МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами

Кафедра програмного забезпечення автоматизованих систем

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня програма «Інженерія програмного забезпечення»

«Допущений до захисту»

Завідувач кафедри

_____ проф. Приходько С.Б.

«___» _____ 2022 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти «бакалавр»

на тему: Розробка програмного забезпечення телеграм-боту кіноафіши м. Миколаєва

Виконав: студент групи 4151

_____Кузнецов О.С.

(підпис, ПІБ)

Керівник роботи:

доцент каф. ПЗАС, к.т.н., доцент

(посада, науковий ступень вчене звання)

_____Макарова Л.М.

(підпис, ПІБ)

Миколаїв – 2022 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра програмного забезпечення автоматизованих систем
Спеціальність 121 «Інженерія програмного забезпечення»
Освітня програма «Інженерія програмного забезпечення»

«ЗАТВЕРДЖУЮ»
Гарант освітньої програми
_____ доц. Макарова Л.М.
(підпис)
«___» _____ 2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ
на здобуття ступеня вищої освіти «бакалавр»

Студенту Кузнецов Олексій Сергійович _____
(Прізвище, ім'я, по батькові)

1. Тема роботи: Розробка програмного забезпечення телеграм-боту кіноафіши м. Миколаєва _____

Керівник роботи Макарова Лідія Миколаївна _____

Затверджені наказом ректора №256-уч від «11» травня 2022 р.

2. Термін подання роботи: 10.06.2022 р. _____

3. Вихідні дані по роботі: _____
_____

4. Перелік питань, що належать до розробки (найменування розділів)_____
Титульний аркуш; Завдання на кваліфікаційну роботу; Анотація (українською та англійською мовами); Зміст; Перелік умовних позначень, символів, одиниць та термінів (при необхідності); Вступ; Опис предметної галузі та постановка задачі; Проєкт програмного забезпечення (ескізний, технічний та робочий); Результати розробки програмного забезпечення; Розділ з охорони праці; Висновки; Список використаних джерел; Додатки (технічне завдання, текст програми, опис програми, інструкція користувача, програма та методика випробувань програмного забезпечення).

5. Перелік презентаційних матеріалів Титульний слайд, Мета кваліфікаційної роботи, Актуальність розробки, Аналіз існуючих аналогів, Функціональне призначення програмного забезпечення телеграм-боту кіноафіші м. Миколаєва, Діаграма варіантів використання, Діаграма діяльності для варіанту використання "Отримання інформації", Статична модель, представлена діаграмою класів, Інструменти розробки та технології, Результати розробки, Заключний слайд. _____
_____
_____
_____
_____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
4	Гурець Н.В., ст. викладач		

7. Дата видачі завдання 11.05.2022 р.

К А Л Е Н Д А Р Н И Й П Л А Н

Номер	Назва етапів роботи	Терміни виконання	Примітка
1.	Підготовка розділу Вступ	24.03.2022	ПП*
2.	Опис та аналіз предметної галузі	28.03.2022	ПП*
3.	Постановка задачі	31.03.2022	ПП*
4.	Ескізний проєкт програмного забезпечення	27.04.2022	
5.	Технічний проєкт програмного забезпечення	06.05.2022	
6.	Робочий проєкт програмного забезпечення	13.05.2022	
7.	Підготовка розділу Результати розробки програмного забезпечення	17.05.2022	
8.	Підготовка розділу з охорони праці	18.05.2022	ПП*
9.	Підготовка розділу Висновки	19.05.2022	
10.	Оформлення списку використаних джерел та додатків	20.05.2022	
11.	Подання на кафедру ПЗАС тексту остаточного варіанту роботи, підписаного її керівником, у роздрукованому та електронному форматі разом із заявами щодо самостійності виконання роботи та ідентичності друкованої та електронної версій роботи (Додатки 1 і 2 «Порядку здійснення заходів з перевірки робіт на наявність текстових збігів/ідентичності/схожості із використанням програмно-технічних засобів», який введений в дію наказом ректора НУК за №20 від 20.01.2020 р.)	10.06.2022	не пізніше, ніж за 14 діб до захисту (згідно п.4.1 зазначеного Порядку)
12.	Підготовка презентації та доповіді	12.06.2022	
13.	Попередній захист роботи на засіданні кафедри ПЗАС	14.06.2022	
14.	Подання на кафедру ПЗАС електронних версій наступних документів у форматі pdf: кваліфікаційної роботи; файлу-опису кваліфікаційної роботи (згідно Додатку до наказу ректора НУК за №287-уч від 19.05.2020 р.); презентації доповіді	24.06.2022	

* - за результатами переддипломної практики (ПП), яка була з 21.03.2022 до 08.05.2022 р.

Студент _____ Кузнецов О.С.
(підпис) (ПІБ)

Керівник роботи _____ Макарова Л.М.
(підпис) (ПІБ)

АНОТАЦІЯ

Кваліфікаційна робота присвячена розробці програмного забезпечення телеграм-боту кіноафіши м. Миколаєва, котре дозволить мешканцям міста зручніше отримувати повідомлення стосовно кінопоказів.

У кваліфікаційній роботі міститься аналіз та опис предметної галузі за темою кваліфікаційної роботи, постановка задачі на розробку програмного забезпечення, проект програмного забезпечення, результати розробки та розділ з охорони праці.

Дана кваліфікаційна робота передбачає розв’язання практичної задачі інженерії програмного забезпечення, що характеризується комплексністю та невизначеністю умов.

Кваліфікаційна робота викладена на 87 сторінках друкованого тексту та містить 29 рисунків, 2 таблиці, 5 додатків та список використаних джерел з 20 найменувань.

ABSTRACT

The qualification work is devoted to the development of the software of the telegram-bot of the movie poster of Mykolaiv, which will allow the residents of the city to more conveniently receive notifications about film screenings.

The qualification work contains an analysis and description of the subject area on the topic of qualification work, setting a task for software development, software design, development results and a section on labor protection.

This qualification work involves solving a practical problem of software engineering, which is characterized by complexity and uncertainty of conditions.

Qualification work is presented on the 87 pages of printed text and contains 29 figures, 2 tables, 5 appendices and a list of references from 20 names.

ЗМІСТ

ВСТУП

У даній кваліфікаційній роботі представлена інформація щодо розробки програмного забезпечення, яке призначено для інформування користувачів кіноафіши м. Миколаєва шляхом використання телеграм-боту.

Досить важко уявити сучасне життя людини без кіно, та світ без кінотеатрів. Майже кожен з нас хоча б раз відвідував кінотеатр свого рідного міста, та перед тим як купити квитки проводив декілька хвилин розглядаючи афішу показів. З розвитком технологій та інтернету, багато кінотеатрів задля покращення інформування своїх клієнтів почали обзаводитися веб-сайтами, де тепер можна детально та у зручному форматі ознайомитися з «асортиментом» і навіть заздалегідь замовити квитки. Із поширенням кількості кінотеатрів, зростала і кількість веб-сайтів до них, тож якщо людина хоче підшукати найзручніший час показу або порівняти ціну квитків, потрібно кожного разу відкривати нову сторінку браузеру, на фоні цього виникає потреба створення більш зручного інтерфейсу для перегляду інформації яку надають сайти кінотеатрів та її аналізу.

Метою роботи є розробка програмного забезпечення, що буде надавати інформацію з різних сайтів кінотеатрів окремого міста у більш зручному форматі для користувача. Можливість такого може надати чат-бот популярного месенджеру Telegram який має більш простий та зрозумілий інтерфейс для виведення оброблених даних.

Дана кваліфікаційна робота передбачає розв’язання практичної задачі інженерії програмного забезпечення, що характеризується комплексністю та невизначеністю умов. Виділимо цілі які необхідні для розробки ПЗ:

- Аналіз предметної галузі;
- Аналіз існуючих аналогів;
- Вибір середовища розробки ПЗ та необхідних технологій;
- Розробка чат-боту на платформі Telegram.

1 АНАЛІЗ РОБОТИ ІНФОРМАЦІЙНОЇ СИСТЕМИ КІНОТЕАТРІВ ТА ПОСТАНОВКА ЗАДАЧІ НА РОЗРОБКУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТЕЛЕГРАМ-БОТУ КІНОАФІШИ МІСТА МИКОЛАЄВА

1.1 Опис та аналіз роботи інформаційної системи кінотеатрів

Кінотеатр – це приміщення з обладнанням яке призначене для публічної демонстрації кінофільмів. Робота кожного кінотеатру побудована таким чином, що покази йдуть сім днів на тиждень майже цілодобово (ранкові, денні, вечірні, та нічні сеанси). Кожний кінотеатр має розклад сеансів фільмів та інформацію про них, а саме назва фільму, жанр, вікові обмеження, дата та час показу.

Більш детальна інформація стосовно фільмів зазвичай вказана на веб-сайті кінотеатру. Там вже можуть міститись рік виходу фільму так як іноді відбуваються покази старих фільмів або ж тематичні кіно вечори. Також вказують ім’я режисера, назву країни та студії виробника, авторів сценарію, мову перекладу у випадку іноземних картин. Також на сайтах зазвичай міститься стислий опис фільму для того щоб людині було зрозуміло про що буде йти розповідь у цій стрічці.

«Multiplex» – найбільша мережа кінотеатрів в Україні яка була заснована ще у 2004 році. Вона включає до себе 28 кінотеатрів та 141 кінозал у найбільших містах країни [1].

«Планета Кіно» – з’явилась у 2008 році та відома як перша компанія яка завезла в Україну канадську технологію IMAX (Екран розміром з п’ятиповерховий будинок, два потужні проектори, гучний звук та чітке зображення) [2].

Веб-сайти обох мереж досить схожі за функціоналом. Титульні сторінки обох сайтів представлені на рисунках 1.1 та 1.2.

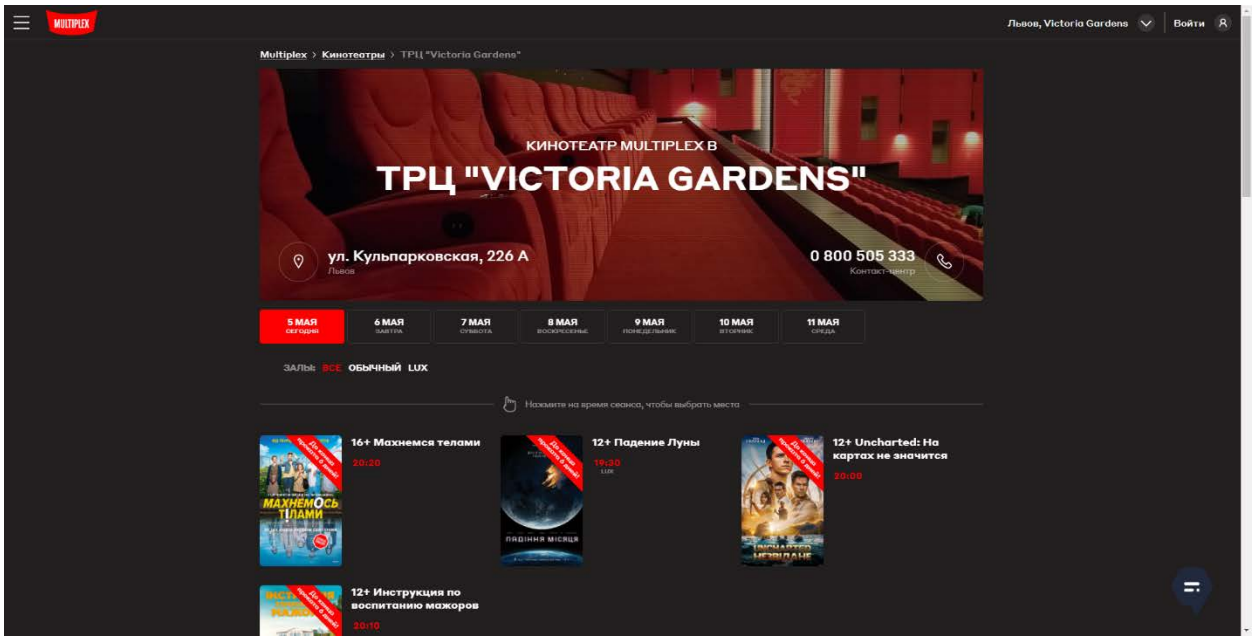

Рисунок 1.1 – Титульна сторінка сайту multiplex.ua [1]

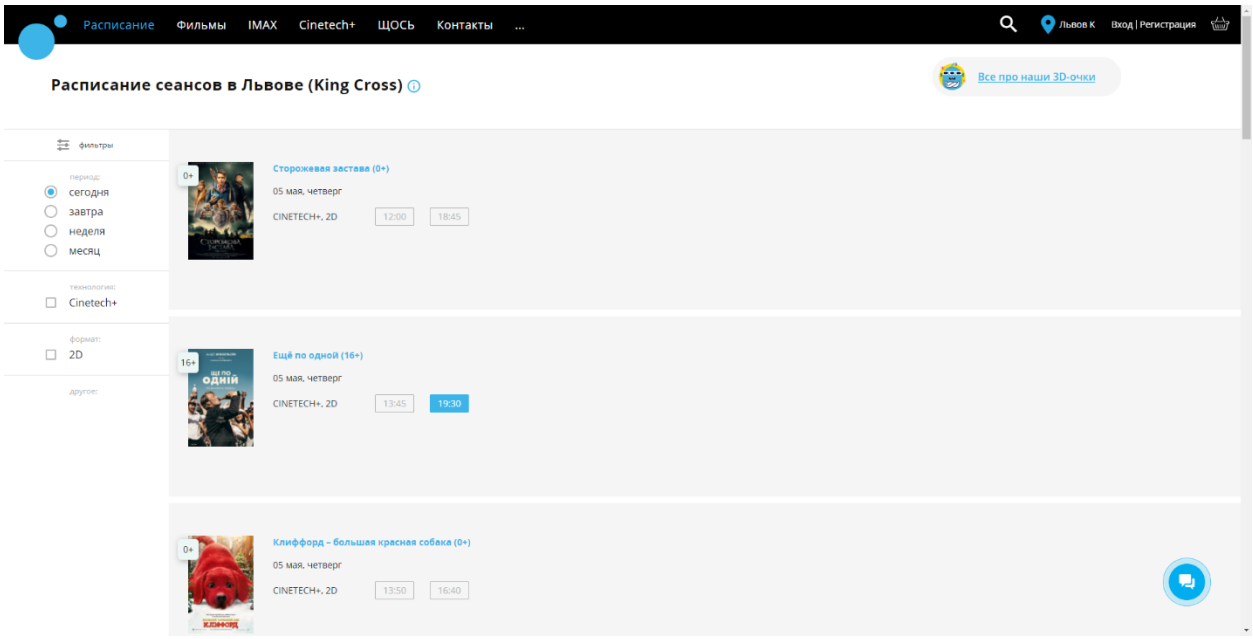

Рисунок 1.2 – Титульна сторінка сайту planetakino.ua [2]

Кожен фільм має свою власну сторінку з більш детальною інформацію, описом та можливістю замовити квитки, приклад сторінки наведено на рисунку 1.3.

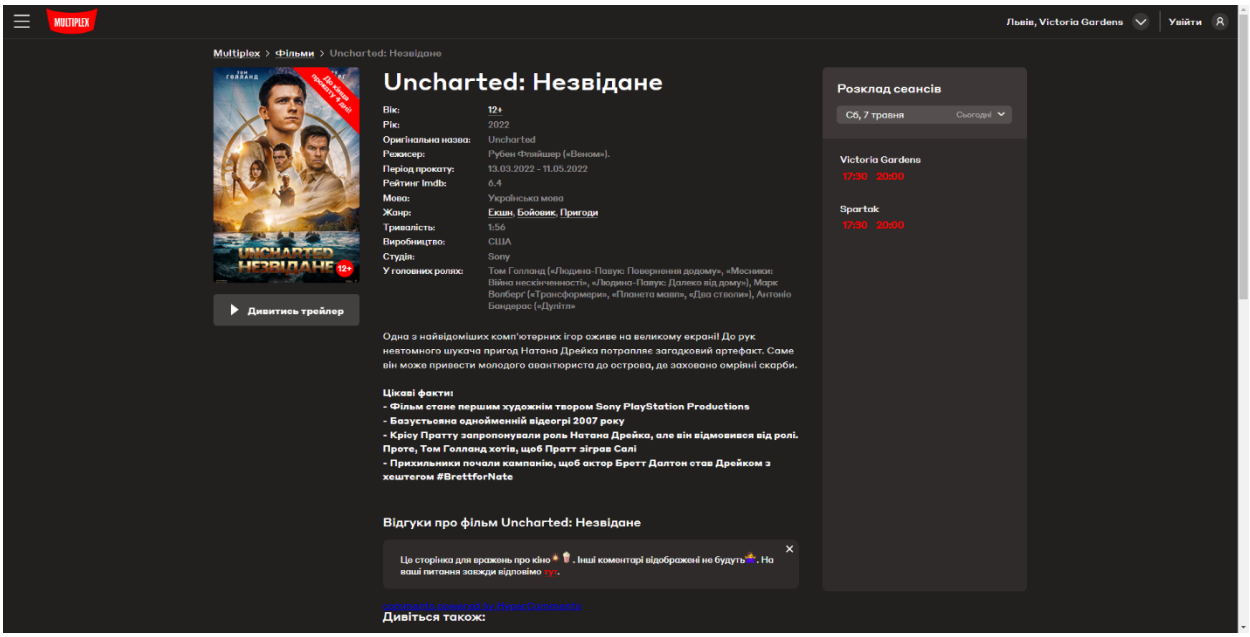

Рисунок 1.3 – Сторінка фільму сайту multiplex.ua [1]

Основною та необхідною інформацією для користувача є інформація стосовно кінотеатру, фільму та розкладу сеансів.

1.2 Боти і месенджери у сучасному житті

На сьогоднішній день велика кількість людей має смартфон, який наділений великою кількістю функціональних можливостей, але однією з найважливіших завжди був і залишається – зв’язок, можливість для людей комунікувати один з одним. Таку можливість надають месенджери – програмне забезпечення, яке дозволяє обмінюватись повідомленнями у текстовому, голосовому, а у деяких випадках і відео форматах.

До популярних месенджерів на момент написання цієї роботи відносяться:

Facebook Messenger – це сервіс для обміну миттєвими повідомленнями, посиланнями, відео, знімками та іншою інформацією [3]. Безпосередньо пов’язаний із повідомленнями на сайті Facebook. Цей месенджер схожий на багато інших, але зі своїми особливостями. Він вміє надсилати повідомлення з телефонної книги, на конкретний номер, створювати групові чати, прикріплювати

до повідомлень медіа-файли, підтримує дзвінки. При цьому листування можна урізноманітнити наклейками та голосовими повідомленнями.

Discord – це додаток створений для повідомлень, аудіодзвінків та різного роду відеозв'язку [3]. Спочатку був створений для гравців у відеоігри, надавав їм можливість знаходити один одного, координувати гру та розмовляти під час гри. Однак зараз сервіс зручно використовувати не лише для спілкування під час ігор. Програма робить спілкування в чаті досить простим та пропонує функції пошуку, які допоможуть знайти інших людей та додати їх до списку друзів для швидкого спілкування. Користувачі можуть запрошувати інших приєднатися до Discord через веб-посилання, що означає, що для використання сервісу не потрібно створювати обліковий запис або встановлювати програму. Сервіс підтримує відеодзвінки з кількістю учасників до 50, голосовий чат, дозволяє надсилати миттєві повідомлення, створювати групові чати, спільно використовувати екрани.

Viber – це сервіс обміну повідомленнями та здійснення дзвінків, представлений у вигляді додатку [3]. Він дозволяє безкоштовно обмінюватися повідомленнями, телефонувати колегам та друзям на будь-які пристрої. При цьому Viber вміє синхронізувати базу контактів, повідомлень та історію дзвінків між різними гаджетами.

Telegram – це веб-додаток для обміну миттєвими повідомленнями з акцентом на швидкість та безпеку [3]. Це швидкий, простий, безпечний та безкоштовний сервіс. Він легко синхронізується на всіх пристроях, працює на настільних ПК, планшетах та телефонах. За допомогою нього можна надсилати необмежену кількість повідомлень, фотографій, відео та файлів будь-якого типу. Сервіс використовує розподілену інфраструктуру з центрами обробки даних у всьому світі для прискорення доступу до серверів та роботи програм. При цьому Telegram заснований на протоколі MTProto, який побудований на перевірених часом алгоритмах, забезпечуючи високу швидкість доставки та надійність. Telegram пропонує безпечний доступ до даних разом із збереженням історії листування у хмарі.

Серед великої кількості переваг месенджеру Telegram можна додати ще одну – боти. Телеграм-бот – це спеціальний обліковий запис, створений в автоматичному режимі, який дозволяє користувачам здійснювати різні дії через сам месенджер. Боти виконують дії за текстовими командами користувача, за принципом «питання-відповідь» після натискання кнопки «Старт». Бота визначити легко, в його назві завжди є слово «bot», він не може розпочати бесіду і не має статусу «онлайн» або «не в мережі», користувач завжди буде бачити підпис «bot».

