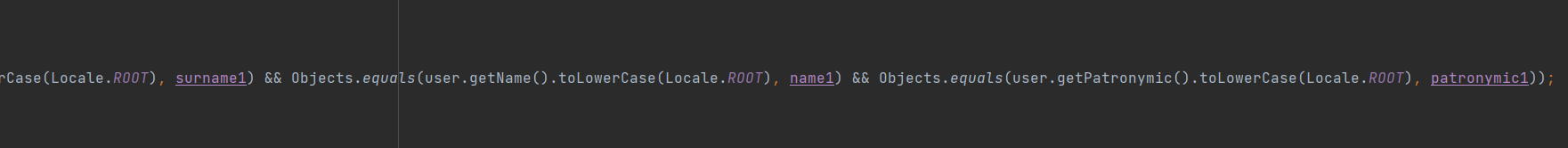
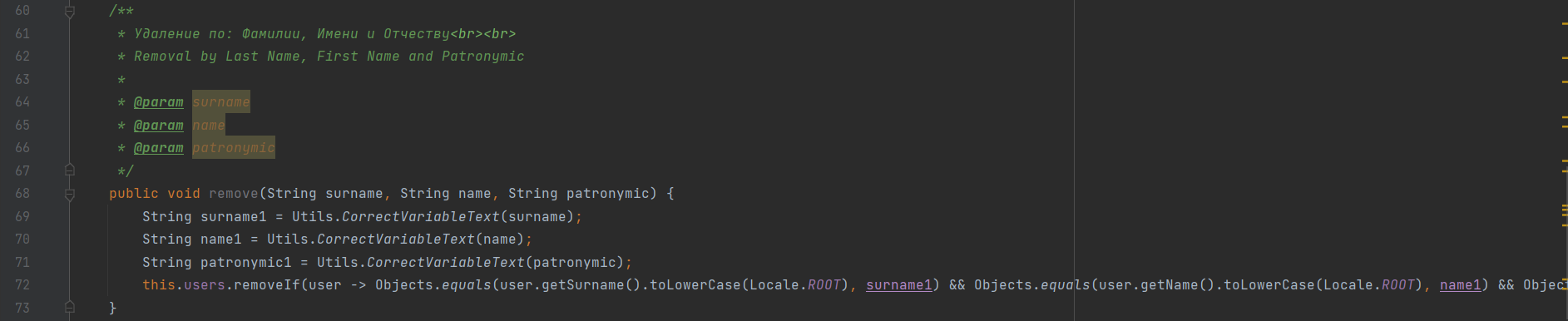
(Рис. 2.1)



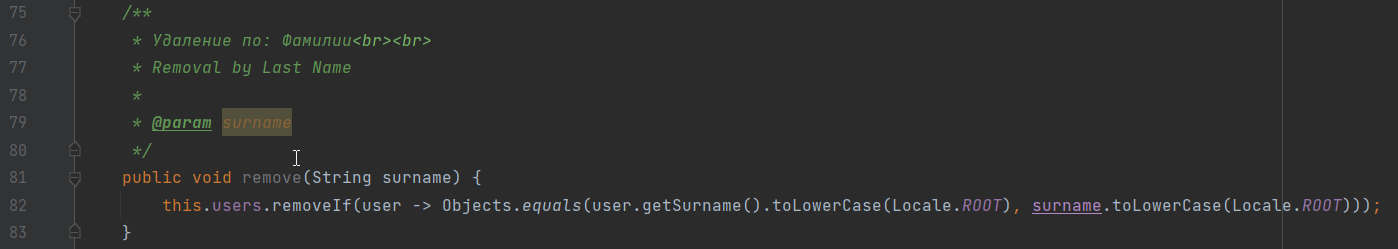
(Рис. 2.2)

Також не забудемо про завдання яку нам було дано, а саме:

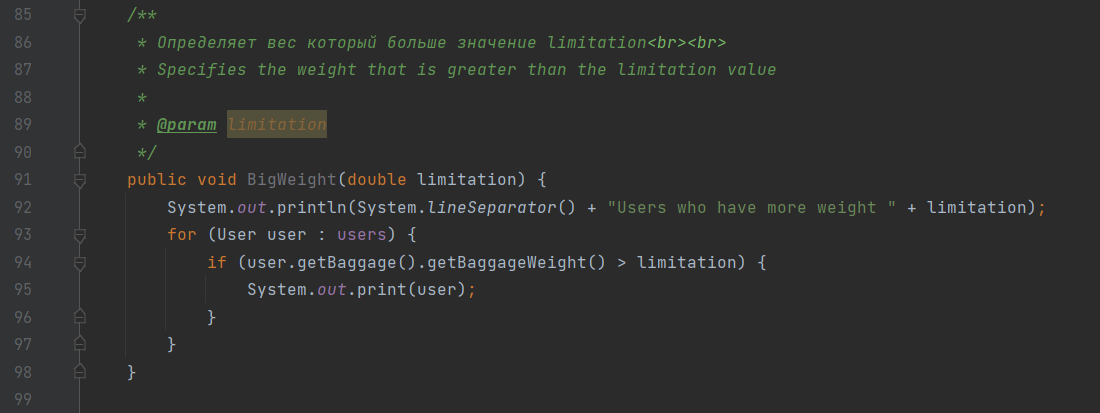
(Рис 2.3 - 2.7)



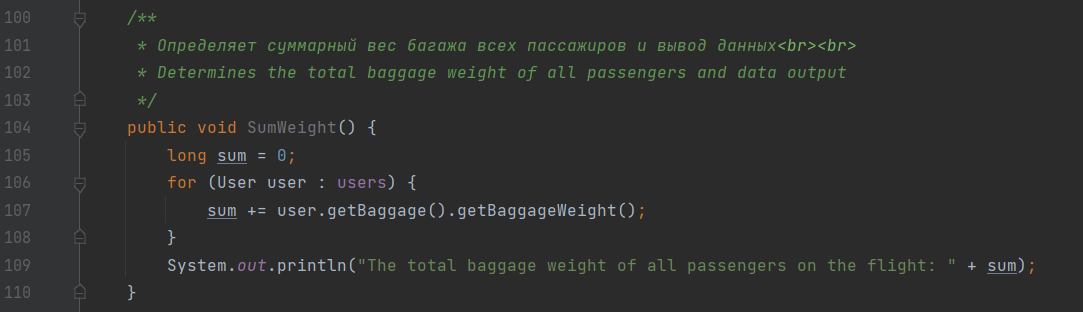
(Рис. 2.3 видалення пасажира по його ***Імені***, ***Прізвищу*** та ***По Батькові***)



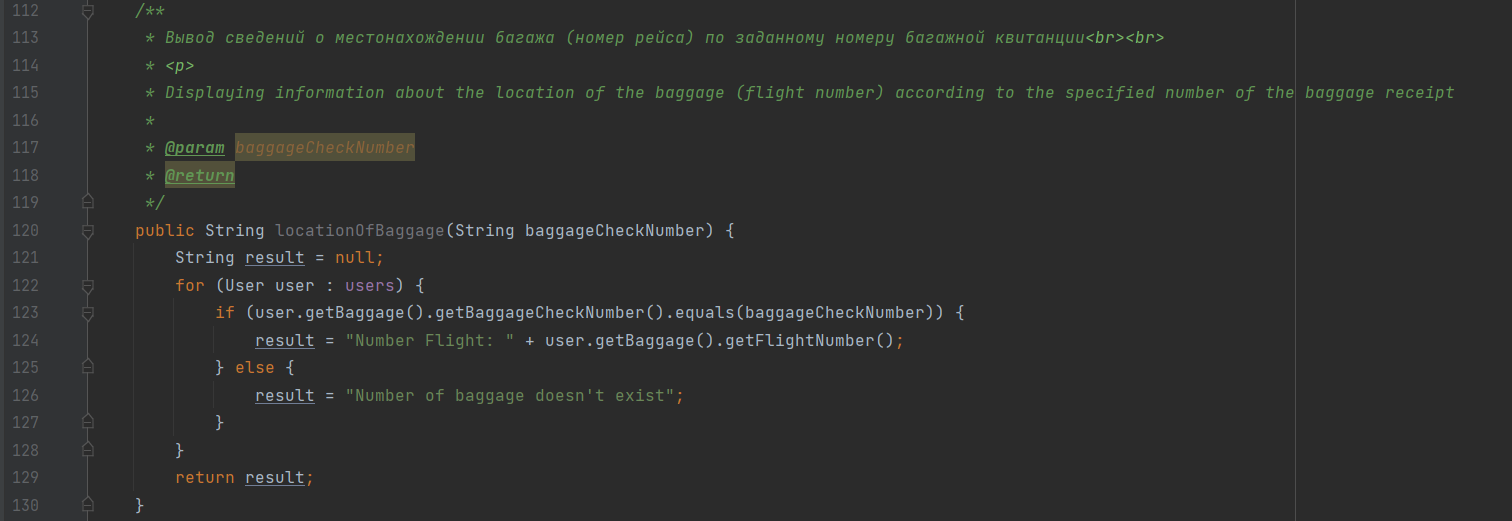
(Рис. 2.4 видалення пасажира виключно по його ***Прізвищу***)



(Рис. 2.5 Визначення та отримання даних пасажирів, які мають вагу багажа вище 30кг)

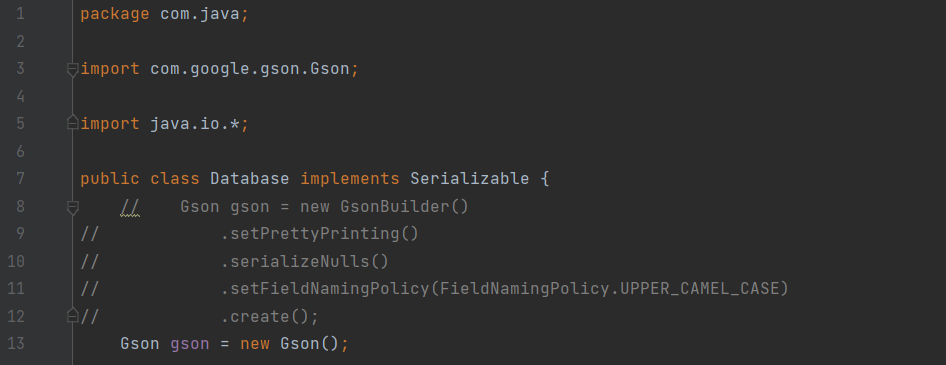


(Рис. 2.6 Визначення сумарної ваги усього багажа пасажирів)