Сьогодні боти стали дуже популярними, вони допомагають користувачам виконувати типові рутинні дії в автоматичному режимі, значно спрощуючи їм життя. Для власників телеграм-каналів, боти стали незамінними помічниками в роботі.

Телеграм-боти мають велику кількість плюсів:

– Моментальна відповідь користувачеві;

– Зручність у користуванні, спілкування за принципом «питання-відповідь», текстові завдання під силу давати навіть зовсім недосвідченому користувачеві месенджера;

– Не вимагають встановлення додаткових програм, додатків, тощо. Все спілкування з ботом ведеться безпосередньо через месенджер;

– Безпека особистих даних, боти працюють виключно за заданими командами.

– Необмежені можливості, віртуального помічника можна запрограмувати на відправку новин, розповідь анекдотів, нагадування важливої інформації, пошук закладів, бронювання столиків у ресторані, замовлення квитків тощо.

Функції телеграм-боту:

Розваги – боти можуть надсилати смішні картинки, анекдоти, допомагають обрати фільм, знайти пісню за голосовим повідомленням і т.д.

Пошук та обмін файлів – бот допомагає відправляти та зберігати файли з різних джерел, знаходити електронні книжки та багато іншого.

Новини, важлива інформація – бот надасть новини, погоду, курси валют тощо.

Утиліти та інструменти – бот допомагає перекладати тексти, нагадувати про важливі події тощо.

Інтеграція з іншими сервісами – бот може надсилати повідомлення, управляти системами «розумного будинку» і т.д.

Пошук місць – бот допомагає шукати готелі, кінотеатри, ресторани та інші заклади.

Транзакції – бот дозволяє бронювати квитки, робити замовлення, викликати таксі та ін.

1.3 Аналіз існуючого програмного забезпечення

На сьогоднішній день існує досить багато веб-сайтів за темою «афіша кіно». Тож переглянемо та проаналізуємо декілька існуючих аналогів програмного забезпечення та розглянемо їх переваги і недоліки.

«i.ua» – Інтернет-портал, що з’явився у 2008 році, це один з найбільших безкоштовних сервісів електронної пошти україномовного сектору мережі Інтернет [4]. Як кажуть автори, за результатами жовтня 2011 року входив до десятки найбільш відвідуваних ресурсів України. На сторінці цього сайту є розділ «Кінотеатри» на якому надається інформація стосовно показів на тиждень. Сторінка розділу представлена на рисунку 1.4.

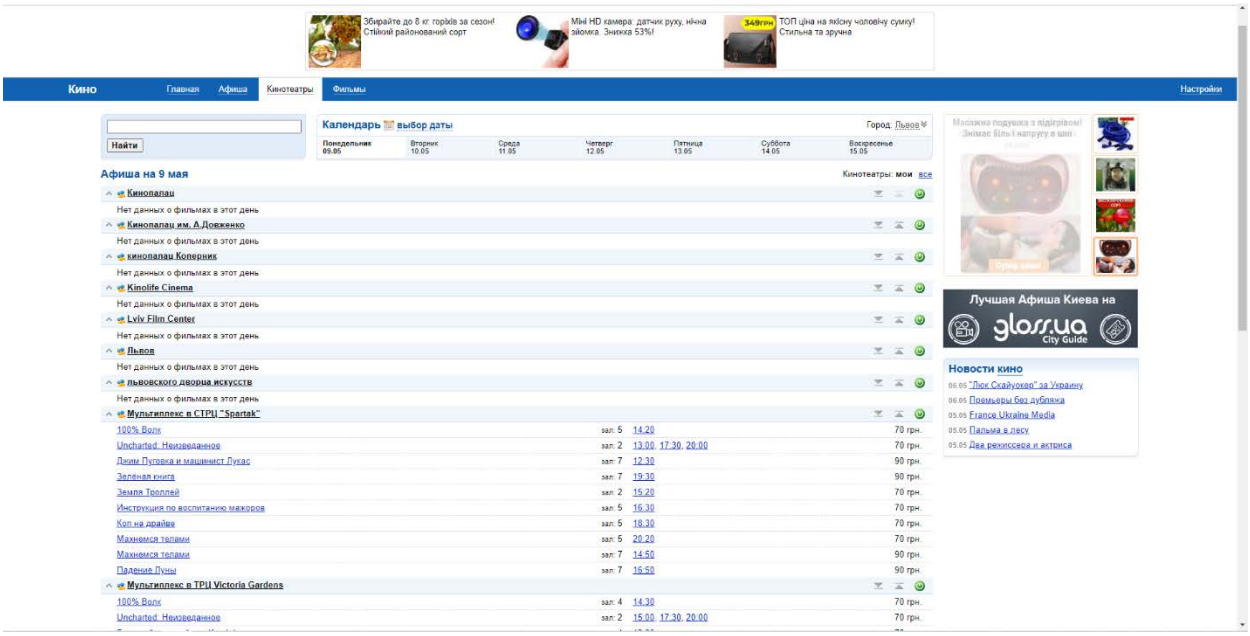

Рисунок 1.4 – Сторінка розділу «Кінотеатри» сайту «i.ua» [4]

«kinofilms.ua» – сайт, присвячений світу кіно та зроблений для тих, хто полюбляє фільми [5]. На порталі розміщується ознайомча інформація про фільми та знаменитості, надана прокатниками фільмів у рекламних цілях. Містить розділ з кіноафішою, приклад показано на рисунку 1.5.

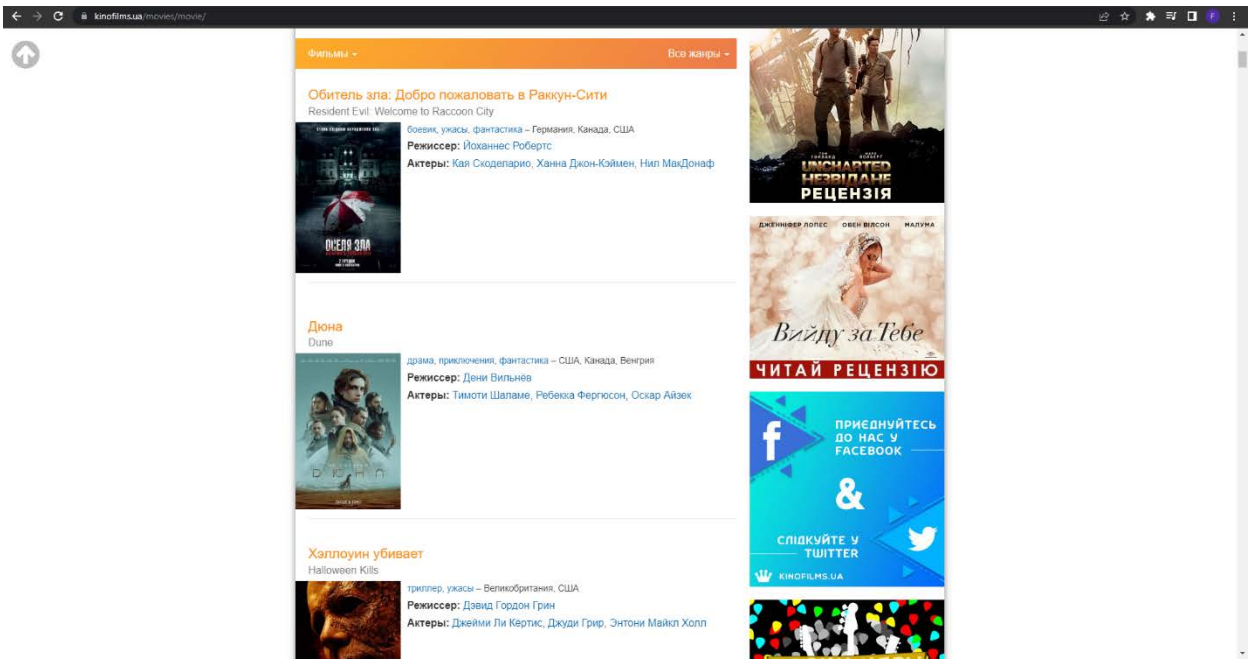

Рисунок 1.5 – Сторінка розділу «Фільми» сайту «kinofilms.ua» [5]

«kinoafisha.ua» – На сайті глядача чекають новинки зі світу кіно, цікавий і актуальний погляд на фільми в підбірках, рецензіях, оглядах і новинах [6]. На сайті представлені трейлери і анонси самих очікуваних фільмів, фотографії, постери і кадри, інформація про рейтинги і касові збори. Можна знайти авторські рецензії на кіно новинки та кращі фільми останніх десятиліть, а також написати свою рецензію або поділитися враженнями про фільм в короткому відгуку. Сайт надає можливість ознайомитися з розкладами сеансів в кінотеатрах України. Приклад сторінки розкладу сеансів показано на рисунку 1.6.

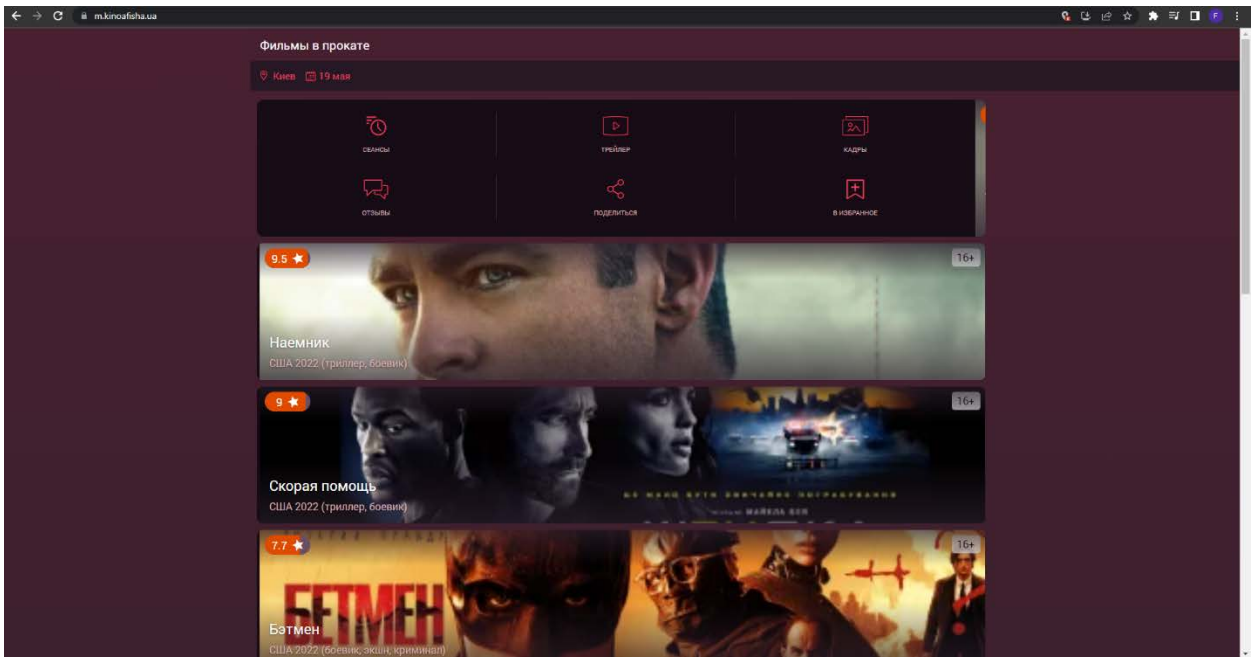

Рисунок 1.6 – Сторінка розділу «Фільми у прокаті» сайту «kinoafisha.ua» [6]

Проаналізувавши аналогічні існуючі програмні рішення з надання інформації стосовно кіноафіши міст, можна виділити наступні недоліки вже існуючих рішень:

– Існуюче програмне забезпечення має велику кількість рекламних банерів ,що заважають перегляду контенту або відволікають від нього.

– Деякі сервіси не призначені саме для вирішення поставленої задачі.

– Сервіси не надають зручного обсягу інформації, яка необхідна користувачеві, десь забагато зайвої, десь замало необхідної.

– Вже існуюче ПЗ має непотрібні в контексті вирішуваної задачі функції.

– Декотрі аналоги мають застарілий або не зручний інтерфейс.

1.4 Постановка задачі для розробки програмного забезпечення телеграм-боту кіноафіши м. Миколаєва

Враховуючи описану та проаналізовану інформацію стосовно предметної галузі телеграм-боту кіноафіши м. Миколаєва в підрозділі 1.1 та аналіз існуючого програмного забезпечення в підрозділі 1.4, дійшли до висновків стосовно необхідності розробки телеграм-боту кіноафіши м. Миколаєва.

Програмне забезпечення розроблюється виключно для користувача – людини, яка бажає дізнатися інформацію стосовно кінопоказів міста Миколаєва. Користувач взаємодіє з інтерфейсом телеграм-боту за допомогою команд або спеціальної клавіатури, бот в залежності від функції яку обрав користувач видає на екран оброблену інформацію.

Тож, розпишемо більш детально функції які доступні користувачеві. На вибір є чотири основні функції, перша – це «старт», яка запускає роботу самого телеграм-боту та виводить на екран спеціальну клавіатуру для спрощеної взаємодії, друга – це «Зараз у кіно», з подальшим вибором дня тижня який цікавить користувача, вказавши необхідний день, бот виводить інформацію стосовно сеансів на обрану дату, третя – це «Скоро у кіно», яка виводить на екран інформацію щодо майбутніх показів які заплановані у кінотеатрах міста Миколаєва, четверта – це «Кінотеатри», яка надає користувачеві список кінотеатрів міста та їх контактні дані для замовлення квитків, або для того щоб завдати питання по телефону гарячої лінії.

При роботі з ботом, користувач отримуватиме структуровану інформацію стосовно фільмів та кінотеатрів, визначимо атрибути, які будуть їх характеризувати:

1.Атрибути, що будуть характеризувати фільми:

- назва;
- країна;
- рік випуску;
- жанр;
- режисер;
- актори;
- дата прем’єри;
- рейтинг IMDB;
- вікове обмеження;
- назва кінотеатру у якому наявні сеанси;
- номер зали;
- час сеансу.

2.Атрибути, що будуть характеризувати кінотеатри:

- назва кінотеатру;
- адреса;
- телефон гарячої лінії;
- адреса сайту кінотеатру.

Докладніша інформація стосовно вимог до програмного забезпечення наведена в технічному завданні у Додатку А.

2 ПРОЕКТ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТЕЛЕГРАМ-БОТУ КІНОАФІШИ МІСТА МИКОЛАЄВА

2.1 Ескізний проект програмного забезпечення

За результатами проведеного аналізу вимог до програмного забезпечення був розроблений ескізний проект за допомогою мови моделювання UML.

Уніфікована мова моделювання (UML) – це стандартизована мова моделювання, яка дозволяє розробникам визначати, візуалізувати, конструювати та документувати артефакти програмної системи [7].

До програмного забезпечення була розроблена діаграма варіантів використання та виконано структуризацію і специфікацію варіантів використання.

З метою формалізації представлення сценаріїв варіантів використання, для кожного з них було побудовано діаграми діяльності та розроблена інформаційна модель за допомогою діаграми класів.

2.1.1 Побудова моделі варіантів використання

Діаграма варіантів використання описує який функціонал програмної системи, що розробляється, доступний кожній групі користувачів [8].

Після аналізу функціональних вимог до програмного забезпечення, були виділені основні варіанти використання:

- переглянути сеанси на сьогодні;
- переглянути сеанси на завтра;
- переглянути сеанси на післязавтра;
- переглянути сеанси на подальший день;
- отримати інформацію стосовно майбутніх показів;
- отримати інформацію стосовно кінотеатрів.

На рисунку 2.1 зображено діаграму варіантів використання.

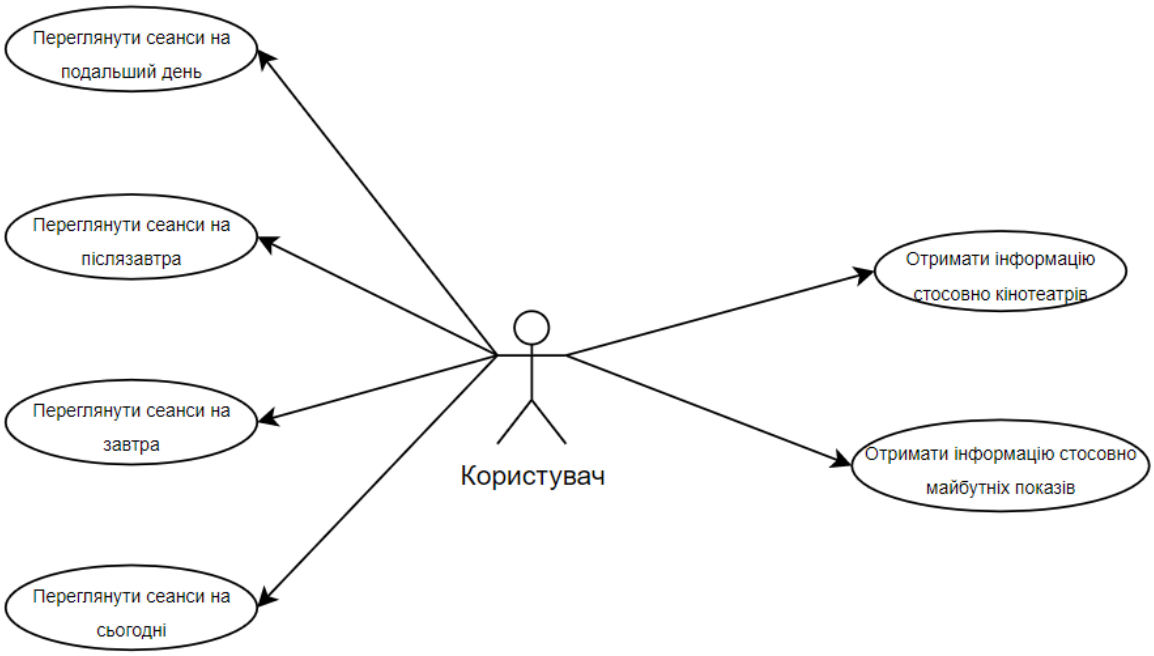

Рисунок 2.1 – Діаграма варіантів використання

Таким чином, була розроблена діаграма варіантів використання для подальшого проектування програмного забезпечення.

2.1.2 Специфікації варіантів використання

Проаналізувавши діаграму варіантів використання, проведемо специфікації для кожного з варіантів використання. Нижче представлений реєстр варіантів використання, який зображено на таблиці 1.

Таблиця 2.1 – Реєстр варіантів використання

Код	Основна дійова особа (актор)	Назва варіанту використання	Стислий опис
1	2	3	4
A1	Користувач	Переглянути сеанси на сьогодні	Прецедент дозволяє переглянути список фільмів, що йдуть у місті за сьогоднішньою датою

Продовження таблиці 2.1

1	2	3	4
A2	Користувач	Переглянути сеанси на завтра	Прецедент дозволяє переглянути список фільмів, що йдуть у місті завтрашньою датою
A3	Користувач	Переглянути сеанси на післязавтра	Прецедент дозволяє переглянути список фільмів, що йдуть у місті післязавтра
A4	Користувач	Переглянути сеанси на подальший день	Прецедент дозволяє переглянути список фільмів, що йдуть у місті на подальші дні тижня
A5	Користувач	Отримати інформацію стосовно майбутніх показів	Прецедент дозволяє переглянути список майбутніх кінопрем’єр у кінотеатрах міста
A6	Користувач	Отримати інформацію стосовно кінотеатрів	Прецедент дозволяє отримати контактну інформацію стосовно кінотеатрів міста

Конкретизуємо варіанти використання.

A1. Переглянути сеанси на сьогодні

Короткий опис: Прецедент дозволяє переглянути список фільмів, що йдуть у місті сьогоднішньою датою.

Основна дійова особа: Користувач.

Інші учасники прецеденту: Відсутні.

Передумови: Користувач ініціював перегляд сеансів на сьогодні.

Основний потік подій:

1. Користувач запускає бота;

2. Бот виводить вітальне повідомлення та клавіатуру;

3. Користувач натискає кнопку «Сеанси на сьогодні».

Альтернативний потік подій: Відсутній.

Постумови: Бот виводить інформацію стосовно сеансів на сьогодні.

А2 Переглянути сеанси на завтра

Короткий опис: Прецедент дозволяє переглянути список фільмів, що йдуть у місті завтра.

Основна дійова особа: Користувач.

Інші учасники прецеденту: Відсутні.

Передумови: Користувач ініціював перегляд сеансів на завтра.

Основний потік подій:

1. Користувач запускає бота;

2. Бот виводить вітальне повідомлення та клавіатуру;

3. Користувач натискає кнопку «Сеанси на завтра».

Альтернативний потік подій: Відсутній

Постумови: Бот виводить інформацію стосовно сеансів на завтра.

А3 Переглянути сеанси на післязавтра

Короткий опис: Прецедент дозволяє переглянути список фільмів, що йдуть у місті післязавтра.

Основна дійова особа: Користувач.

Інші учасники прецеденту: Відсутні.

Передумови: Користувач ініціював перегляд сеансів на післязавтра.

Основний потік подій:

1. Користувач запускає бота;

2. Бот виводить вітальне повідомлення та клавіатуру;

3. Користувач натискає кнопку «Сеанси на післязавтра».

Альтернативний потік подій: Відсутній.