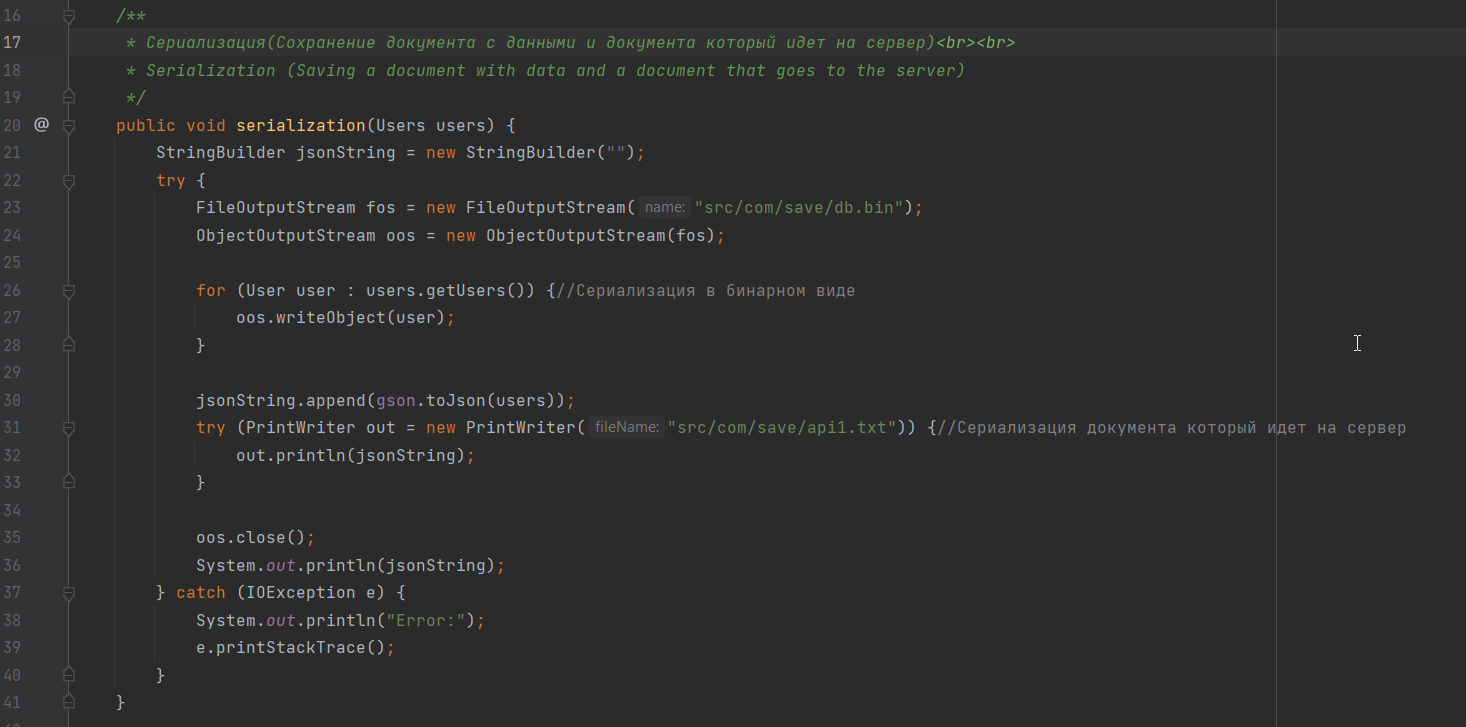


(Рис. 2.7 Визначення інформації о місце знаходжені багажа)

3. Створюємо клас ***Database*** для того, щоб реалізувати методи ***serialization*** та ***deserialization***(Рис. 3.1 - 3.4)

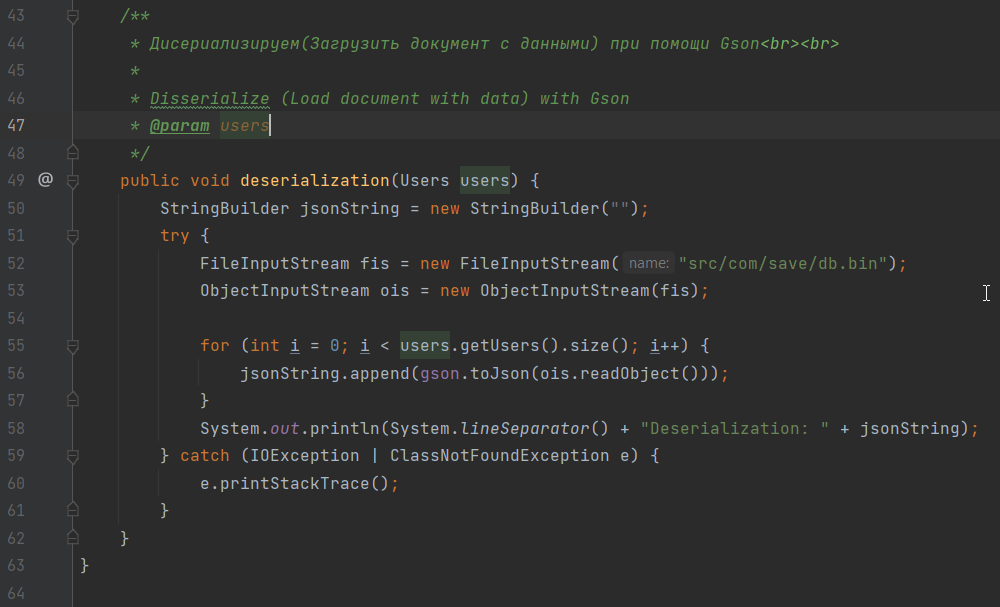


(Рис. 3.1 додаємо бібліотеку ***Gson*** та implements ***Serializable***)



(Рис. 3.2 Робимо сіреалізацію та створюємо два файли, перший – бінарний, який ми будемо використовувати у десіреалізації, а другий – це файл який буде іти на наш локальній сервер)

У першому файлі ми використовуємо нативний спосіб, а у другому використовуємо бібліотеку ***Gson***

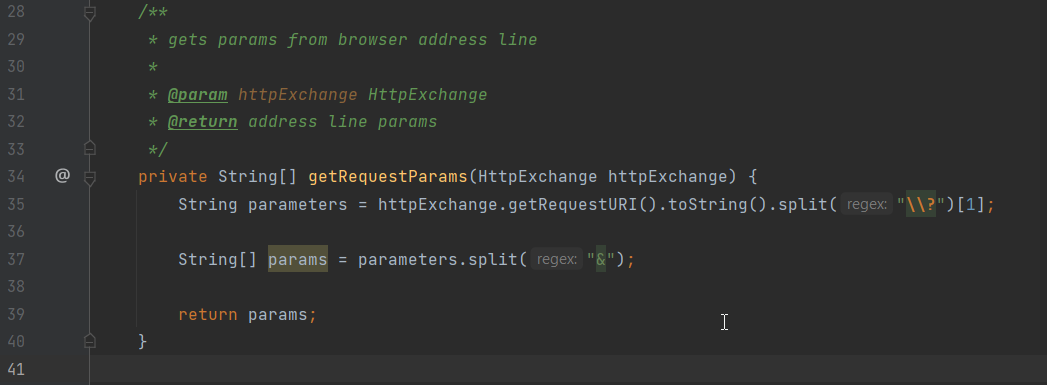


(Рис. 3.4 Робимо десіреалізацію завдяки бібліотеці ***Gson***)

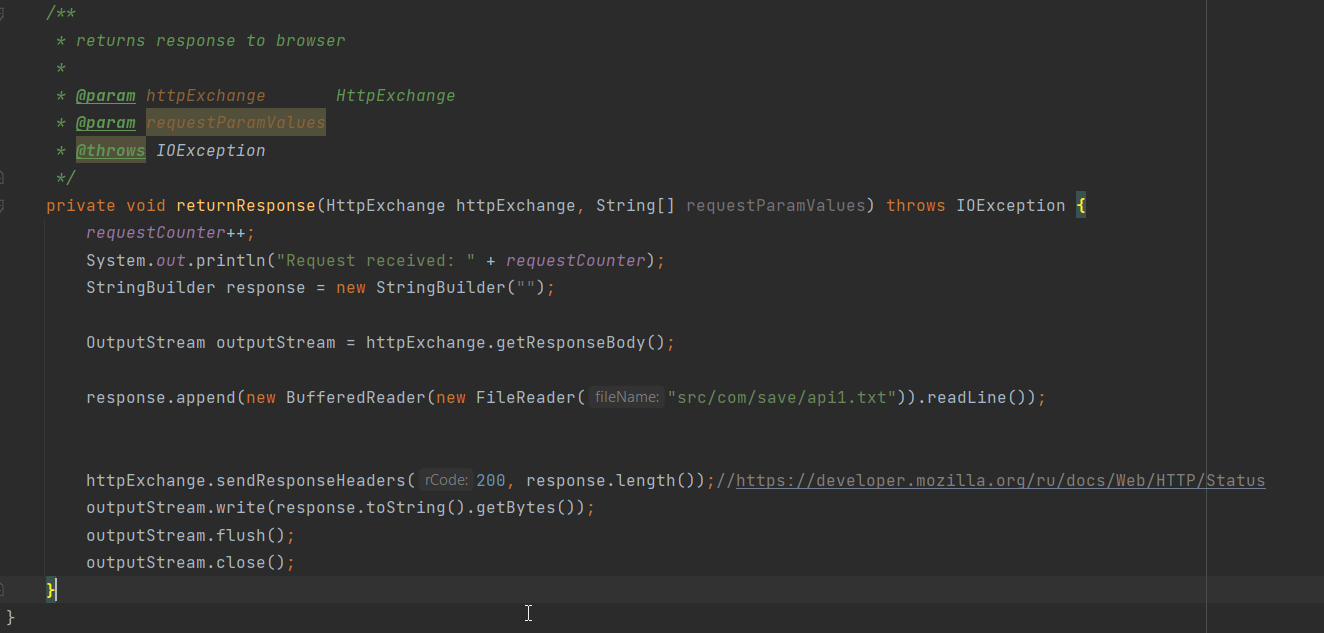
4. Створюємо клас ***HttpServer***, який буде керувати сервером, одразу імплементуєм ***HttpHandler*** (Рис.)



(Рис. 4.1 перевизначаємо функцію ***handle***)



(Рис 4.2 завдяки цієї функції ми можемо отримати данні з строки посилання і записуємо їх у масив типа ***String***)



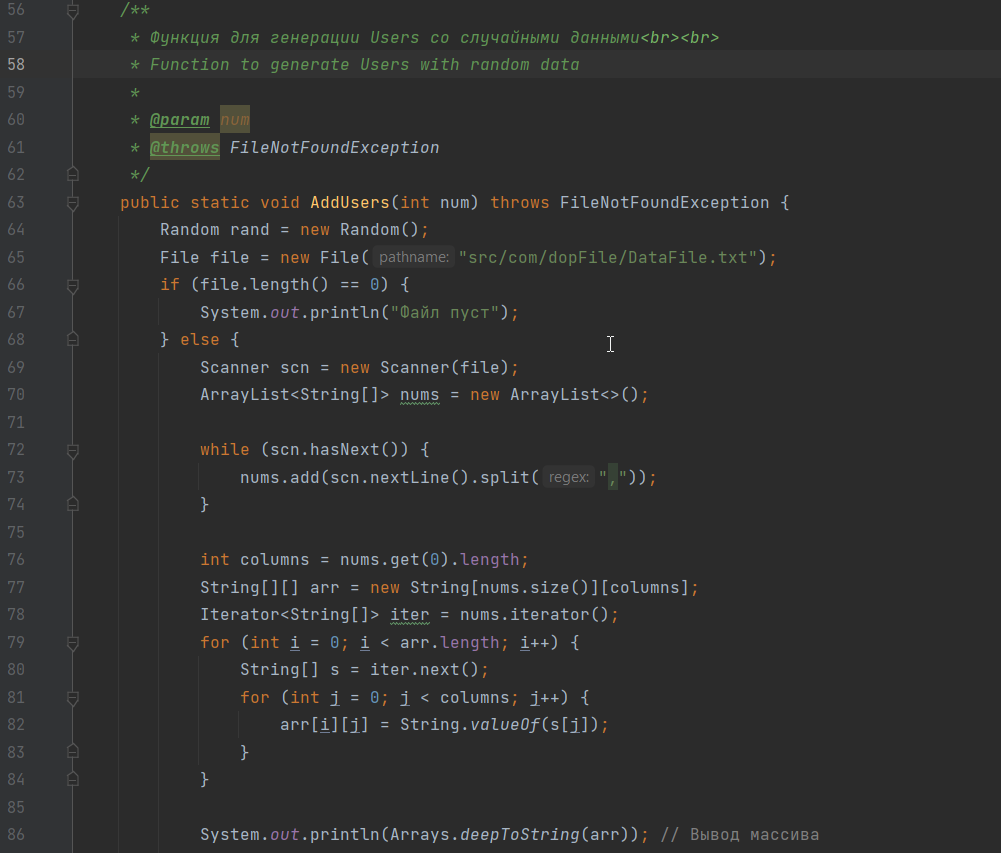
(Рис. 4.3 ця функція виводить наші дії на сторінку сервера, а саме створюємо змінну формату ***StringBuilder***, додаємо до отриманої строки вміст нашого файлу ***ap1.txt*** )