Постумови: Бот виводить інформацію стосовно сеансів на післязавтра.

А4 Переглянути сеанси на подальший день

Короткий опис: Прецедент дозволяє переглянути список фільмів, що йдуть у місті на четвертий день.

Основна дійова особа: Користувач.

Інші учасники прецеденту: Відсутні.

Передумови: Користувач ініціював перегляд сеансів на 4-й день.

Основний потік подій:

1. Користувач запускає бота;

2. Бот виводить вітальне повідомлення;

3. Користувач натискає кнопку з подальшим днем тижня.

Альтернативний потік подій: Відсутній.

Постумови: Бот виводить інформацію стосовно показів на подальший день тижня.

А5 Переглянути інформацію стосовно майбутніх показів

Короткий опис: Прецедент дозволяє переглянути список майбутніх кінопрем’єр у кінотеатрах міста.

Основна дійова особа: Користувач.

Інші учасники прецеденту: Відсутні.

Передумови: Користувач ініціював перегляд інформації майбутніх показів.

Основний потік подій:

1. Користувач запускає бота;

2. Бот виводить вітальне повідомлення та клавіатуру;

3. Користувач натискає кнопку «Скоро у кіно».

Альтернативний потік подій: Відсутній.

Постумови: Бот виводить інформацію стосовно майбутніх прем’єр у кінотеатрах міста.

А6 Отримати інформацію стосовно кінотеатрів

Короткий опис: Прецедент дозволяє отримати контактну інформацію стосовно кінотеатрів міста.

Основна дійова особа: Користувач.

Інші учасники прецеденту: Відсутні.

Передумови: Користувач ініціював перегляд інформації стосовно кінотеатрів.

Основний потік подій:

1. Користувач запускає бота;
2. Бот виводить вітальне повідомлення та клавіатуру;
3. Користувач натискає кнопку «Інформація про кінотеатри».

Альтернативний потік подій: Відсутній.

Постумови: Бот виводить інформацію стосовно кінотеатрів міста.

2.1.3 Сценарії варіантів використання

На даній стадії процесу розробки програмного забезпечення створюється діаграма діяльності.

Діаграма діяльності відображає динамічні аспекти поведінки системи, є блок-схемою, яка наочно показує, як потік управління переходить від однієї діяльності до іншої [9].

Доцільне створення діаграми діяльності для основного варіанту використання, а саме – отримання інформації користувачем, який наведено на рисунку 2.2.

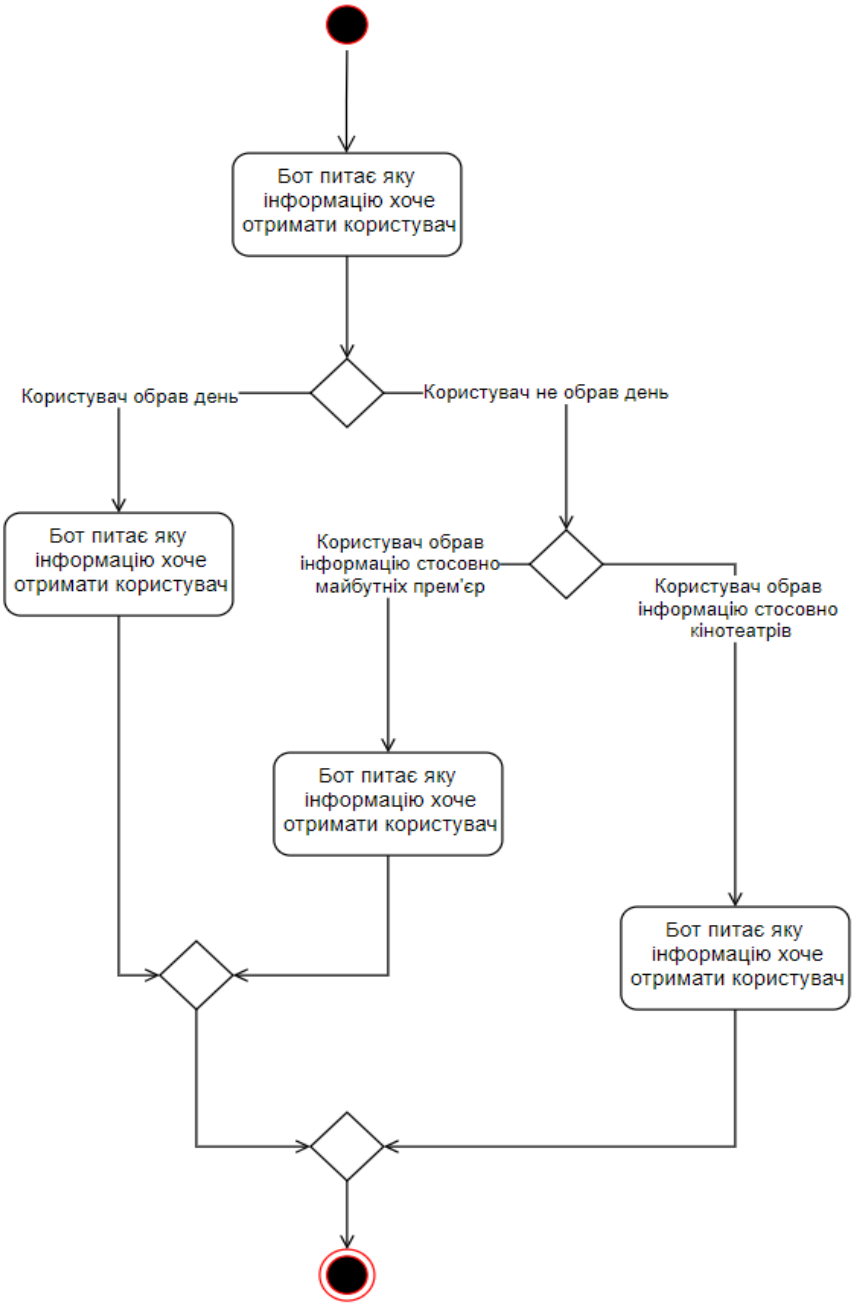

Рисунок 2.2 – Діаграма діяльності для варіанту використання «Отримання інформації

Таким чином було розроблено діаграму діяльності для візуалізації основних варіантів використання програмного забезпечення.

2.1.4 Інформаційна модель

Діаграма відносин сутностей (ERD) – це візуальне уявлення різних сутностей усередині системи та їх взаємозв’язку один з одним [10].

Інформаційна модель програмного забезпечення представлена ER діаграмою, яку наведено на рисунку 2.3.

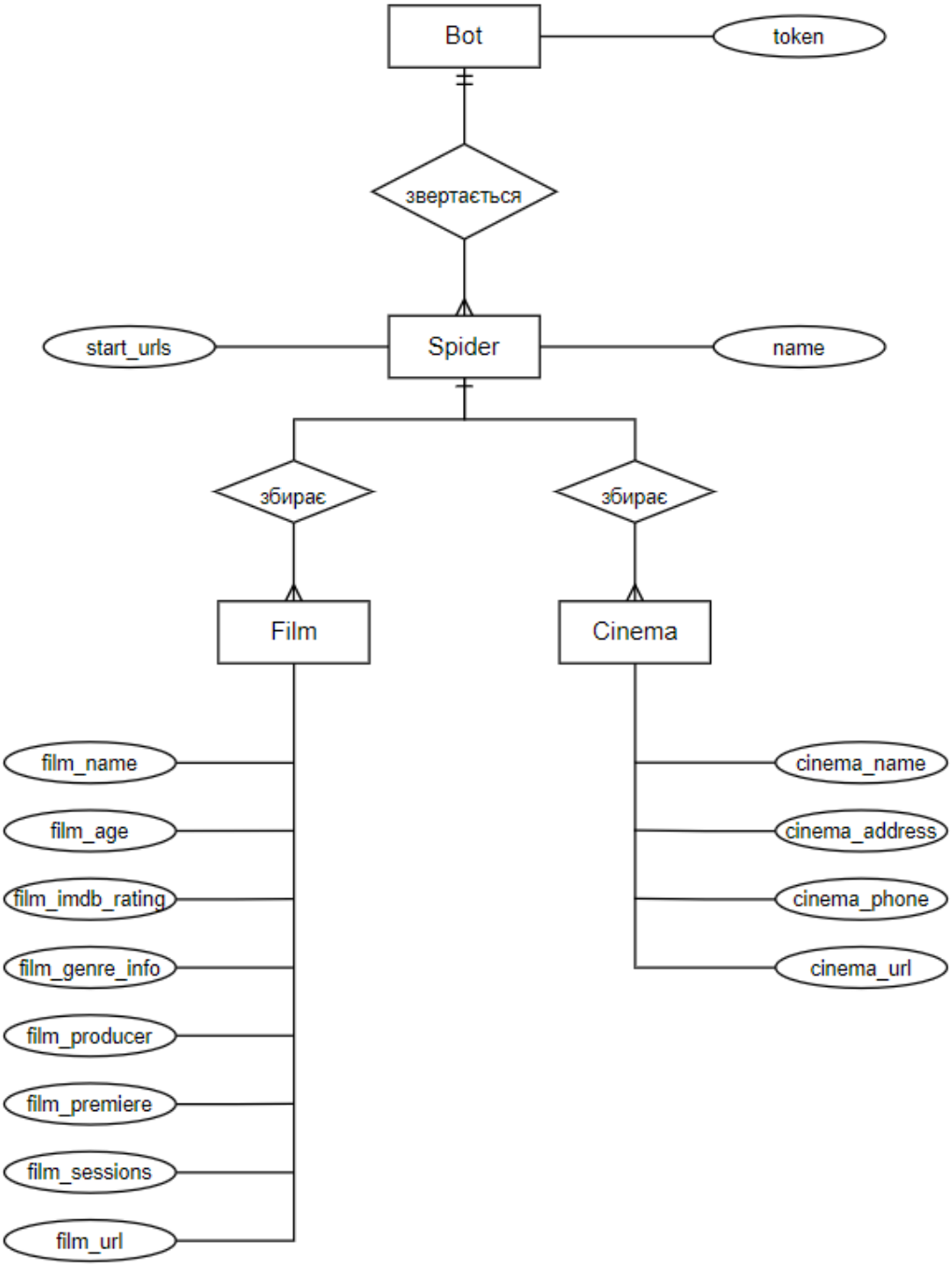

Рисунок 2.3 – ER діаграма

Таким чином було створено ER діаграму до розроблюваного ПЗ.

2.1.5 Розробка інтерфейсу програмного забезпечення

Інтерфейс – це механізм чи інструмент, що представляє з себе набір об’єктів, піктограм та графічних елементів, які функціонують як метафори чи символи для дій чи завдань, які користувач може виконувати на комп’ютері чи смартфоні [11].

У даному розділі кваліфікаційної роботи представлені схеми взаємодії користувача з ботом. При початку чату з ботом, він відправляє вітальне повідомлення, на якому вказані інструкції взаємодії з ним та надає пункти за якими користувач може отримати інформацію. Вітальне повідомлення зображено на рисунку 2.4.

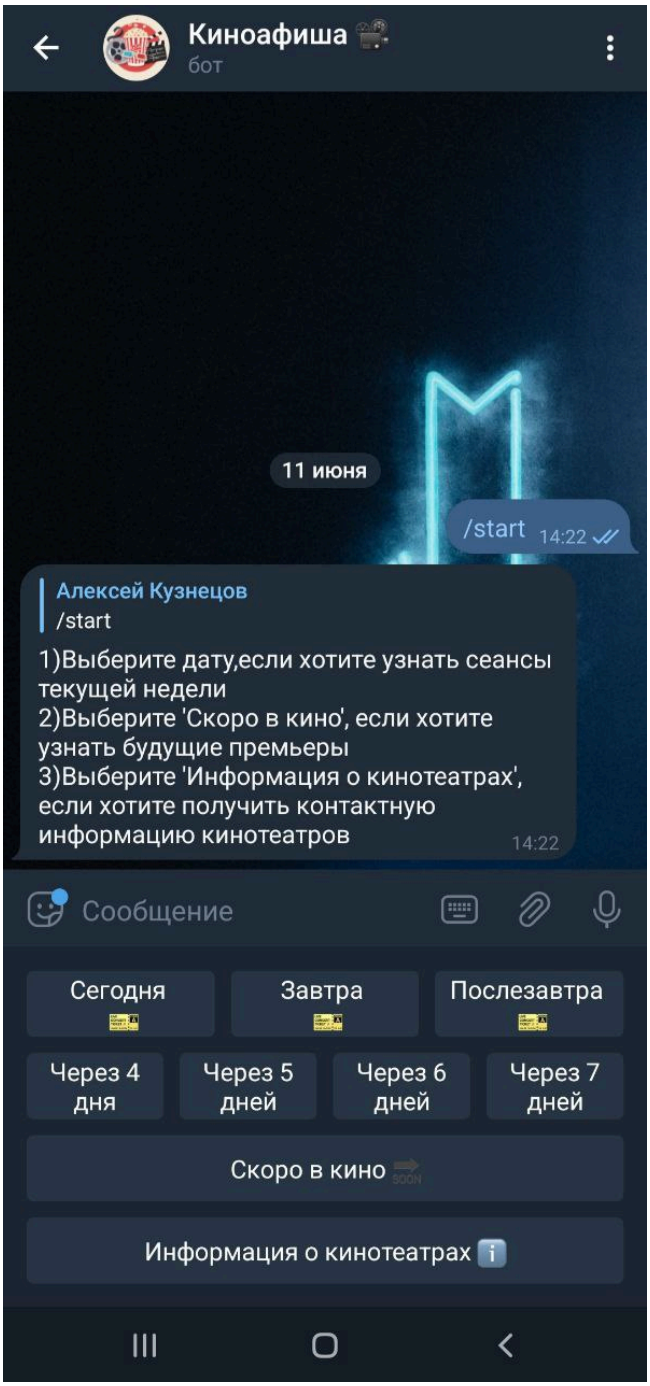

Рисунок 2.4 – Вітальне повідомлення

Надання інформації стосовно показів сьогодні зображено на рисунку 2.5.

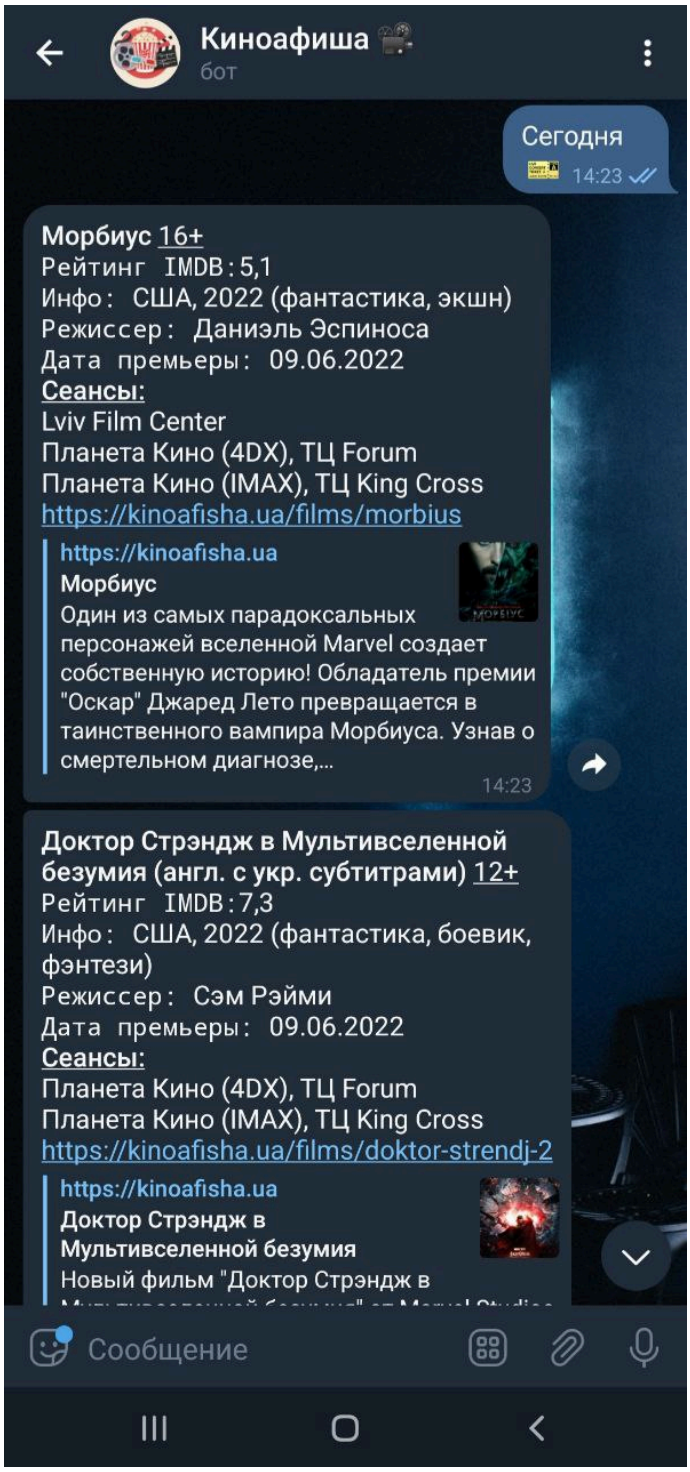

Рисунок 2.5 – Надання інформації стосовно показів

Надання інформації стосовно майбутніх прем’єр зображено на рисунку 2.6.

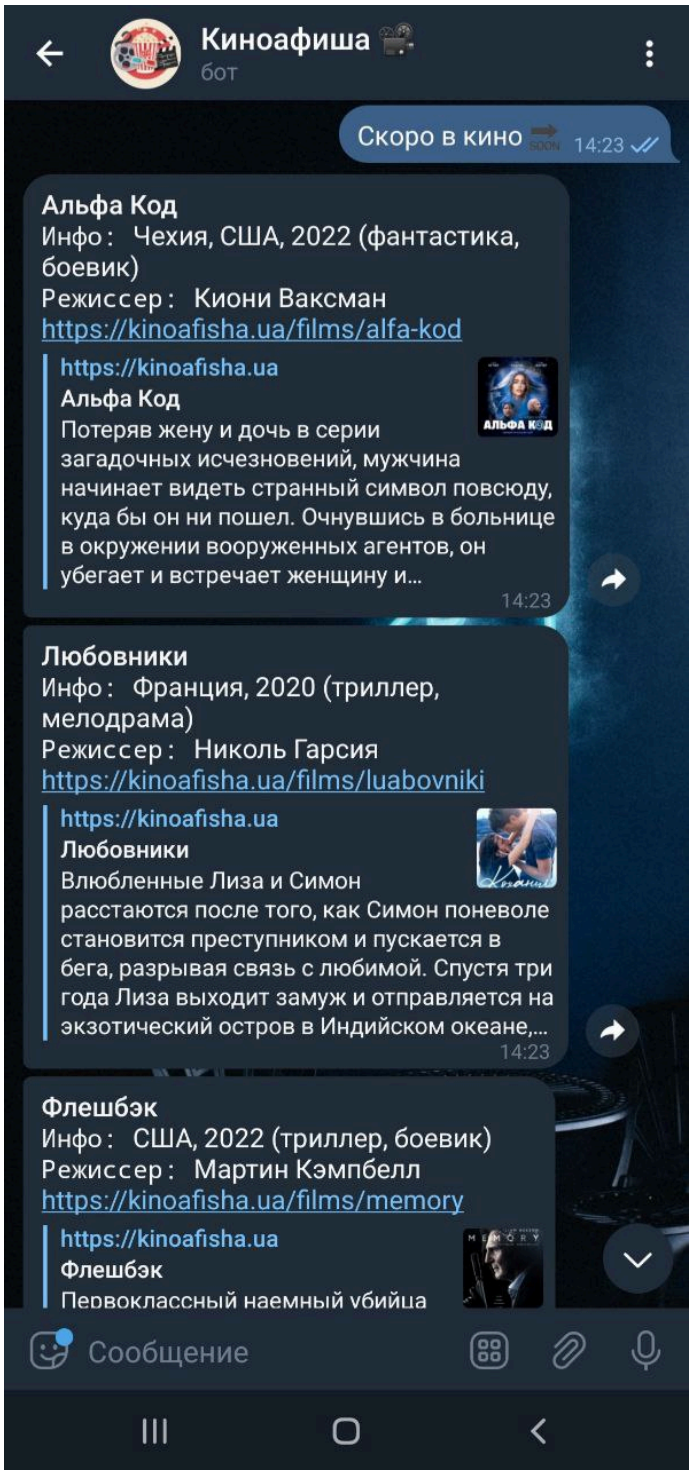

Рисунок 2.6 – Надання інформації стосовно майбутніх прем’єр

Надання інформації стосовно кінотеатрів міста зображено на рисунку 2.7.