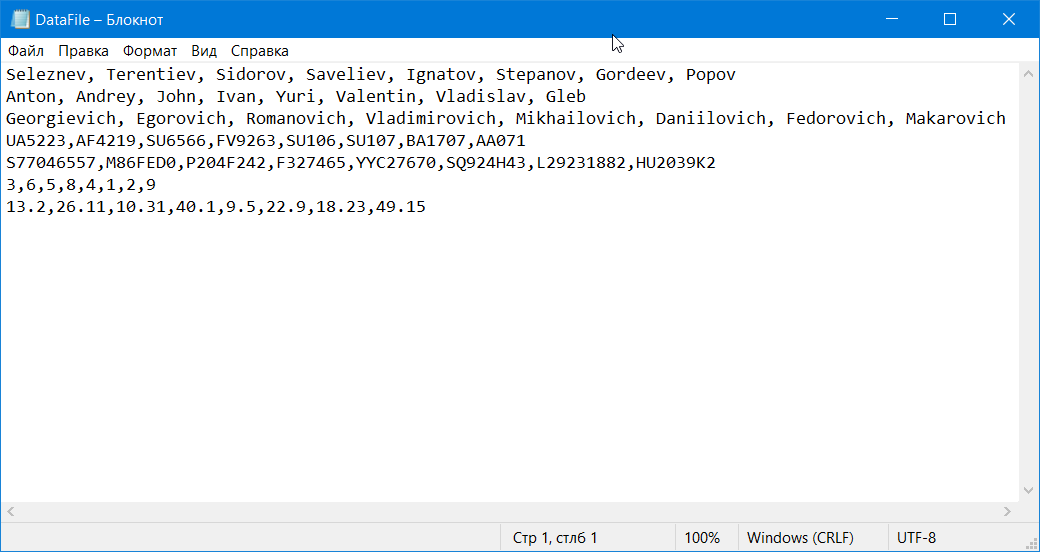
Далі повертаємося до файлу ***Main***. Створюємо функцію, яка буде запускати наш локальній сервер(Рис. 4.4).



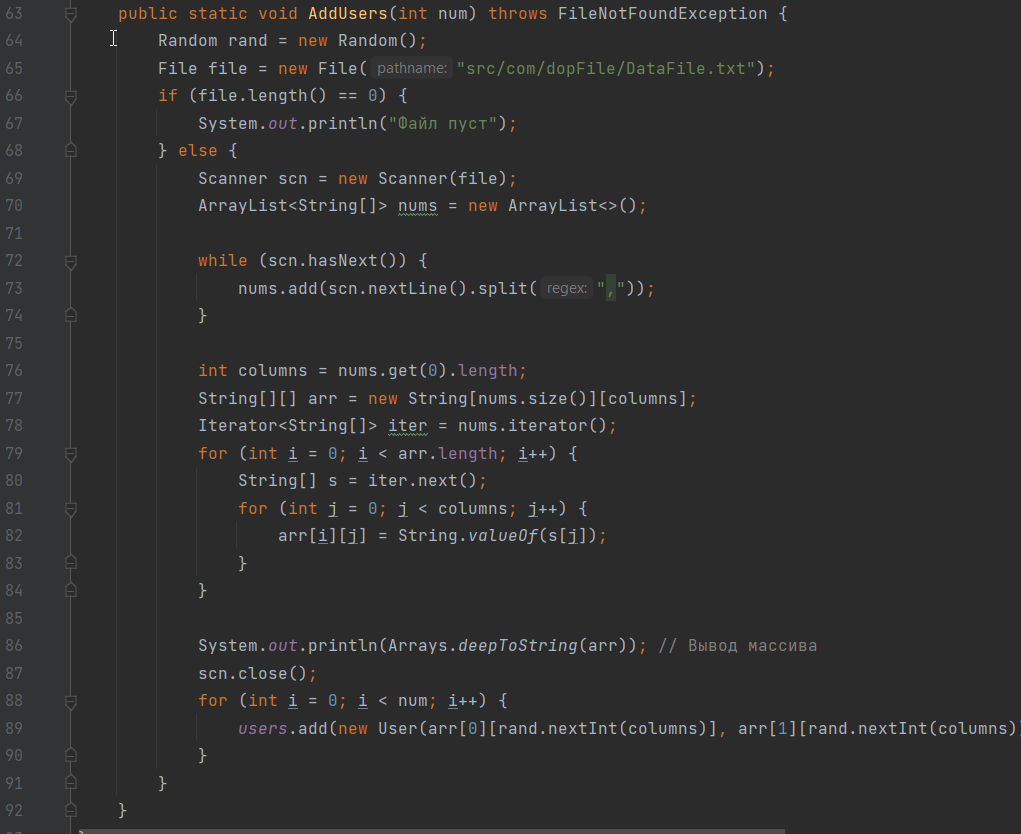
(Рис. 4.4 Запуск серверу)

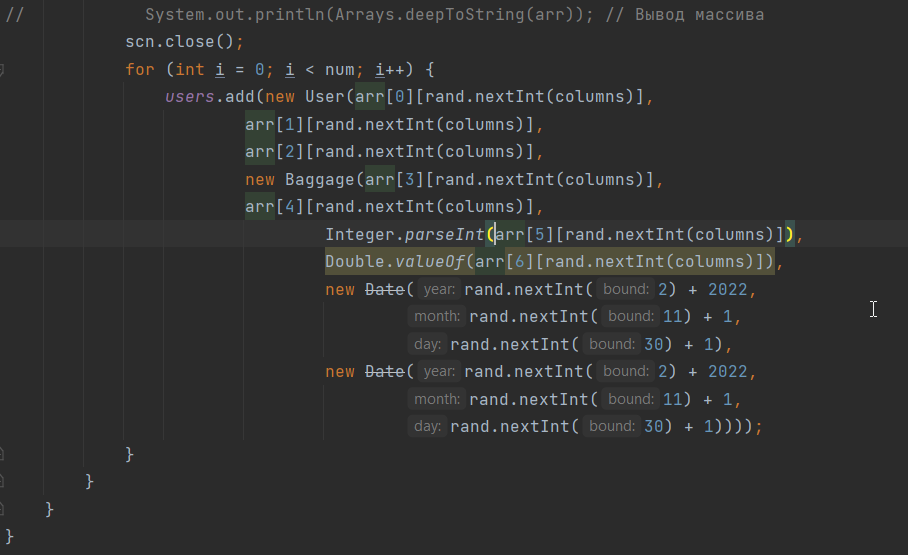
Додаємо функцію, яка генерує нам данні користувачів(Рис. 4.5, Рис. 4.7)

(Рис 4.5 У нас є файл ***DataFile.txt*** в якому збережені всі данні у декількох екземплярах,)



(Рис 4.6 Файл ***DataFile.txt***)



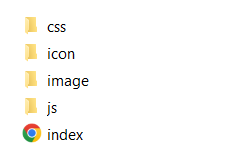


(Рис. 4.7)

Ми отримаємо строки, які поділяються на слова і записуються у масив, після цього ми завдяки циклу вказуємо кількість ***Users***

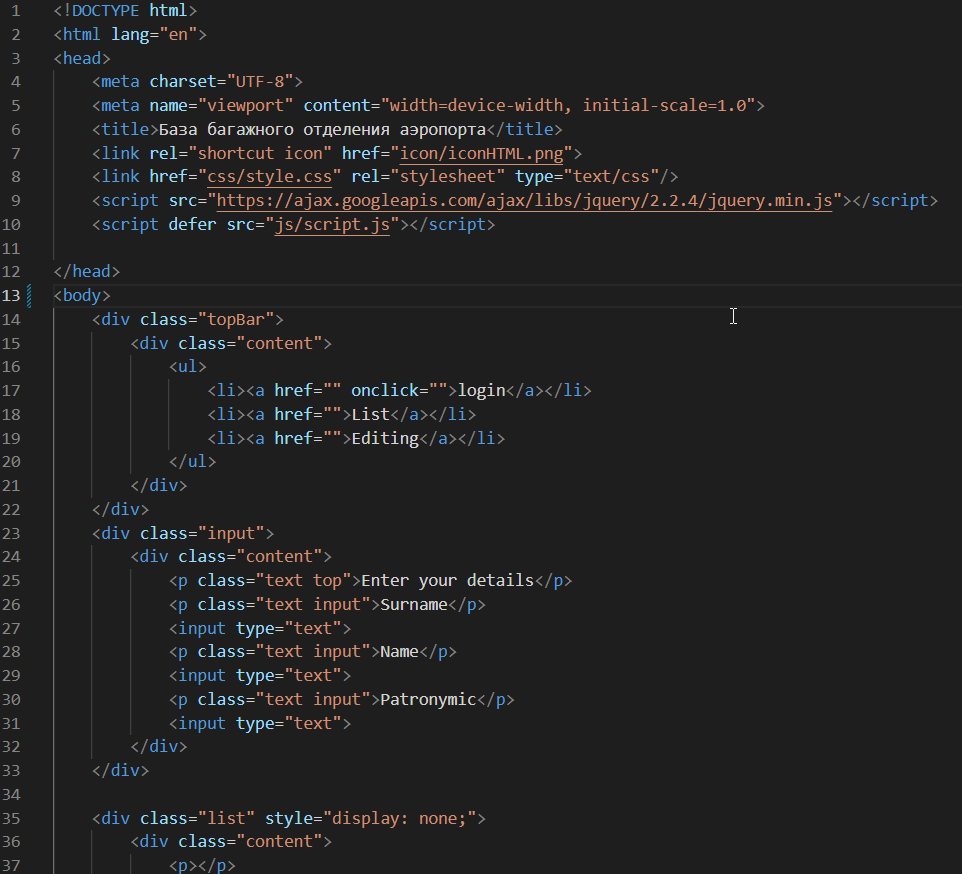
**FRONT-END**

5. Спочатку структуровано створюємо файли ***Html, Css, Js*** (Рис. 5.1)