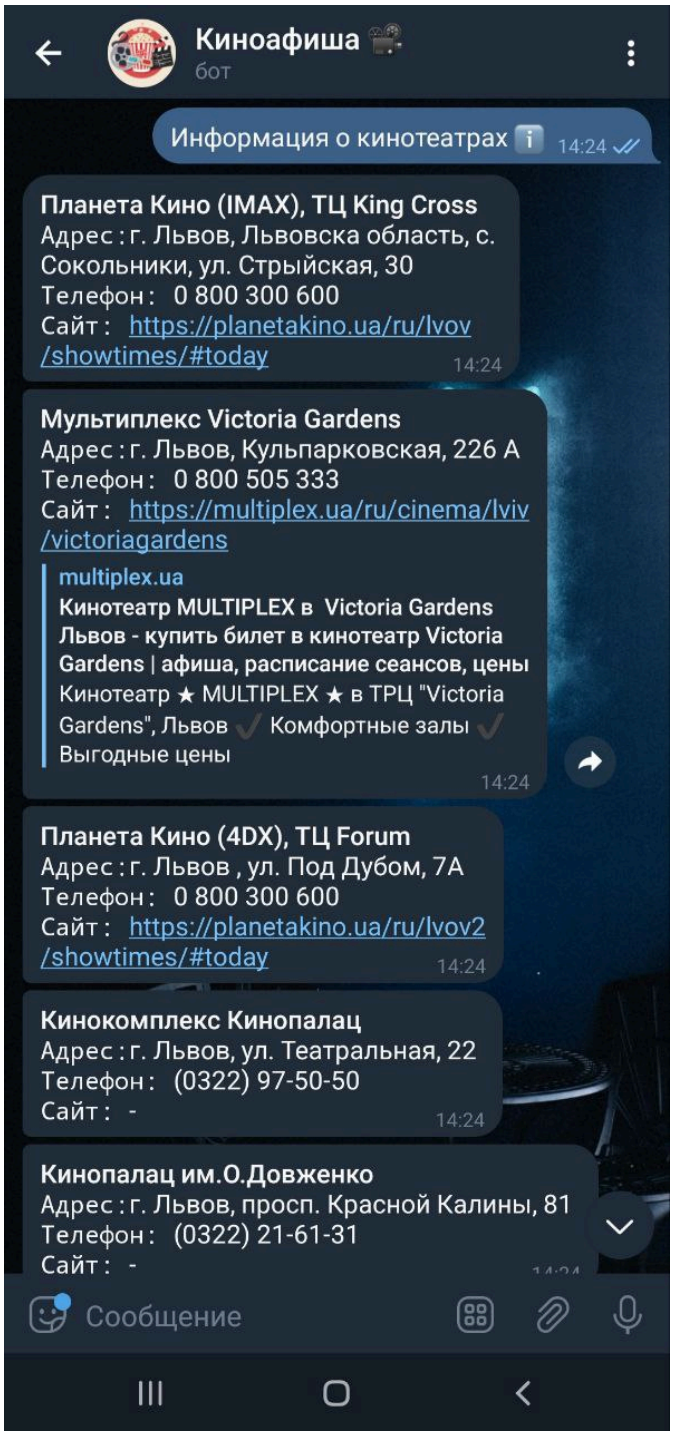

Рисунок 2.7 – Надання інформації стосовно кінотеатрів міста

Таким чином було розроблено інтерфейс до розроблюваного програмного забезпечення.

2.2 Технічний проект програмного забезпечення

Після проведення аналізу ескізного проекту, був розроблений технічний проект програмного забезпечення шляхом побудови статичної моделі, яка представлена діаграмою класів, та динамічної, що представлена діаграмою послідовності.

2.2.1 Діаграма класів

Діаграма класів – це набір статичних, декларативних елементів моделі, інформація з діаграми класів відображається у вихідний код програми [12]. Таким чином, діаграма класів кінцевий результат проектування та відправна точка процесу розробки.

Статична модель, зображена на рисунку 2.8, представлена діаграмою класів.

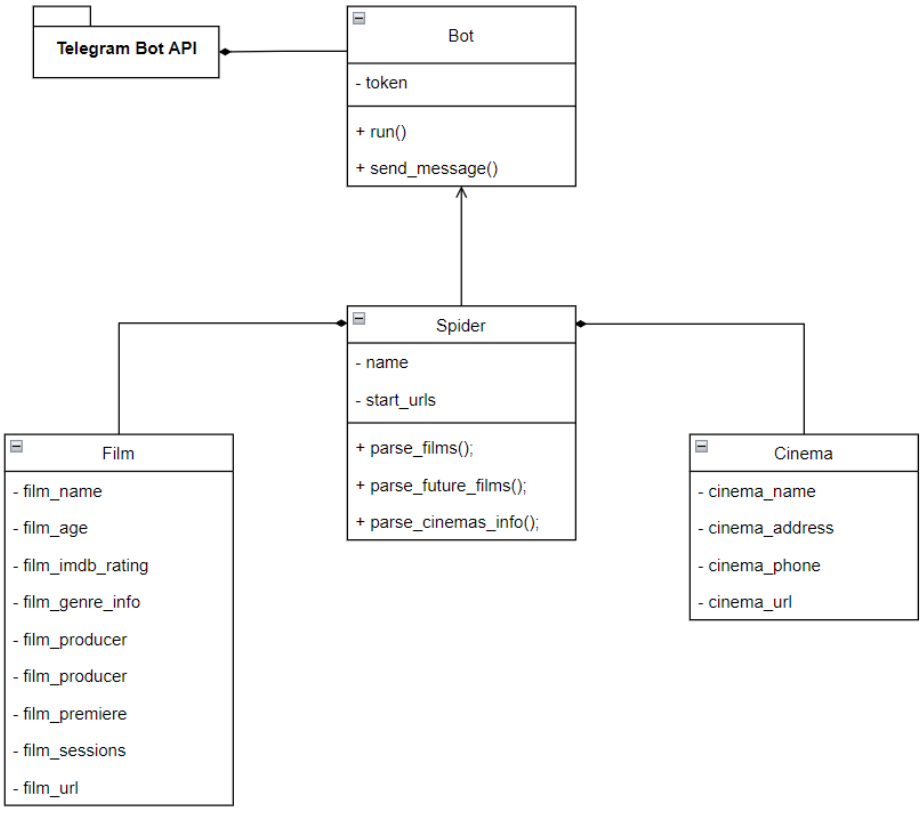

Рисунок 2.8 – Статична модель, представлена діаграмою класів

Таким чином було побудовано діаграму класів, яка розроблена на основі моделі варіантів використання та моделі програмного інтерфейсу.

2.2.2 Специфікації класів

Клас: «Bot»

Відповідальність: Головний клас, відповідає за взаємодію між програмою та користувачами завдяки «Telegram Bot API»

Атрибути: token

Нижче представлені функції класу

- run() – головна функція, запускає роботу програми;
- send_message() – відправити повідомлення користувачу.

Клас: «Spider»

Відповідальність: Клас, що збирає інформацію стосовно фільмів та кінотеатрів з веб-сторінок за посиланнями.

Атрибути: name, start_urls

Нижче представлені функції класу

- parse_films() – зібрати інформацію стосовно фільмів, що зараз у прокаті;
- parse_future_films() – зібрати інформацію стосовно майбутніх прем’єр;
- parce_cinemas_info() – зібрати інформацію стосовно кінотеатрів міста.

Клас: «Film»

Відповідальність: Клас, що описує роботу об’єкту «фільм».

Атрибути: film_name, film_age, film_imbd_rating, film_genre_info, film_producer, film_premiere, film_sessions, film_url.

Клас: «Cinema»

Відповідальність: Клас, що описує роботу об’єкту «кінотеатр».

Атрибути: cinema_name, cinema_address, cinema_phone, cinema_url.

2.2.3 Діаграми послідовностей

Діаграми послідовностей використовують для більш детального опису логіки сценаріїв використання. Діаграми послідовностей зазвичай містять об’єкти, які взаємодіють у рамках сценарію, повідомлення, якими вони обмінюються, і результати, пов’язані з повідомленнями, що звертаються [13].

На основі побудованої статичної моделі та діаграми варіантів використання розробимо динамічну модель програмного забезпечення. Діаграма послідовності для варіанту використання «отримання інформації» зображена на рисунку 2.9.

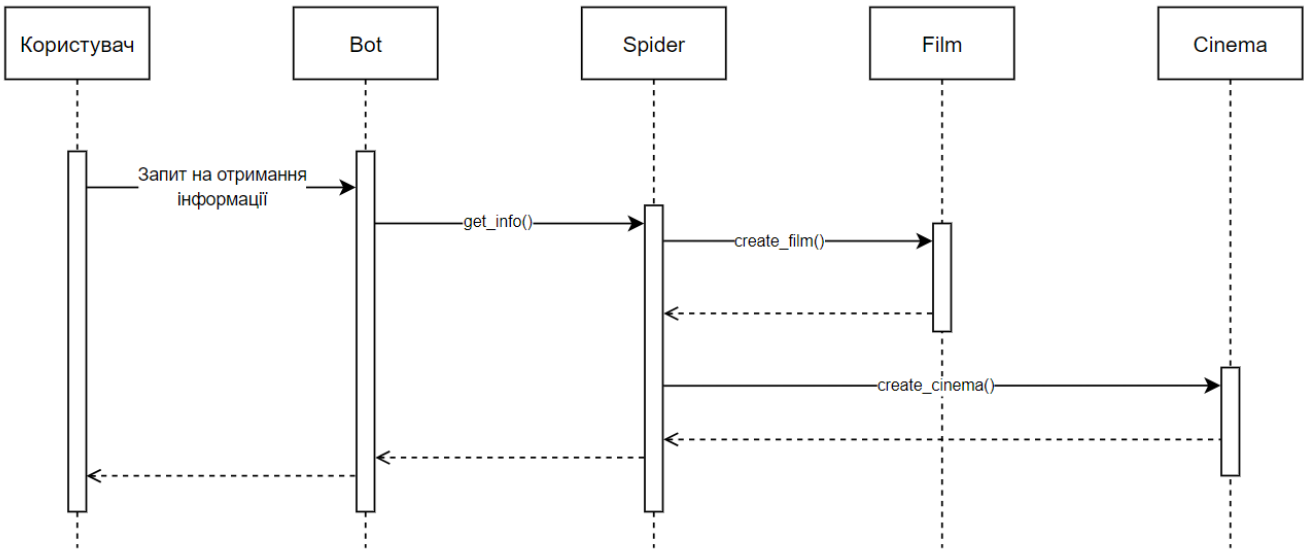

Рисунок 2.9 – Діаграма послідовності для варіанту використання «отримання інформації»

Таким чином було створено діаграму послідовності для зображення динамічної моделі програмного забезпечення.

2.3 Робочий проект програмного забезпечення

2.3.1 Вибір мови програмування

У ході розробки даного програмного забезпечення вирішено використовувати мову програмування – Python.

Python представляє з себе високорівневу мову програмування, яка призначена для створення програм різних типів. Це і веб-програми, і ігри, і десктопні програми, і робота з базами даних. Досить велике поширення пітон отримав у галузі машинного навчання та досліджень штучного інтелекту [14].

Основні особливості мови програмування Python:

– Скриптова мова. Код програм визначається у вигляді скриптів;

– Підтримка найрізноманітніших парадигм програмування, у тому числі об’єктно-орієнтованої та функціональної парадигм;

– Інтерпретація програм.

– Портативність та платформонезалежність. Не має значення, яка операційна встановлена – Windows, Mac OS, Linux, достатньо написати скрипт, який запускатиметься на всіх цих ОС за наявності інтерпретатора;

– Автоматичне керування пам’яті;

– Динамічна типізація.

Python дуже проста мова програмування, вона має короткий, простий та зрозумілий синтаксис. Відповідно його легко вивчати, і власне це одна з причин, через яку він є однією з найпопулярніших мов програмування для навчання. Python також популярний не тільки у сфері навчання, але й у написанні конкретних програм, у тому числі комерційного характеру. Тому для цієї мови написано багато бібліотек. Крім того, у даної мови програмування дуже велике ком’юніті, в інтернеті можна знайти безліч корисних матеріалів, прикладів, отримати кваліфіковану допомогу фахівців.

2.3.2 Фізична модель даних

На етапі створення фізичної моделі детальне проектування виконується з використанням діаграми компонентів, яка наведена на рисунку 2.10.

Діаграми компонентів – це, в основному, діаграми класу, що фокусуються на компонентах системи, часто використовуються для моделювання статичного вигляду реалізації системи [15].

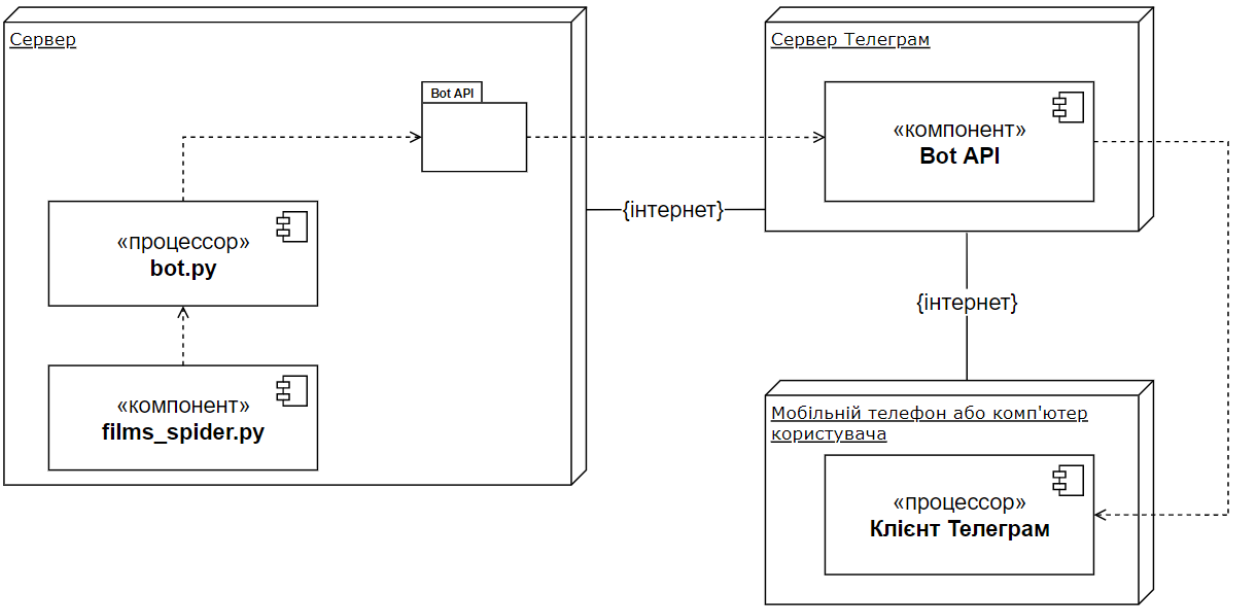

Рисунок 2.10 – Діаграма компонентів

Таким чином було побудовано діаграму компонентів, яка демонструє взаємозв’язок між різними компонентами, описує особливості фізичного представлення та дозволяє визначити архітектуру розроблюваної системи.

2.3.3 Реалізація та тестування програмного забезпечення

ПЗ було написано мовою програмування Python, з використанням технологій інтерфейсу Telegram Bot API.

Для тестування була обрана лише основна функція, а саме варіант використання «отримання інформації».

Поведінкове тестування, або тестування чорного ящика – метод тестування функціональної поведінки програми з точки зору зовнішнього світу, при якому знання про внутрішній устрій не використовуються, тобто тестувальники не мають доступу до вихідного коду тестованого об’єкту. Стратегією такого виду тестування є систематичний метод відбору і створення тестів для тестового набору [16].

Тестування варіанту використання «Отримання інформації»

Для отримання інформації стосовно показів на неділю, майбутніх прем’єр або контактної інформації кінотеатрів, користувачеві необхідно обрати бажаний варіант на клавіатурі бота або коректно ввести його самому завдяки клавіатурі телефона, комп’ютера чи планшетного ПК. Правильними вхідними даними є:

– Пункт інформації: поточна текстова назва.

Для даного варіанту використання створимо 6 тестів:

Тест 1

Вхідні дані:

– Пункт інформації: не коректна текстова назва.

Очікуваний результат:

– Після некоректного вводу тексту користувачем, бот повинен вивести повідомлення про помилку та змінити розкладку клавіатури на клавіатуру бота.

Результат тестування представлено на рисунку 2.11:

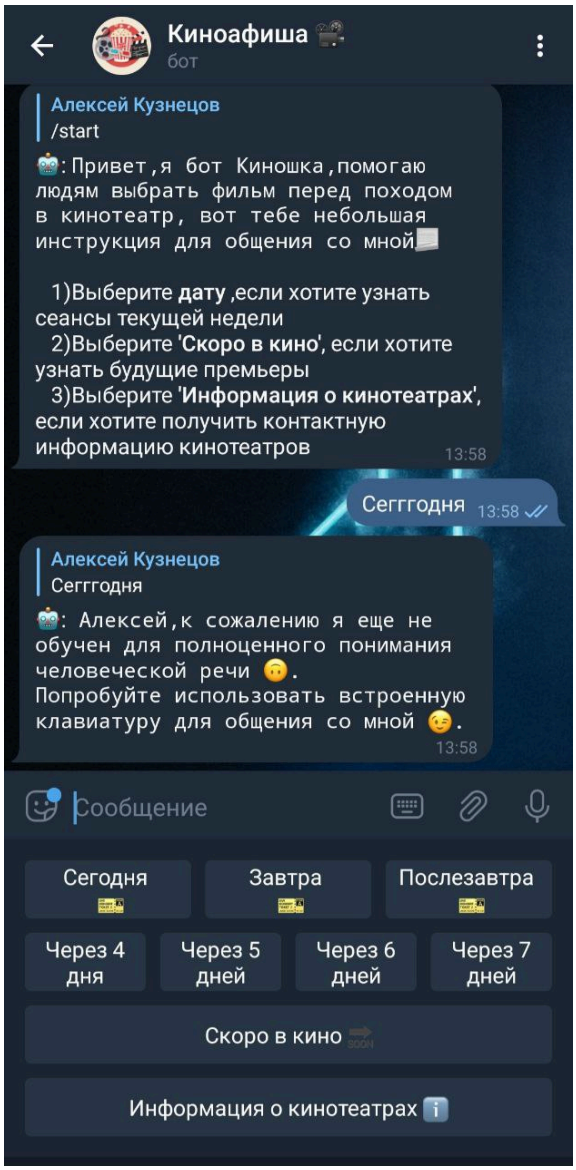

Рисунок 2.11 – Результат тестування варіанту використання «Отримання інформації» (тест 1)

Тест 2

Вхідні дані:

– Пункт інформації: фото.

Очікуваний результат:

– У випадку відправки користувачем фото, буде виведено повідомлення про некоректний запит, та на екрані з’явиться клавіатура бота.

Результати тестування представлений на рисунку 2.12:

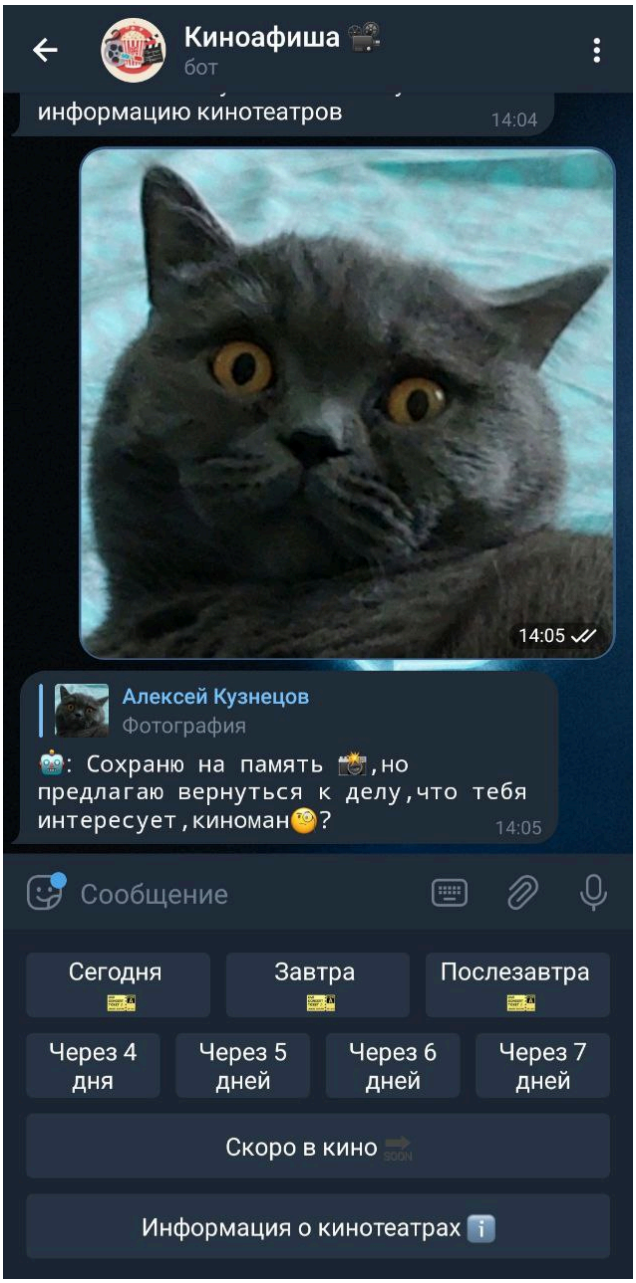

Рисунок 2.12 – Результат тестування варіанту використання «Отримання інформації» (тест 2)

Тест 3

Вхідні дані:

– Пункт інформації: аудіофайл.