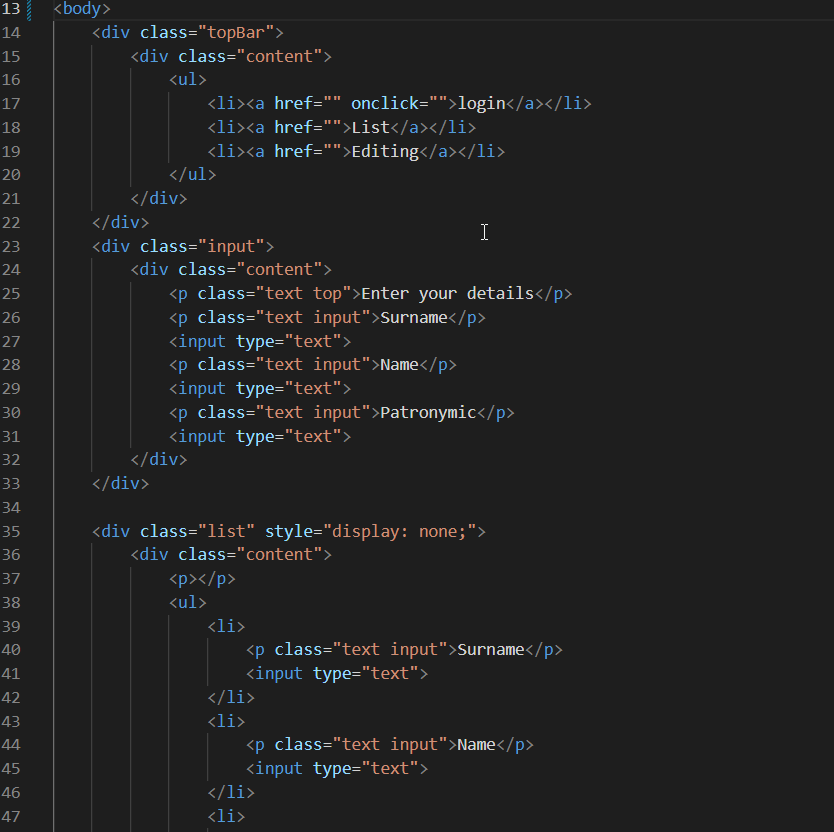
(Рис. 5.1 структура сайту)

Записуємо наступний ***Html-код***(Рис. 5.2 – Рис 5.4)



(Рис. 5.2)

У ньому ми робимо ***topBar*** в якому знаходяться кнопки управління. Клас ***input*** – це div в якому знаходиться панель для вводу даних. У клас ***conText*** передаються данні з ***api***

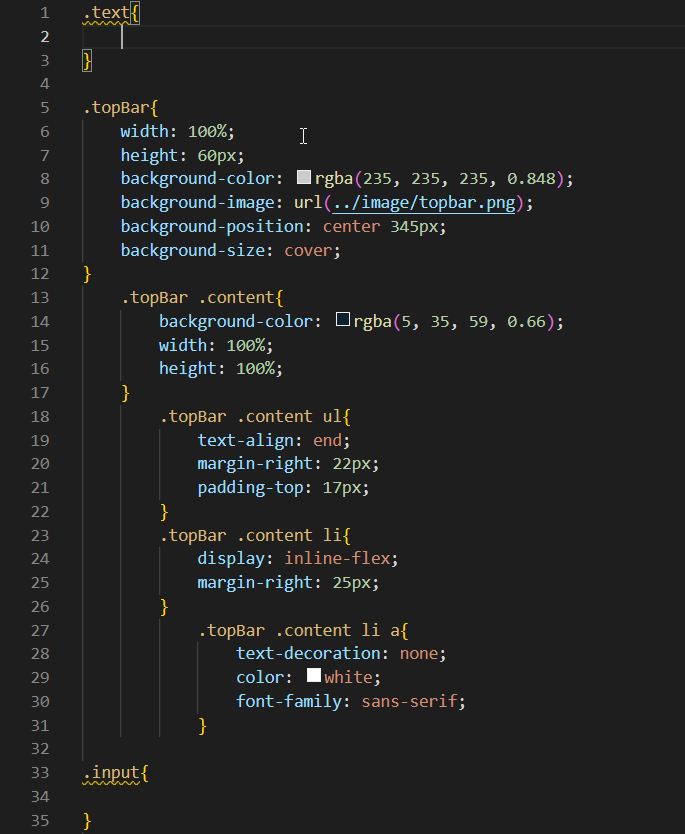


(Рис. 5.3)

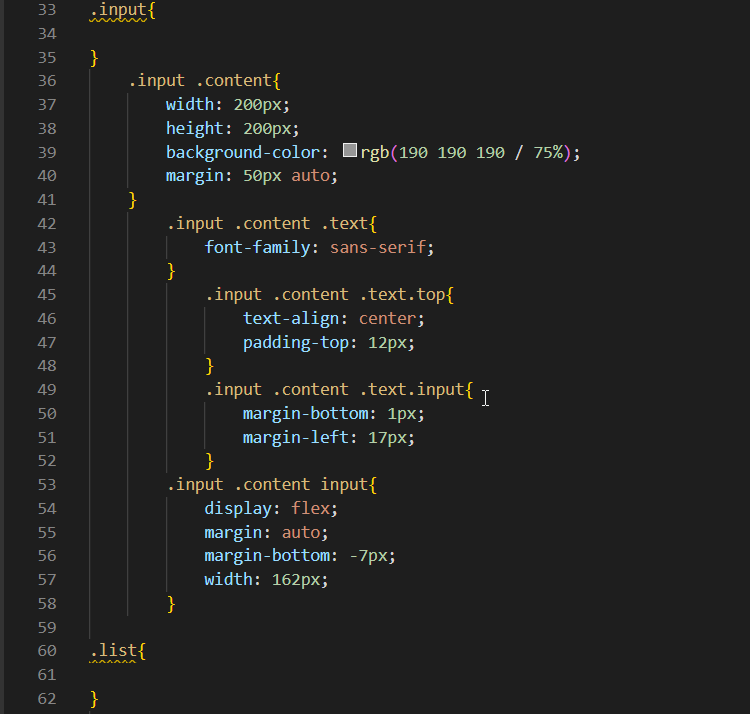


(Рис. 5.4)