Очікуваний результат:

– У випадку відправки користувачем аудіофайлу, буде виведено повідомлення про некоректний запит, та на екрані з’явиться клавіатура бота.

Результат тестування представлений на рисунку 2.13:

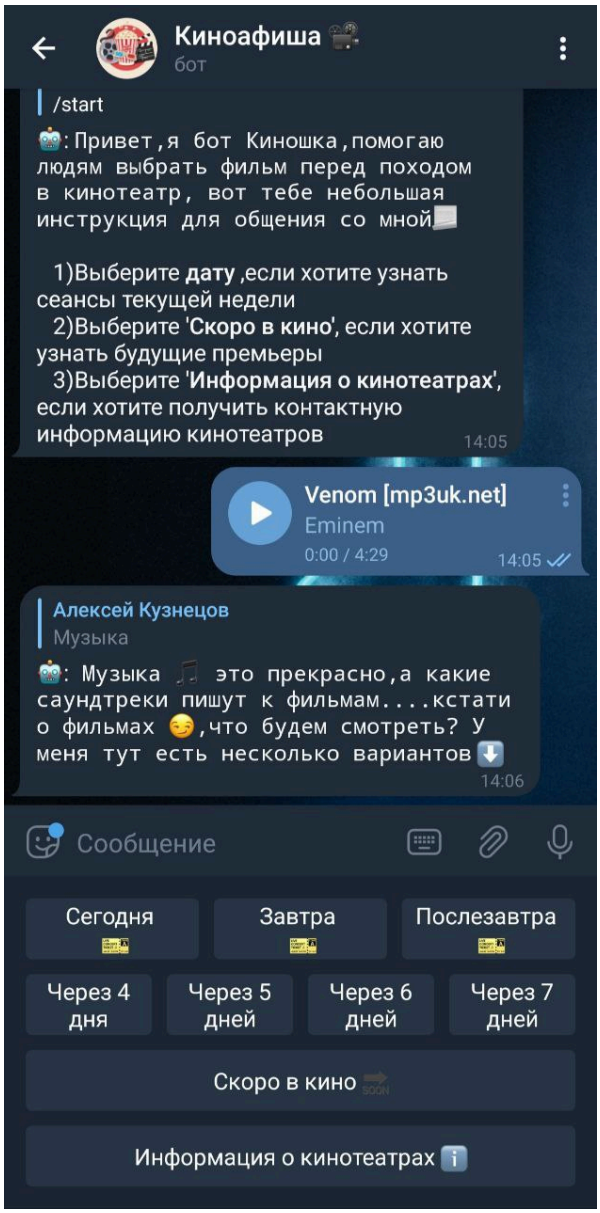

Рисунок 2.13 – Результати тестування варіанту використання «Отримання інформації» (тест 3)

Тест 4

Вхідні дані:

– Пункт інформації: відеофайл.

Очікуваний результат:

– У випадку відправки користувачем відеофайлу, буде виведено повідомлення про некоректний запит, та на екрані з’явиться клавіатура бота.

Результат тестування представлений на рисунку 2.14:

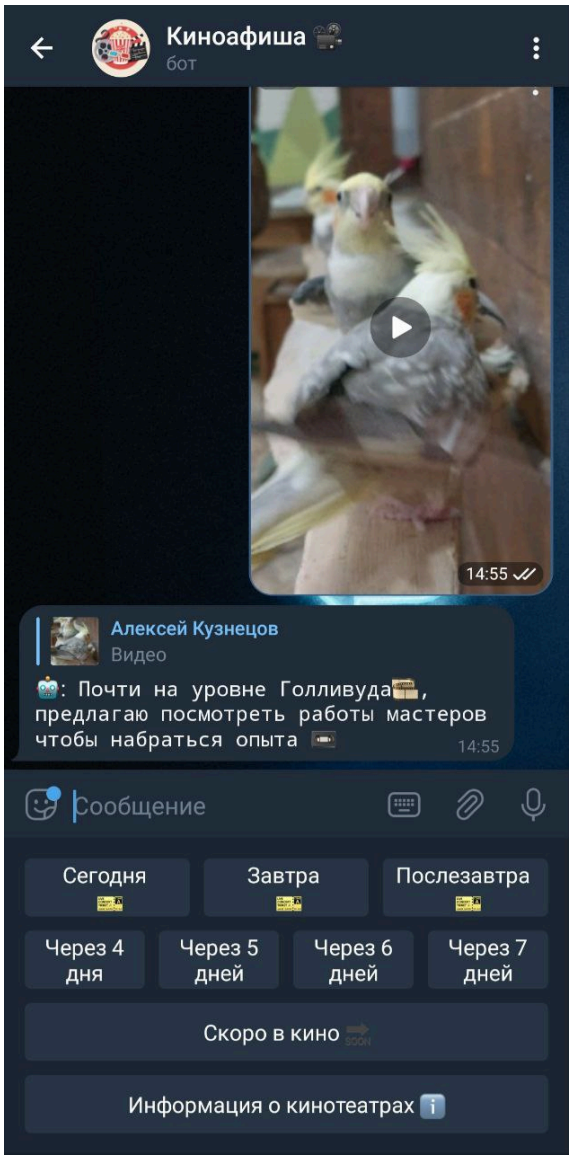

Рисунок 2.14 – Результат тестування варіанту використання «Отримання інформації» (тест 4)

Тест 5

Вхідні дані:

– Пункт інформації: стікер.

Очікуваний результат:

– У випадку відправки користувачем стікеру мережі Telegram, буде виведено повідомлення про некоректний запит, та на екрані з’явиться клавіатура бота.

Результат тестування представлений на рисунку 2.15:

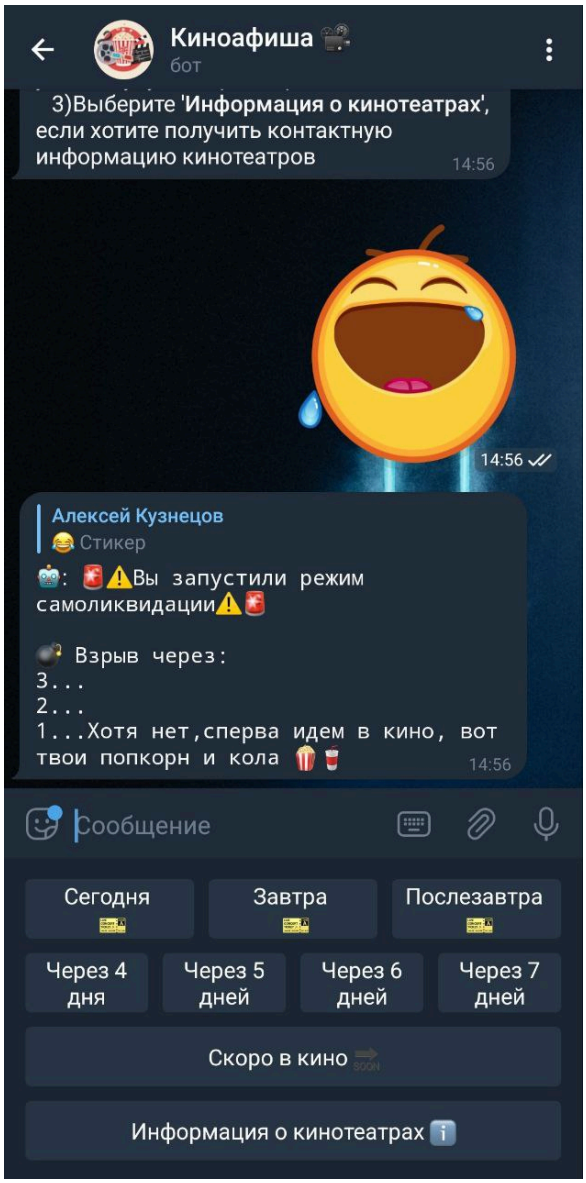

Рисунок 2.15 – Результат тестування варіанту використання «Отримання інформації» (тест 5)

Тест 6

Вхідні дані:

– Пункт інформації: поточна текстова назва.

Очікуваний результат:

– На етапі введення пункту інформації не повинно виникнути ніяких помилок.

Результат тестування представлений на рисунку 2.16:

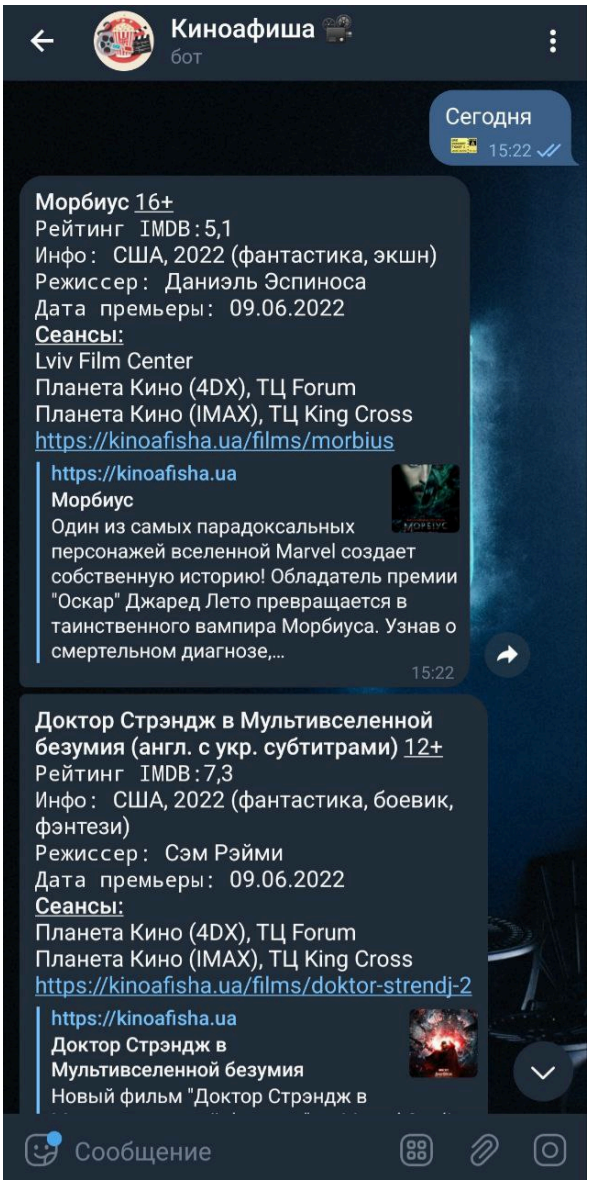

Рисунок 2.16 – Результат тестування варіанту використання «Отримання інформації» (тест 6)

Отже, в ході тестування варіанту використання «Отримання інформації» не було знайдено ніяких помилок.

2.3.4 Випробування програмного забезпечення

Програму і методику випробувань розробленого ПЗ наведено в Додатку В.

На основі інструкцій, що наведені в цьому додатку, проведемо випробування розробленого програмного забезпечення.

Після запуску бота, користувачу виводить на екран вітальне повідомлення, та стисла інструкція з користування ботом. Вітальне повідомлення зображене на рисунку 2.17:

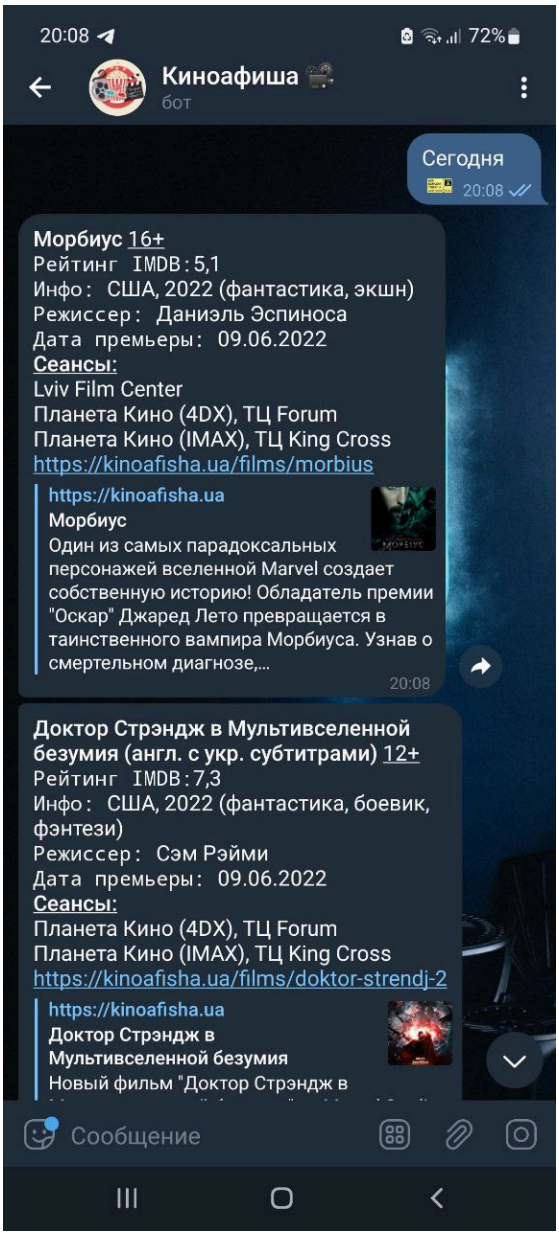

Рисунок 2.17 – Вітальне повідомлення

Отримання інформації стосовно показів на тиждень, майбутніх прем’єр та контактна інформація кінотеатрів міста зображені на рисунках 2.18 – 2.23:

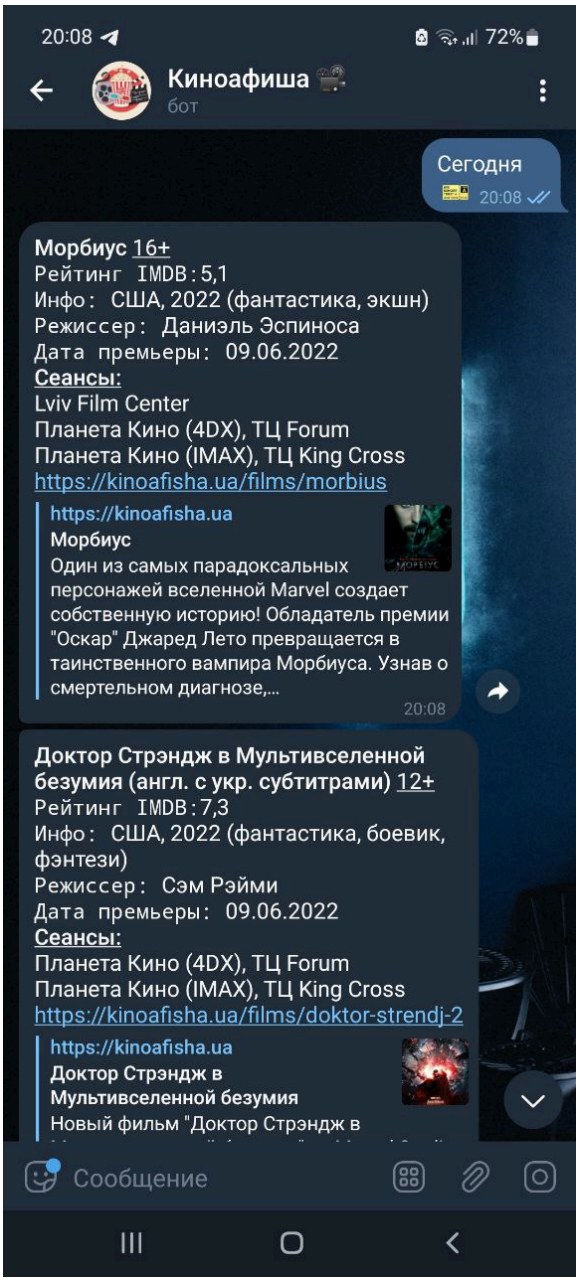

Рисунок 2.18 – Інформація стосовно показів на сьогодні

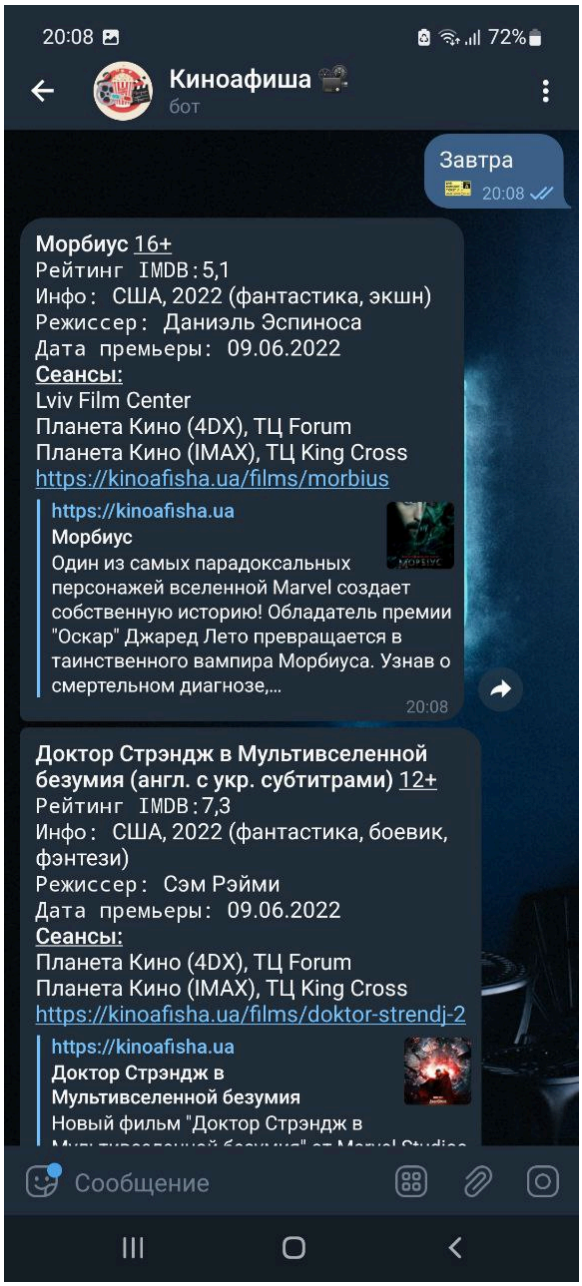

Рисунок 2.19 – Інформація стосовно показів на завтра

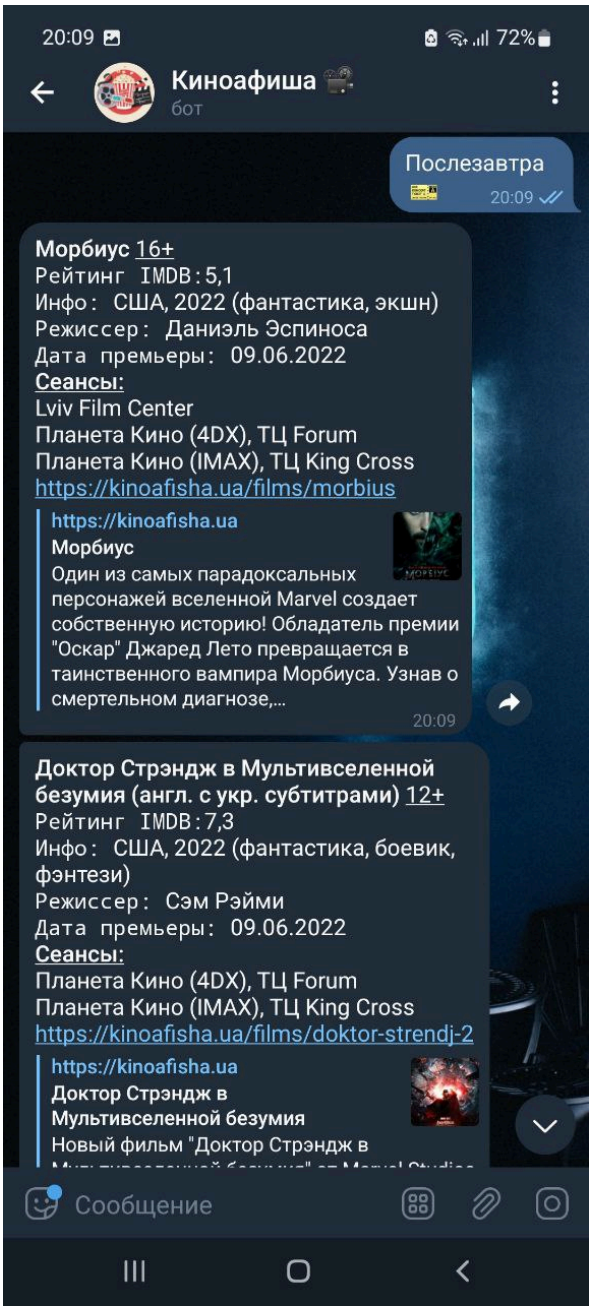

Рисунок 2.20 – Інформація стосовно показів на післязавтра

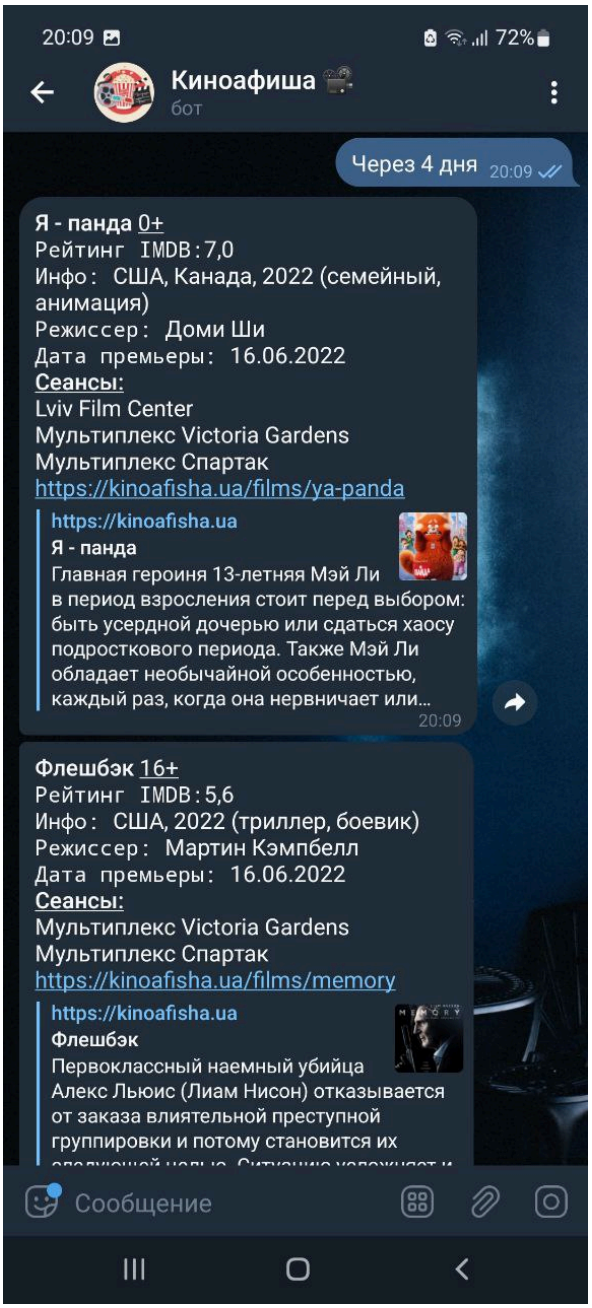

Рисунок 2.21 – Інформація стосовно показів на четвертий день тижня

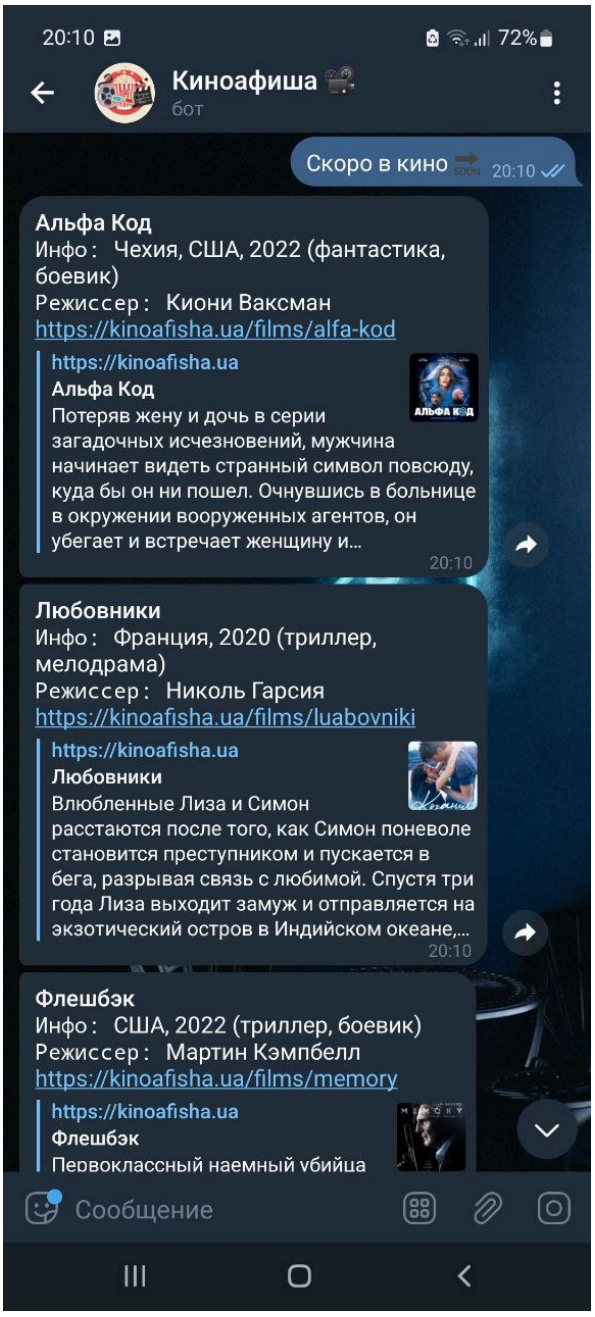

Рисунок 2.22 – Інформація стосовно майбутніх прем’єр

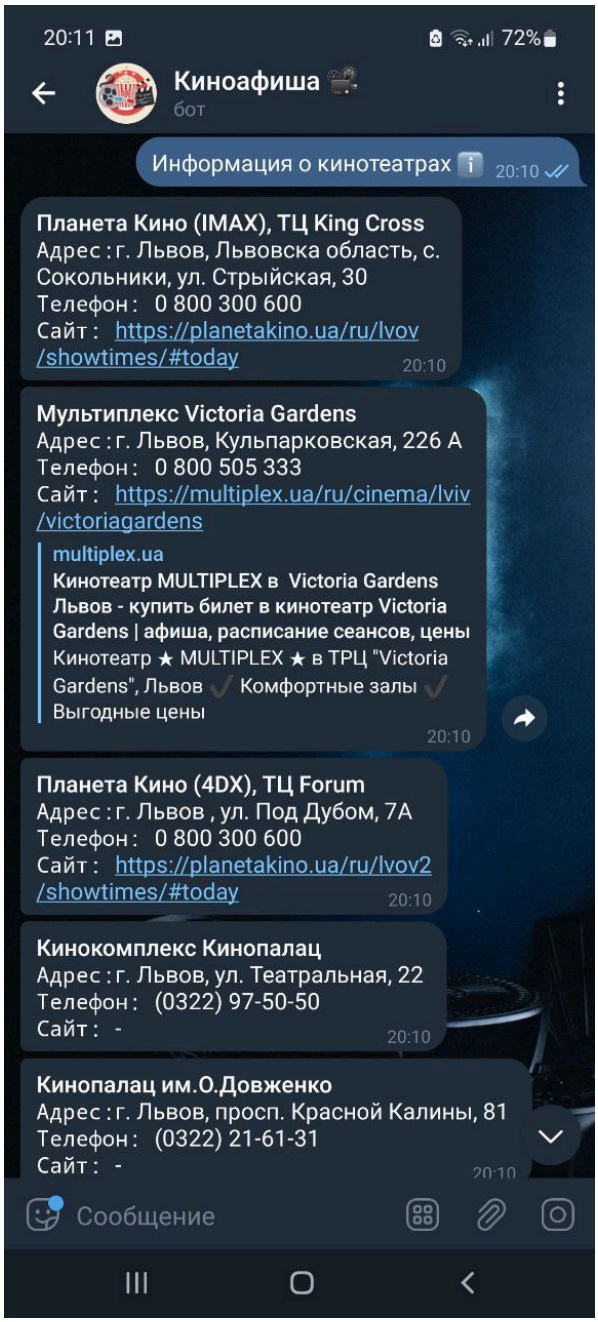

Рисунок 2.23 – Контактна інформація кінотеатрів міста

Таким чином було проведено випробування розробленого програмного забезпечення за інструкціями, що наведено у Додатку В – «Програма та методика випробування ПЗ».

3 РЕЗУЛЬТАТИ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТЕЛЕГРАМ-БОТУ КІНОАФІШИ МІСТА МИКОЛАЄВА

В процесі виконання кваліфікаційної роботи було розроблено ПЗ телеграм-боту кіноафіши міста Миколаєва. Розроблене програмне забезпечення задовольняє вимогам, які поставлені у технічному завданні, а саме: збір та надання інформації стосовно фільмів, поточних показів, майбутніх показів та кінотеатрів.

В даній кваліфікаційній роботі було проведено огляд нормативних документів та Internet-джерел для аналізу роботи інформаційних систем кінотеатрів, месенджерів та ботів, і методів можливої розробки телеграм-боту кіноафіши міста Миколаєва.

Було проведено пошук та аналіз чинних програмних рішень для автоматизації надання і перегляду інформації кіноафіши для користувачів.

Здійснено постановку задачі та розроблено технічне завдання для розробки програмного забезпечення телеграм-боту кіноафіши міста Миколаєва, що наведену в додатку А – «Технічне завдання».

У рамках даної кваліфікаційної роботи було виконано ескізний, технічний та робочий проекти.

У рамках ескізного проекту було обрано мову моделювання UML, засобами якої було побудовано модель варіантів використання, інформаційну модель та розроблено ескіз інтерфейсу користувача.

У рамках технічного проекту було розроблено діаграму класів для статичного представлення системи та діаграму послідовності для динамічного представлення системи.

У рамках робочого проекту було розроблено фізичну модель даних яка представлена діаграмою компонентів, проведено тестування та випробування ПЗ, що показало повну працездатність та відповідність всім вимогам, які представлені у технічному завданні для розробки телеграм-боту кіноафіши міста Миколаєва.

При розробці програмного забезпечення використовувалась мова програмування Python.

Текст програми складається з 274 рядків і займає 54 МБ дискового простору.

Текст програми наведено у Додатку Б – «Текст програми».

Після проведеного тестування про її характеристик та виконуваних функцій технічному завданню, готовність до впровадження. Випробування були виконані згідно з програмою та методикою випробувань, яку наведено у Додатку В – «Програма та методика випробування ПЗ».

Опис програми наведено у Додатку Г – «Опис програми».

На програмному забезпеченні сервера повинен бути встановлений інтерпретатор мови програмування Python.

Для взаємодії з телеграм ботом, користувачеві достатньо мати встановлений додаток Telegram на своєму пристрої, або використовувати веб-версію додатку при роботі з ПК.

Інструкцію користувача для роботи з даним телеграм ботом наведено в Додатку Д.

4 ОХОРОНА ПРАЦІ

Охорона праці – це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності [17].

Мета охорони праці – забезпечення безпечних, нешкідливих і сприятливих умов праці через вирішення багатьох складних завдань, основними з яких є:

- проектування підприємств, технологічних процесів і конструювання обладнання з обов'язковим виконанням вимог охорони праці;
- знаходження оптимальних співвідношень між різними факторами виробничого середовища, що дозволяє забезпечити мінімум несприятливого впливу їх на здоров'я працівників;
- встановлення, законодавче оформлення визначених норм кожного з несприятливих або небезпечних факторів, систематичний контроль за їх застосуванням;
- розробка конкретних заходів щодо покращення умов праці та забезпечення її безпеки на основі застосування у виробництві новітніх досягнень науки і техніки;
- застосування раціональних засобів захисту працівників від впливу несприятливих факторів виробничого середовища, а також втілення організаційних заходів, які нейтралізують або послаблюють ступінь їх впливу на організм людини;
- розробка та застосування методів і засобів оцінки ефективності заходів з охорони праці, що плануються і здійснюються [18].

Основним законом в галузі охорони праці є Закон України «Про охорону праці». Дія Закону України «Про охорону праці» поширюється на всіх юридичних та фізичних осіб, які відповідно до законодавства використовують найману працю, та на всіх працюючих.

4.1 Аналіз шкідливих та небезпечних факторів на робочому місці

Під час укладання трудових договорів роботодавець повинен проінформувати працівника під розписку про умови праці та про наявність на його робочому місці небезпечних і шкідливих виробничих факторів, які ще не усунуто, можливі наслідки їх впливу на здоров'я та про права працівника на пільги і компенсації за роботу в таких умовах відповідно до законодавства і колективного договору.

Шкідливий виробничий фактор – виробничий фактор, вплив якого може призвести до погіршення стану здоров'я, зниження працездатності працівника.

Небезпечний виробничий фактор – виробничий фактор, дія якого за певних умов може призвести до травм або іншого раптового погіршення здоров'я працівника.

В таблиці 4.1 перераховані шкідливі та небезпечні виробничі фактори.

Таблиця 4.1 – Шкідливі та небезпечні виробничі фактори.

Найменування фактору	Можливі джерела його виникнення	Характер дії
Небезпека ураження електричним струмом	Мережа живлення	Небезпечний
Пожежонебезпечність приміщень	Наявність матеріалів, що загоряються і джерел запалення	Небезпечний та шкідливий
Електромагнітне випромінювання в тому числі і рентгенівське	ЕПТ (Дисплей с джерелом рентгенівського, радіочастотного, ультрафіолетового й інфрачервоного випромінювання та випромінювання звукового діапазону)	Шкідливий
Несприятлива освітленість	Недостатнє штучне і природне освітлення	Шкідливий
Психофізіологічні напруження	Монотонність праці, перенапруженість зорових аналізаторів, розумова напруженість, незручність і статичність пози	Шкідливий

Тривала і інтенсивна робота за комп’ютером може стати джерелом важких професійних та звичайних захворювань таких як:

- Неправильна постава – це призводить до подальшого розвитку викривлення хребта, сколіозу, лордозу, кіфозу, а як результат голові болі, болі в області шиї і всього хребта, болі в області тазу.

- Порушення фокусування – є наслідком напруженої роботи за монітором комп’ютера, може бути викликана перенапруженням очних м’язів.

- Тунельний синдром або синдром зап’ястного каналу – проявляється після напруженої роботи за ПК.

- Короткозорість – розвивається через те, що екран монітора за контрастом вище ніж навколишні об’єкти.

- Сухість очей – при погляді на джерело світла людина починає менше моргати та виникає «обсушування» рогівки ока, що призводить до болів.

- Геморой – причиною якого стає застій крові у венах.

4.2 Заходи щодо зменшення шкідливих факторів

Перш ніж починати роботу за ПК рекомендується обладнати робоче місце, а саме:

- Висоту робочої поверхні столу відрегулювати в межах 680-800 мм, при відсутності регулювання висота робочої поверхні столу повинна бути 725 мм.

- Робочий стіл повинен мати простір для ніг висотою не менше 600 мм, шириною не менше 500 мм, глибиною на рівні колін не менше 450 мм і для витягнутих не менше 650 мм.

- Клавіатуру розташувати на поверхні столу на відстані 100 – 300 мм від краю, зверненого до користувача, або на спеціальній регульованій по висоті стільниці.

- Рівень очей при вертикально розташованому екрані має припадати на центр або 2/3 висоти екрану, лінія погляду повинна бути перпендикулярна центру екрану.

Рекомендації під час роботи за ПК:

- Для зменшення негативного впливу на здоров’я працівників виробничих факторів необхідно застосовувати регламентовані перерви.

- Тривалість безперервної роботи за візуальним дисплейним терміналом без регламентованої перерви не повинна перевищувати 2 години.

- Під час перерви не варто читати або дивитися телевізор.

- У робочому приміщенні, де розташовані комп’ютери, щодня потрібно виконувати вологе прибирання. Також приміщення потрібно провітрювати щогодини.

- Необхідно слідкувати за станом екрану монітора, який має бути без пилу та плям.

- Не рекомендовано використовувати звичайний телевізор замість монітора.

- У процесі роботи рекомендується періодично (приблизно раз на 20-30 хвилин) переводити погляд з екрана на найбільш віддалений предмет у кімнаті, а ще краще – на віддалений об’єкт за вікном.

- Якщо з’явилося відчуття втоми, напруження, сонливості, тяжкості в очах, потрібно припинити роботу та хоча б трохи відпочити.

- Потрібно слідкувати за поставою: ноги повинні твердо стояти на підлозі чи на спеціальній підставці: стегна повинні бути під прямим кутом до

- тулуба, а гомілки – під прямим кутом до стегон; сидіти потрібно прямо або злегка нахилившись вперед; відстань від очей до екрана монітора – не менше 55-60 см; центр екрану має знаходитися на рівні очей чи трохи нижче; рекомендується хоча б раз на день виконувати гімнастику для очей.

- Щоб уникнути синдрому «сухого ока» рекомендовано моргати кожні 3-5 секунд для зволоження рогівки ока [19].

Одна з причин, яка викликає втому очей при роботі за комп’ютером – постійне мерехтіння і світлова пульсація картинки на моніторі [20].

Вправи для зняття втоми очей:

- Розітріть долоні, щоб у них з’явилося відчуття тепла, складіть їх навхрест, прикладіть до повік, передаючи таким чином їх енергію тепла до очей. Затримайтеся в такому положенні на 30-40 секунд. Після закрийте очі і повільно натискайте на повіки кінчиками пальців, по 20 разів.

- Закрийте очі, зробіть по 10 обертаючих рухів очними яблуками за годинниковою стрілкою і проти неї. Після цього, заплющивши очі, відкрийте очі. Зробити п’ять повторень.

- Зробіть кілька постукуючих легких рухів пальцями двох рук по голові, почавши з чола і поступово перейшовши до потилиці.

- Моргніть очима два рази, сильно і різко примружтеся і різко відкрийте очі. Зробіть 10 разів.

- Сильно розітріть долоні, щоб відчути теплоту, а після погладжуючими рухами п’ять разів натисніть на очні яблука [20].

ВИСНОВКИ

У даній кваліфікаційній роботі був виконаний аналіз та постановка задачі до розробки програмного забезпечення телеграм-боту кіноафіши міста Миколаєва, розроблено проект програмного забезпечення, а також представлені результати розробки та спеціальний розділ з охорони праці.

У даній кваліфікаційній роботі було вирішено практичну задачу інженерії програмного забезпечення, що характеризується комплексністю та невизначеністю умов.

Задача розробленого програмного забезпечення – полегшення інформування мешканців міста стосовно сеансів у кінотеатрах Миколаєва та покращення зручності надання інформації користувачу.

Також у даній кваліфікаційній роботі було розроблено технічне завдання, опис програми, програми та методики випробувань програмного продукту, та інструкцію користувача за темою кваліфікаційної роботи «Розробка програмного забезпечення телеграм-боту кіноафіши м. Миколаєва».

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Інформаційний сервіс афіши кінопалацу Multiplex Victoria Gardens «multiplex.ua». URL: https://multiplex.ua/cinema/lviv/victoriagardens (дата звернення: 03.03.2022).

2. Інформаційний сервіс афіши кінопалацу Планета Кіно King Cross «planetakino.ua». URL: https://planetakino.ua/ru/lvov/showtimes/#today (дата звернення: 03.03.2022).

3. Пошук та підбір сервісів для бізнесу. URL: https://startpack.ru/application/ (дата звернення: 26.04.2022).

4. Інформаційний сервіс афіши кіно «i.ua». URL: http://kino.i.ua/afisha/ (дата звернення: 04.03.2022).

5. Інформаційний сервіс афіши кіно у містах України «kinofilms.ua». URL: https://www.kinofilms.ua/afisha/ (дата звернення: 04.03.2022).

6. Інформаційний сервіс афіши кіно у містах України «kinoafisha.ua». URL: https://kinoafisha.ua/ua/kinoafisha/ (дата звернення: 05.03.2022).

7. Що таке уніфікована мова моделювання (uml)?. URL: https://uk.theastrologypage.com/unified-modeling-language (дата звернення: 02.05.2022).

8. Використання діаграми варіантів використання UML під час проектування програмного забезпечення. URL: https://habr.com/ru/post/566218/ (дата звернення: 02.05.2022).

9. Діаграма діяльності. URL: https://flexberry.github.io/ru/fd_activity-diagram.html (дата звернення: 04.05.2022).

10. Навчальний посібник з «Остаточна діаграма взаємозв’язків між сутностями» (ER діаграми). URL: https://creately.com/blog/ru/uncategorized-ru/ (дата звернення: 05.05.2022).

11. Що таке інтерфейс. URL: https://uk.encyclopedia-titanica.com/significado-de-interfaz (дата звернення: 06.05.2022).

12. Діаграма класів. URL: https://flexberry.github.io/ru/fd_class-diagram.html (дата звернення: 07.05.2022).

13. Діаграма послідовності. URL: https://flexberry.github.io/ru/fd_sequence-diagram.html (дата звернення: 08.05.2022).

14. Мова програмування Python. URL: https://metanit.com/python/tutorial/ (дата звернення: 12.05.2022).

15. Діаграма компонентів UML. URL: https://uk.education-wiki.com/2574102-uml-componentdiagram (дата звернення: 15.05.2022).

16. White/Black/Grey Box-тестування. URL: https://qalight.ua/ru/baza-znaniy/white-black-grey-box-testirovanie/ (дата звернення: 17.05.2022).

17. Про охорону праці: Закон України від 21.11.2002 р. № 229-IV. URL: https://zakon.rada.gov.ua/laws/show/229-15 (дата звернення: 17.03.2022).

18. Поняття, мета і завдання охорони праці. URL: https://pidru4niki.com/18211001/bzhd/vstup_ohorona_pratsi_v_turizmi (дата звернення: 17.03.2022).