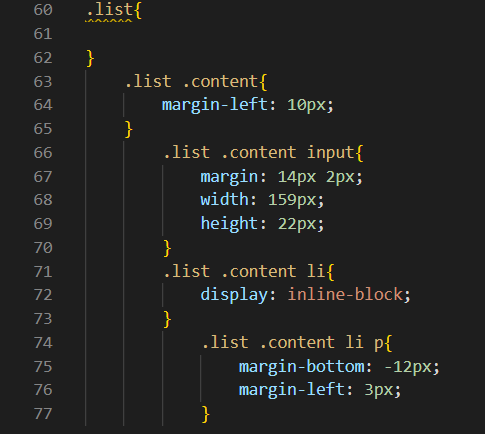
Далі пишемо ***CSS***(Рис. 5.5 – Рис. 5.7)



(Рис. 5.5)

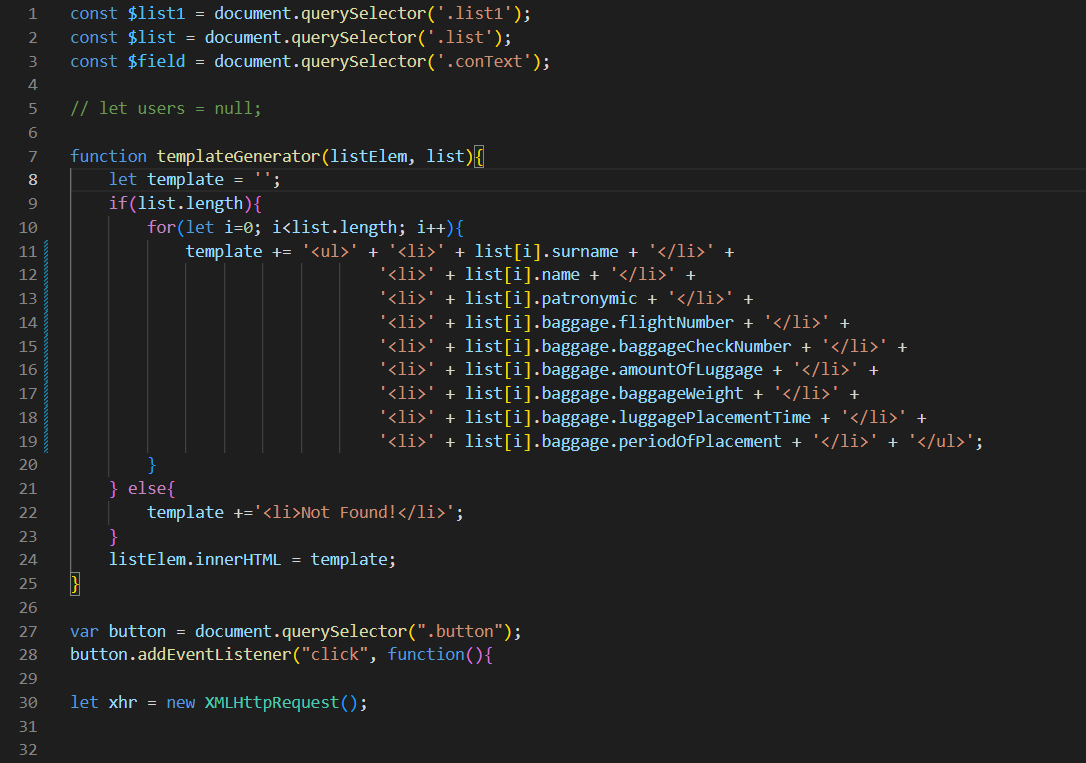


(Рис. 5.6)



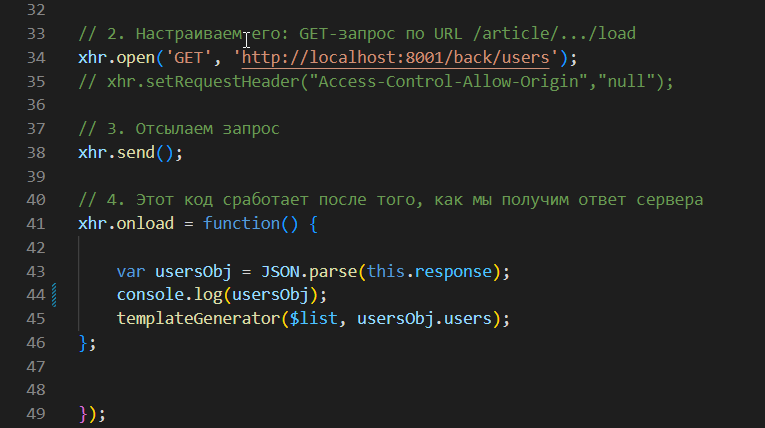
(Рис 5.7)

Ну і звісно ***JS***(Рис. 5.8 – 5.9)



(Рис. 5.8)

У функції ***templateGenerator*** ми зберемо отримані дані з ***АРІ*** і формуємо Html-код.



(Рис. 5.9)

**ВИСНОВОК**

При виконанні даної курсової роботи, була створена база даних та веб-додаток для доступу, перегляду, та редагуванню даних. При виконанні даної роботи було використано такі курси як: «Сучасні інтернет технології» та «Сучасні технології програмування»

Також було отримані навички використання та взаємодії з локальним сервером.

**ПЕРЕЛІК ЛІТЕРАТУРИ:**

# 1. Коды ответа HTTP:

# <https://developer.mozilla.org/ru/docs/Web/HTTP/Status>

# 2. Java ObjectOutputStream Class:

<https://www.programiz.com/java-programming/objectoutputstream>

3. Using the fetch API:

<https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch>;

4. JSON in JS:

<https://learn.javascript.ru/json>

5. Приклад HttpServer:

<https://github.com/it-school/SimpleServer.git>

6. Gson:

<https://www.javadoc.io/doc/com.google.code.gson/gson/2.8.0/com/google/gson/Gson.html>

7.Gson docs:

<https://github.com/google/gson>