19. Інструкція з охорони праці при роботі на персональному комп’ютері. URL: https://services.uteka.ua/publication/zrazky-34-trudovi-vidnosyny-ta-oplata-pratsi-138-instrukciya-po-oxrane-truda-pri-rabote-na-personalnom-kompyutere-obrazec (дата звернення 19.03.2022).

20. Кращі вправи для очей під час роботи з комп’ютером. URL: https://fitness.org.ua/krashi-vprava-dlia-ochei-vid-vtomi-pislia-roboti-z-komputer/ (дата звернення 20.03.2022).

ДОДАТОК А – ТЕХНІЧНЕ ЗАВДАННЯ

ВСТУП

Найменування програмного додатку, що розробляється – «Telegram-bot of the movie posters of Mykolaiv». Область застосування – месенджер Telegram.

1 ПІДСТАВА ДЛЯ РОЗРОБКИ

Розробка програмного забезпечення виконується на підставі завдання на кваліфікаційну роботу, виданого кафедрою ПЗАС НУК імені адмірала Макарова.

2 ПРИЗНАЧЕННЯ РОЗРОБКИ

2.1 Експлуатаційне призначення

Експлуатаційним призначенням програмного забезпечення телеграм-боту кіноафіши є полегшення інформування мешканців міста стосовно сеансів у кінотеатрах Миколаєва та покращення зручності надання інформації користувачу.

2.2 Функціональне призначення

Функціональним призначенням програмного забезпечення телеграм-боту кіноафіши м. Миколаєва є:

- збір та надання інформації стосовно фільмів;
- збір та надання інформації стосовно поточних показів;
- збір та надання інформації стосовно майбутніх показів;
- збір та надання інформації стосовно кінотеатрів.

3 ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Вимоги до функціональних характеристик

3.1.1 Вимоги до складу виконуваних функцій

Програмний додаток повинен виконувати всі нижчеописані функції.

Збір інформації з сайтів кінотеатрів

Збір інформації стосовно фільму (назва, країна, рік випуску, жанр, режисер, дата прем’єри, рейтинг IMDB, вікове обмеження), сеансів на тиждень (назва кінотеатру), майбутніх показів (назва фільму, країна, жанр) та контактної інформації кінотеатру (назва кінотеатру, адреса, телефон гарячої лінії, адреса сайту кінотеатру) повинен відбуватись автоматично через певний проміжок часу.

Обробка зібраної інформації

Формулювання структурованих блоків з інформацією зібраною з сайтів повинно відбуватися автоматично після її збору з сайтів кінотеатрів.

Формування файлу з обробленою інформацією

Після збору та обробки інформації повинен здійснюватися автоматичний запис оброблених блоків інформації у файл.

Видача обробленої інформації

Видача ботом інформації з файлу де вона записана відбувається шляхом введення користувачем спеціальної команди яка відповідає за той тип інформації, яка цікава користувачеві або шляхом натискання клавіши на спеціальній клавіатурі.

3.1.2 Вимоги до організаційних вхідних та вихідних даних

Дані зберігаються у файлі формату .json.

Вхідні дані: збір інформації стосовно кіноафіши відбувається автоматично, введення інформації від користувача відбувається за допомогою пристроїв введення (клавіатура та миша)

Вихідні дані: виведення даних здійснюється за допомогою монітора.

3.2 Вимоги до надійності

Програмний продукт повинен функціонувати без збоїв при умові безперервної роботи серверу на якому міститься ПЗ, стабільній роботі самого месенджеру Telegram та стабільному підключенні до мережі Internet. Сама інформація повністю оновлюється через деякий проміжок часу шляхом перезапису файлу де вона міститься і зберігається там до наступного оновлення файлу тим самим мінімізуючи факт її втрати.

У випадку введення користувачем некоректної команди, бот не відреагує на неї, а для відсутності необхідності вводу користувачем команд, бот має свою спеціальну клавіатуру.

3.3 Вимоги до умов експлуатації

Необхідний рівень підготовки користувача:

- мінімальні навики володіння комп’ютером або смартфоном;
- вміння користуватися будь-якою пошуковою системою мережи Internet;
- мінімальні навички роботи з месенджером Telegram.

3.4 Вимоги до складу та параметрів технічних засобів

Комп’ютер повинен мати наступну мінімальну комплектацію:

- CPU: Intel(R) Celeron(R) 2.16 МГц;
- RAM: 2 GB;
- HDD: 1 ГБ вільного місця

Мінімальна комплектація смартфону:

– Версія Android 8.0 / iOS 12.4.9
– ОЗУ 3 ГБ
– 1 ГБ вільної пам’яті

3.5 Вимоги до інформаційної програмної сумісності

Для повноцінного функціонування системи на комп’ютері необхідно використовувати операційну систему Windows версії не нижче 7. Також рекомендовано встановити десктопну версію месенджеру Telegram.

3.6 Вимоги до маркування та пакування

Продукт буде розміщено на хмарних серверах Heroku.

4 ВИМОГИ ДО ПРОГРАМНОЇ ДОКУМЕНТАЦІЇ

Програмна документація повинна містити:

– технічне завдання;
– текст програми;
– опис програми;
– інструкцію користувача;
– програму і методику випробувань;

5 СТАДІЇ ТА ЕТАПИ РОЗРОБКИ

Стадії та етапи розробки представлені в таблиці А.1.

Таблиця А.1 – Стадії та етапи розробки програмного забезпечення

Стадії розробки	Етапи робіт	Вміст робіт	Контрольні точки
Технічне завдання	Розробка, узгодження та затвердження технічного завдання	Постановка задачі. Визначення та уточнення вимог до технічних засобів. Визначення вимог до програми і документації на неї. Узгодження і затвердження технічного завдання.	01.03.2022 – 31.03.2022
Ескізний проект	Розробка та затвердження ескізного проекту	Виявлення вимог, складання ескізної частини програмного забезпечення.	31.03.2022 – 27.04.2022
Технічний проект	Розробка та затвердження технічного проекту	Загальні системні рішення, алгоритми задач, оцінка ефективності автоматизованої системи.	27.04.2022 – 06.05.2022
Робочий проект	Розробка та затвердження робочого проекту	Деталізовані загальносистемні у проекті рішення, програми та інструкції для розв’язку завдань, розробка програмного забезпечення.	06.05.2022 – 13.05.2022

6 ПОРЯДОК КОНТРОЛЮ ТА ПРИЙОМУ

Контроль за аналізом та проектуванням кожної окремої частини програмного забезпечення відбувається на кожному етапі з урахуванням вимог, визначених у технічному завданні. Кожна стадія розробки повинна бути представлена в зазначені строки та узгоджена із замовником.

Прийом проводиться відповідно до програми і методики випробувань і документується за допомогою протоколу проведення випробувань. У разі знаходження помилок під час прийому програмного забезпечення, складається акт про знайдені помилки, який підписується представниками замовника і розробника затверджується керівником організації-замовника та організації-розробника. Розробник повинен, на протязі не більше ніж 2 тижні, виправити зазначені помилки і оповістити замовника про повторне проведення перевірки.

ДОДАТОК Б – ТЕКСТ ПРОГРАМИ

bot.py

```
import json
from config import token
from aiogram import Bot, Dispatcher, executor, types
from aiogram.dispatcher import FSMContext
from aiogram.dispatcher.filters import Text
from aiogram.types import ReplyKeyboardRemove, ReplyKeyboardMarkup, KeyboardButton, InlineKeyboardMarkup, \
    InlineKeyboardButton, Message

bot = Bot(token=token, parse_mode=types.ParseMode.HTML)
dp = Dispatcher(bot)

btn_day1 = KeyboardButton('Сегодня \n 📼')
btn_day2 = KeyboardButton('Завтра \n 📼')
btn_day3 = KeyboardButton('Послезавтра \n 📼')
btn_day4 = KeyboardButton('Через 4 дня')
btn_day5 = KeyboardButton('Через 5 дней')
btn_day6 = KeyboardButton('Через 6 дней')
btn_day7 = KeyboardButton('Через 7 дней')
btn_future_films = KeyboardButton('Скоро в кино 🔜')
btn_cinemas_info = KeyboardButton('Информация о кинотеатрах ℹ️')

markup = ReplyKeyboardMarkup(resize_keyboard=True).row(btn_day1, btn_day2, btn_day3) \
    .row(btn_day4, btn_day5, btn_day6, btn_day7) \
    .add(btn_future_films).add(btn_cinemas_info)


@dp.message_handler(commands='start')
async def start(message: types.Message):
    await message.reply(
```

"□: <code>Привет,я бот Киношка,помогаю людям выбрать фильм перед походом в кинотеатр, вот тебе небольшая инструкция для общения со мной</code>🎩 \n\n 1)Выберите дату ,если хотите узнать сеансы текущей недели\n 2)Выберите 'Скоро в кино', если хотите узнать будущие премьеры\n 3)Выберите 'Информация о кинотеатрах', если хотите получить контактную информацию кинотеатров",
reply_markup=markup)

```
@dp.message_handler(
    lambda message: message.text not in ['Сегодня \n 📼', 'Завтра \n 📼', 'Послезавтра \n 📼', 'Через 4 дня',
                                         'Через 5 дней', 'Через 6 дней', 'Через 7 дней', 'Скоро в кино 🔜',
                                         'Информация о кинотеатрах ℹ', 'Бизяна'])
async def invalid_message(message: types.Message):
    return await message.reply(
        f"□:<code> {message.from_user.first_name},к сожалению я еще не обучен для полноценного понимания человеческой речи □.\nПопробуйте использовать встроенную клавиатуру для общения со мной ☺.</code>", reply_markup=markup)


@dp.message_handler(content_types=["sticker"])
async def invalid_message_type(message: types.Message):
    return await message.reply("□:<code> 🕰⚠□Вы запустили режим самоликвидации⚠□🕰\n\n💣 Взрыв через: \n3...\n2...\n1...Хотя нет,сперва идем в кино, вот твои попкорн и кола □□</code>", reply_markup=markup)


@dp.message_handler(content_types=["photo"])
async def invalid_message_type(message: types.Message):
```

```
        return await message.reply("□:<code> Сохраню на память □□</code><code>,но предлагаю вернуться к делу,что тебя интересует,киноман</code><code>□?</code>", reply_markup=markup)


    @dp.message_handler(content_types=["video"])
    async def invalid_message_type(message: types.Message):
        return await message.reply("□:<code> Почти на уровне Голливуда□□, предлагаю посмотреть работы мастеров чтобы набраться опыта 📼</code>", reply_markup=markup)


    @dp.message_handler(content_types=["audio"])
    async def invalid_message_type(message: types.Message):
            return await message.reply("□:<code> Музыка ♪ это прекрасно,а какие саундтреки пишут к фильмам....кстати о фильмах 😉,что будем смотреть? У меня тут есть несколько вариантов</code> ⬇□", reply_markup=markup)


    @dp.message_handler(Text(equals="Сегодня \n 🔋"))
    async def get_films_day1(message: types.Message):
        with open('day1.json', encoding='utf-8') as file:
            films_dict = json.load(file)

        for k, v in films_dict.items():
            film = f"<b>{v['film_name']}</b> " \
                   f"<u>{v['film_age']}</u>\n" \
                   f"<code>Рейтинг
IMDB:</code>{v['film_imdb_rating']}\n" \
                   f"<code>Инфо: </code>{v['film_genre_info']}\n" \
                   f"<code>Режиссер:  </code>{v['film_producer']}\n"
\
                   f"<code>Дата                                  премьеры:
</code>{v['film_premiere']}\n" \
                   f"<b><u>Сеансы:\n</u></b>{v['film_sessions']}\n"
\
```

```
                    f"{v['film_url']}"

            await message.answer(film)


    @dp.message_handler(Text(equals="Завтра \n 🔋"))
    async def get_films_day2(message: types.Message):
        with open('day2.json', encoding='utf-8') as file:
            films_dict = json.load(file)

        for k, v in films_dict.items():
            film = f"<b>{v['film_name']}</b> " \
                   f"<u>{v['film_age']}</u>\n" \
                   f"<code>Рейтинг
IMDB:</code>{v['film_imdb_rating']}\n" \
                   f"<code>Инфо: </code>{v['film_genre_info']}\n" \
                   f"<code>Режиссер:  </code>{v['film_producer']}\n"
\
                   f"<code>Дата                              премьеры:
</code>{v['film_premiere']}\n" \
                   f"<b><u>Сеансы:\n</u></b>{v['film_sessions']}\n"
\
                   f"{v['film_url']}"

            await message.answer(film)


    @dp.message_handler(Text(equals="Послезавтра \n 🔋"))
    async def get_films_day3(message: types.Message):
        with open('day3.json', encoding='utf-8') as file:
            films_dict = json.load(file)

        for k, v in films_dict.items():
            film = f"<b>{v['film_name']}</b> " \
                   f"<u>{v['film_age']}</u>\n" \
                   f"<code>Рейтинг
IMDB:</code>{v['film_imdb_rating']}\n" \
                   f"<code>Инфо: </code>{v['film_genre_info']}\n" \
```

```
                    f"<code>Режиссер:  </code>{v['film_producer']}\n"
\
                    f"<code>Дата                                 премьеры:
</code>{v['film_premiere']}\n" \
                    f"<b><u>Сеансы:\n</u></b>{v['film_sessions']}\n"
\
                    f"{v['film_url']}"

            await message.answer(film)


    @dp.message_handler(Text(equals="Через 4 дня"))
    async def get_films_day4(message: types.Message):
        with open('day4.json', encoding='utf-8') as file:
            films_dict = json.load(file)

        for k, v in films_dict.items():
            film = f"<b>{v['film_name']}</b> " \
                   f"<u>{v['film_age']}</u>\n" \
                   f"<code>Рейтинг
IMDB:</code>{v['film_imdb_rating']}\n" \
                   f"<code>Инфо: </code>{v['film_genre_info']}\n" \
                   f"<code>Режиссер:  </code>{v['film_producer']}\n"
\
                   f"<code>Дата                                 премьеры:
</code>{v['film_premiere']}\n" \
                   f"<b><u>Сеансы:\n</u></b>{v['film_sessions']}\n"
\
                   f"{v['film_url']}"

            await message.answer(film)


    @dp.message_handler(Text(equals="Бизяна"))
    async def monkey_name_protocol(message: types.Message):
        return    await    message.reply("□:<code>    Выходи    за
меня💍</code>")
```

```
    @dp.message_handler(Text(equals="Через 5 дней"))
    async def get_films_day5(message: types.Message):
        with open('day5.json', encoding='utf-8') as file:
            films_dict = json.load(file)

        for k, v in films_dict.items():
            film = f"<b>{v['film_name']}</b> " \
                   f"<u>{v['film_age']}</u>\n" \
                   f"<code>Рейтинг
IMDB:</code>{v['film_imdb_rating']}\n" \
                   f"<code>Инфо: </code>{v['film_genre_info']}\n" \
                   f"<code>Режиссер:  </code>{v['film_producer']}\n"
\
                   f"<code>Дата                                премьеры:
</code>{v['film_premiere']}\n" \
                   f"<b><u>Сеансы:\n</u></b>{v['film_sessions']}\n"
\
                   f"{v['film_url']}"

            await message.answer(film)


    @dp.message_handler(Text(equals="Через 6 дней"))
    async def get_films_day6(message: types.Message):
        with open('day6.json', encoding='utf-8') as file:
            films_dict = json.load(file)

        for k, v in films_dict.items():
            film = f"<b>{v['film_name']}</b> " \
                   f"<u>{v['film_age']}</u>\n" \
                   f"<code>Рейтинг
IMDB:</code>{v['film_imdb_rating']}\n" \
                   f"<code>Инфо: </code>{v['film_genre_info']}\n" \
                   f"<code>Режиссер:  </code>{v['film_producer']}\n"
\
                   f"<code>Дата                                премьеры:
</code>{v['film_premiere']}\n" \
                   f"<b><u>Сеансы:\n</u></b>{v['film_sessions']}\n"
\
                   f"{v['film_url']}"
```

```
            await message.answer(film)


    @dp.message_handler(Text(equals="Через 7 дней"))
    async def get_films_day7(message: types.Message):
        with open('day7.json', encoding='utf-8') as file:
            films_dict = json.load(file)

        for k, v in films_dict.items():
            film = f"<b>{v['film_name']}</b> " \
                   f"<u>{v['film_age']}</u>\n" \
                   f"<code>Рейтинг
IMDB:</code>{v['film_imdb_rating']}\n" \
                   f"<code>Инфо: </code>{v['film_genre_info']}\n" \
                   f"<code>Режиссер:  </code>{v['film_producer']}\n"
\
                   f"<code>Дата                                премьеры:
</code>{v['film_premiere']}\n" \
                   f"<b><u>Сеансы:\n</u></b>{v['film_sessions']}\n"
\
                   f"{v['film_url']}"

            await message.answer(film)


    @dp.message_handler(Text(equals="Скоро в кино 🔜"))
    async def get_future_films(message: types.Message):
        with open('future_films.json', encoding='utf-8') as file:
            films_dict = json.load(file)

        for k, v in films_dict.items():
            film = f"<b>{v['film_name']}</b>\n" \
                   f"<code>Инфо: </code>{v['film_genre_info']}\n" \
                   f"<code>Режиссер:  </code>{v['film_producer']}\n"
\
                   f"{v['film_url']}"

            await message.answer(film)
```

```
    @dp.message_handler(Text(equals="Информация о кинотеатрах ℹ"))
    async def get_cinemas_info(message: types.Message):
        with open('kinoteatr_info_lvov.json', encoding='utf-8') as
file:
            films_dict = json.load(file)

        for k, v in films_dict.items():
            film = f"<b>{v['cinema_name']}</b>\n" \
                   f"<code>Адрес:</code>{v['cinema_address']}\n" \
                   f"<code>Телефон: </code>{v['cinema_phone']}\n" \
                   f"<code>Сайт: </code>{v['cinema_url']}"

            await message.answer(film)


    if __name__ == '__main__':
        executor.start_polling(dp)
```

films_spider.py

```
    import scrapy
    import json
    import datetime
    from datetime import timedelta, datetime


    # class Film(scrapy.Item):
    #     film_title = scrapy.Field()
    #     film_genre_info = scrapy.Field()


    class FilmSpider(scrapy.Spider):
        name = 'kinoafisha_spider'
        start_urls = []

        today_day = datetime.now()
        delta_list = [timedelta(0), timedelta(1), timedelta(2),
timedelta(3), timedelta(4), timedelta(5), timedelta(6)]
        for element in delta_list:
```

```
            day = today_day + element
            day = 'https://kinoafisha.ua/ua/kinoafisha/odessa/#' +
day.strftime("%d-%m-%Y") + '/all'
            start_urls.append(day)

        def parse(self, response, **kwargs):
            blocks = response.css('div.movie__list > ul >
.movie__block')
            for block in blocks:
                film = {
                    'film_title':
block.css('.movie__title::text').get(),
                    'film_age': block.css('.agerate::text').get(),
                    'film_genre_info':
block.css('.countries::text').get(),

                }
                yield film
```

ДОДАТОК В – ПРОГРАМА ТА МЕТОДИКА ВИПРОБУВАННЯ ПЗ

1 Об’єкт випробувань

Об’єктом випробувань є програмне забезпечення телеграм-боту кіноафіши міста Миколаєва.

Область застосування ПЗ – інформування користувачів стосовно кінопоказів міста.

2 Мета випробувань

Перевірка відповідності характеристик та вимог розробленої програми, викладених у технічному завданні, перевірка працездатності даного програмного забезпечення, приведення прикладу роботи та отримання відповідних навичок.

3 Вимоги до додатка

При проведенні випробувань, можливості програми підлягають перевірці на відповідність вимогам, викладених у пункті «Вимоги до функціональних характеристик» технічного завдання.

4 Вимоги до програмної документації

Програмна документація повинна включати в себе наступні документи:

– технічне завдання;

– текст програми;

– програма і методика випробувань;

– опис програми;

– інструкція користувача.

5 Склад та порядок випробувань

Вимоги до складу та параметрів технічних і програмних засобів вказані в Додатку А.

Порядок проведення випробувань:

– виконуються контрольні тести в довільному порядку;

– робиться аналіз отриманих результатів і встановлюється відповідність програмного продукту вимогам і системним документам.

При виявлені помилок у роботі системи складається їх перелік і обговорюється термін їх виправлення розробником. Після цього замовник проводить повторне тестування в повному обсязі (можливе використання нових чи додаткових тестів).

6 Методи випробувань

Методом випробування розробленого програмного забезпечення є тестування чорної скриньки, яке представляє процес виконання програми з метою виявлення помилок. Кроки процесу задаються тестами.

У таблиці В.1, що наведена нижче, вказано перелік випробувань.

Таблиця В.1 – Методика випробування

№	Дія	Очікуваний результат	Результат перевірки	Зауваження
1	Перевірка правильності введення запиту при некоректному текстовому вводі	Програма повинна перевірити коректність запиту	Виконано	

Продовження таблиці В.1

№	Дія	Очікуваний результат	Результат перевірки	Зауваження
2	Перевірка правильності введення запиту при відправці фото	Програма повинна перевірити коректність запиту	Виконано	
3	Перевірка правильності введення запиту при відправці аудіофайлу	Програма повинна перевірити коректність запиту	Виконано	
4	Перевірка правильності введення запиту при відправці відеофайлу	Програма повинна перевірити коректність запиту	Виконано	
5	Перевірка правильності введення запиту при відправці стікеру	Програма повинна перевірити коректність запиту	Виконано	
6	Перевірка правильності введення запиту при коректному вводі	Програма повинна перевірити коректність запиту	Виконано	

В процесі випробування була перевірена функціональність ПЗ.

ДОДАТОК Г – ОПИС ПРОГРАМИ

1Загальні відомості

1.1Позначення і найменування програми

Найменування програмного забезпечення – «Телеграм-бот кіноафіши м. Миколаєва».

Позначення: «Telegram-bot of the movie posters of Mykolaiv»

1.2Програмне забезпечення, необхідне для функціонування програми

Комп’ютер повинен мати наступну мінімальну комплектацію:

- CPU: Intel(R) Celeron(R) 2.16 МГц;
- RAM: 2 GB;
- HDD: 1 ГБ вільного місця.

Мінімальна комплектація смартфону:

- Версія Android 8.0 / iOS 12.4.9;
- ОЗУ 3 ГБ;
- 1 ГБ вільної пам’яті.

При використанні на смартфоні або планшетному ПК, повинен бути встановлений додаток Telegram.

1.3Мови програмування

Програмний продукт «Telegram-bot of the movie posters of Mykolaiv» розроблено мовою програмування Python. Для взаємодії з Telegram Bot API було обрану бібліотеку AIOGram.

2Функціональне призначення

2.1Класи розв’язуваних завдань та призначення програми

Функціональним призначенням програми є автоматизація інформування мешканців міста стосовно сеансів у кінотеатрах Миколаєва та покращення зручності надання інформації користувачу за допомогою месенджеру Telegram.

2.2Функціональні обмеження

Для роботи з телеграм-ботом необхідно мати підключення до мережі Інтернет та встановлений додаток Telegram при використанні на смартфоні або планшетному ПК.

3Опис логічної структури

3.1Структура програми

Програмне забезпечення складається з наступних частин:

– Сервер, на якому запущений скрипт програми;

– Сам сервер Телеграму, з яким спілкується телеграм-бот, завдяки Telegram Bot API;

– Пристрій користувача, із встановленим додатком Telegram та підключенням до мережі Інтернет.

Графічне зображення всіх компонентів можна побачити на рисунку Г.1.

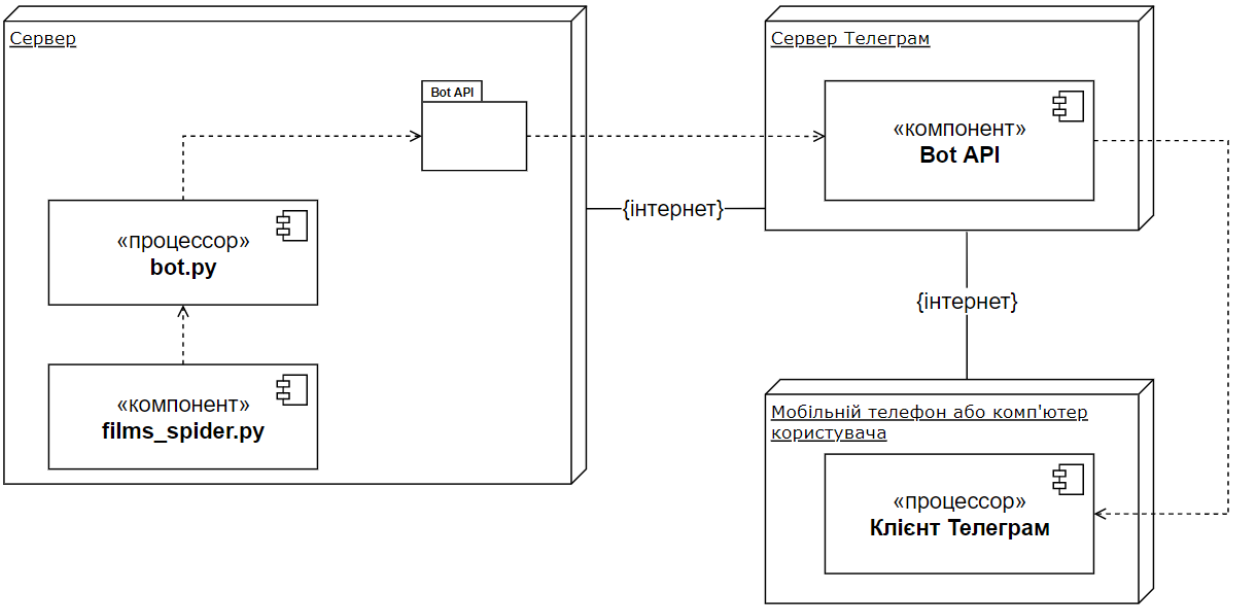

Рисунок Г.1 – Схема взаємодії

3.2Алгоритм програми

Програмне забезпечення телеграм-боту кіноафіши має технологію «клієнт-сервер». Клієнтом виступає додаток Telegram, а сервером – комп’ютер із запущеним скриптом.

Робота ПЗ представлена взаємодією між клієнтською та серверною частинами. Алгоритм роботи програми є наступним:

1) Користувач взаємодіє з ботом через додаток Telegram;

2) Telegram через Інтернет відсилає запити серверу;

3) Сервер обробляє запит;

4) Скрипт збирає інформацію та відсилає результат;

5) Telegram відображає отриманий результат у вигляді повідомлень від телеграм-боту.

3.3Використовувані методи

Програмне забезпечення телеграм-боту кіноафіши м. Миколаєва реалізовано на мові Python3. В якості середовища розробки програм повинна бути використана PyCharm.

ДОДАТОК Д – ІНСТРУКЦІЯ КОРИСТУВАЧА

Після натискання кнопки «Старт» користувачем, бот виведе вітальне повідомлення, стислу інструкцію з користування та клавіатуру бота через команди якої можна отримати інформацію стосовно показів на неділю, майбутніх прем’єр і контактну інформацію кінотеатрів міста. Зображено на рисунку Д.1.

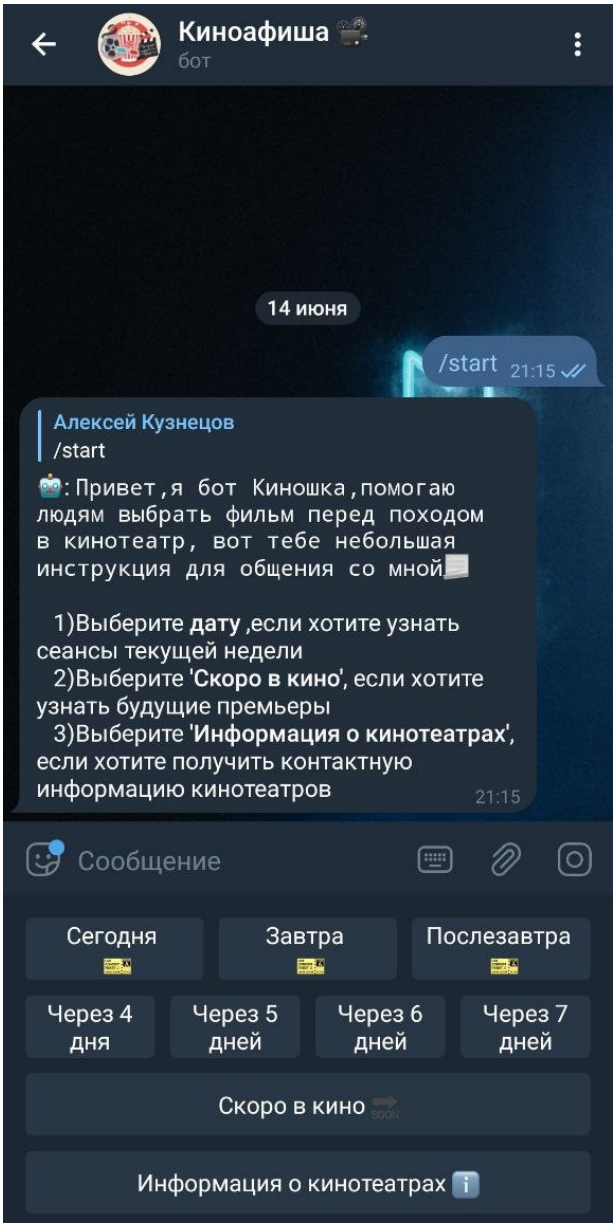

Рисунок Д.1 – Старт роботи з ботом.

Для того щоб дізнатися сеанси фільмів на тиждень, необхідно натиснути на кнопку бажаного дня тижня. Після цього бот надасть інформацію стосовно показів на обраний день. Зображено на рисунках Д.2, Д.3 та Д.4.

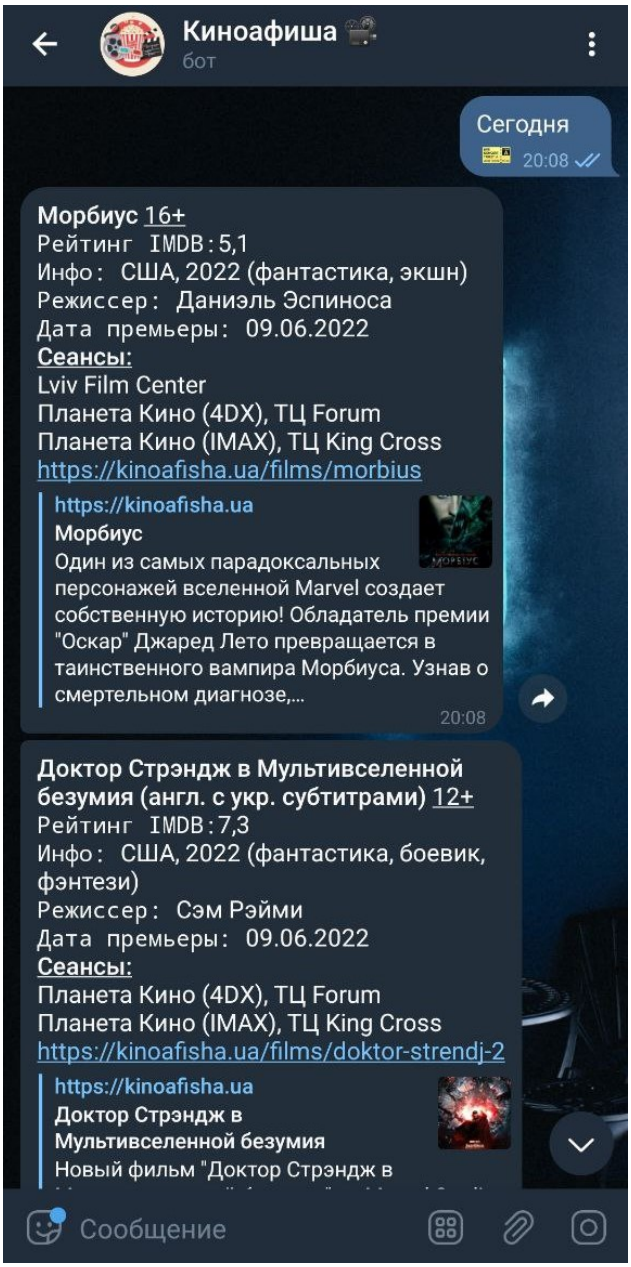

Рисунок Д.2 – Надання інформації стосовно показів на тиждень

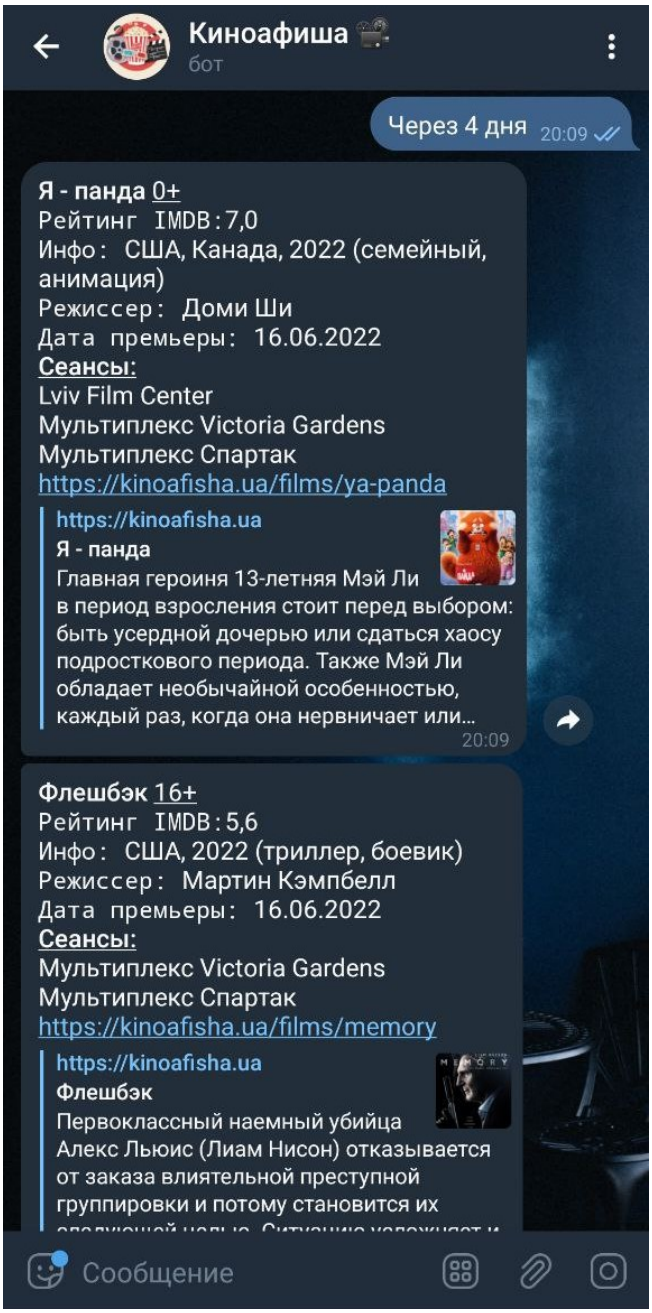

Рисунок Д.3 – Надання інформації стосовно показів на тиждень

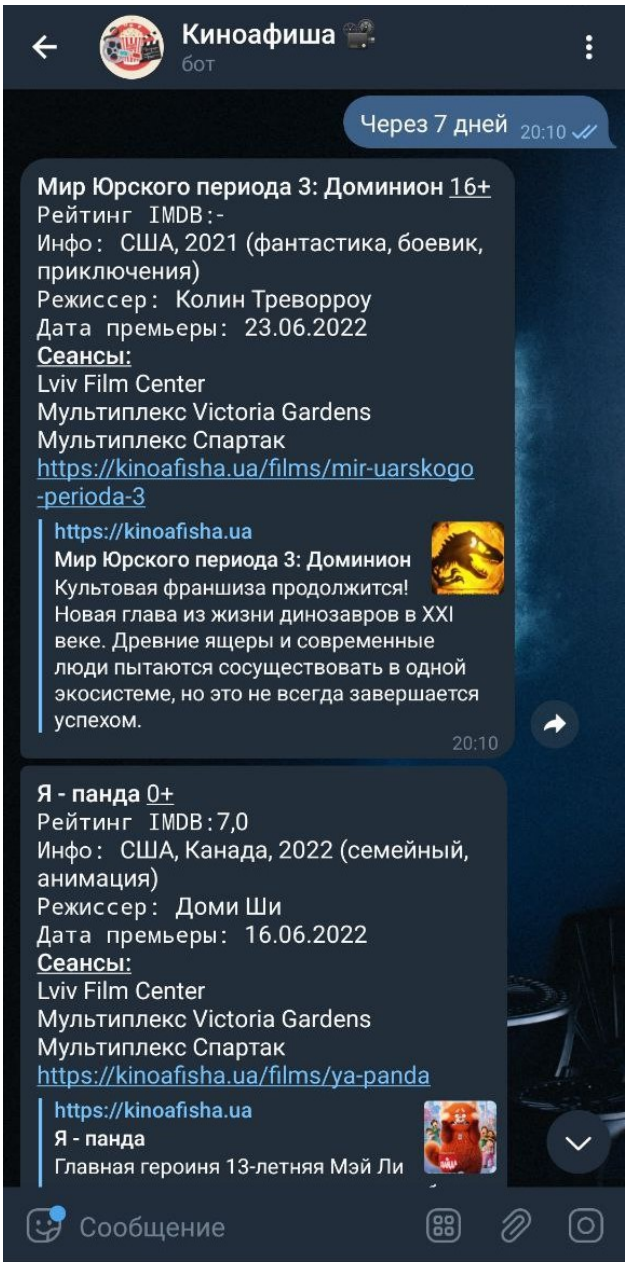

Рисунок Д.4 – Надання інформації стосовно показів на тиждень

Після того як користувач обрав бажаний фільм, він може побачити перелік кінотеатрів де проходять покази у графі «Сеансы». Зображено на рисунку Д.5.

Рисунок Д.5 – Графа «Сеансы»

Далі користувач може натиснути кнопку «Информация о кинотеатрах», знайти кінотеатр де він хоче подивитися обраний фільм та перейти на сайт для замовлення квитків або зателефонувати на гарячу лінію. Зображено на рисунках Д.6 та Д.7.

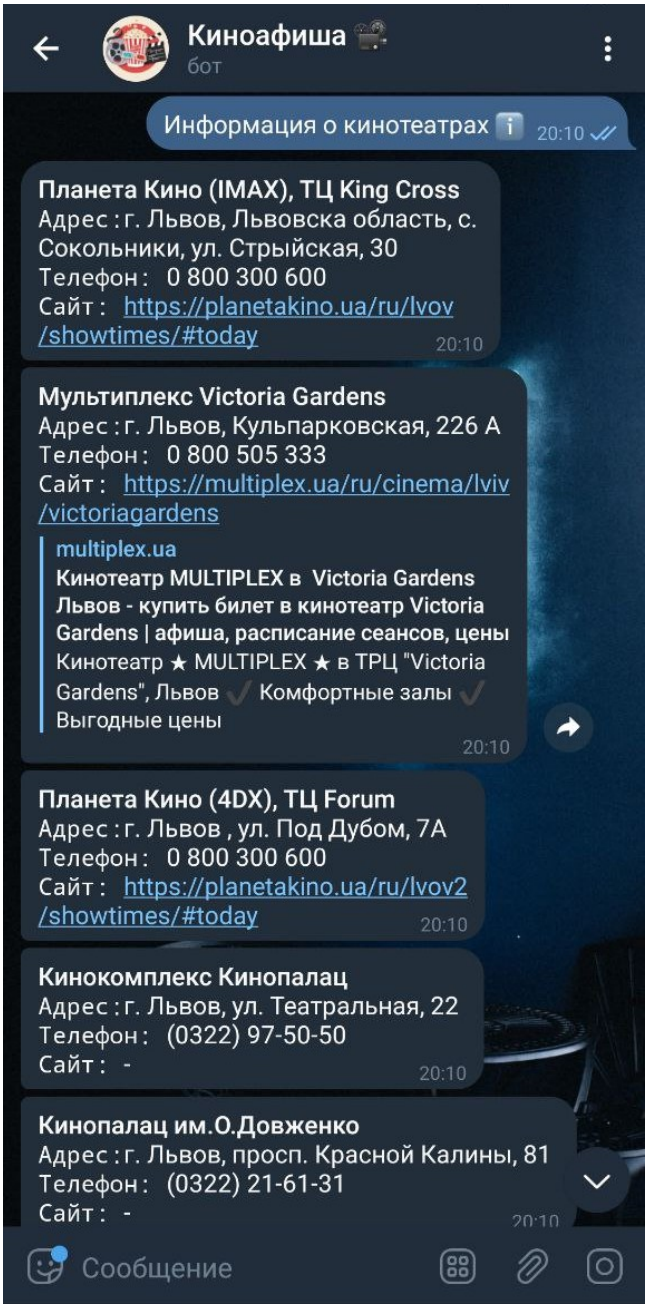

Рисунок Д.6 – Інформація про кінотеатри

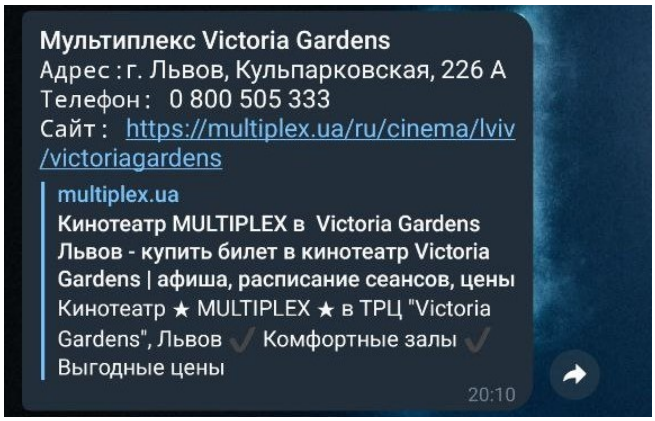

Рисунок Д.7 – Інформація про кінотеатр

Також, якщо користувач бажає дізнатися які заплановано прем’єри у майбутньому в кінотеатрах міста, він може натиснути кнопку «Скоро в кино». Зображено на рисунку Д.8.

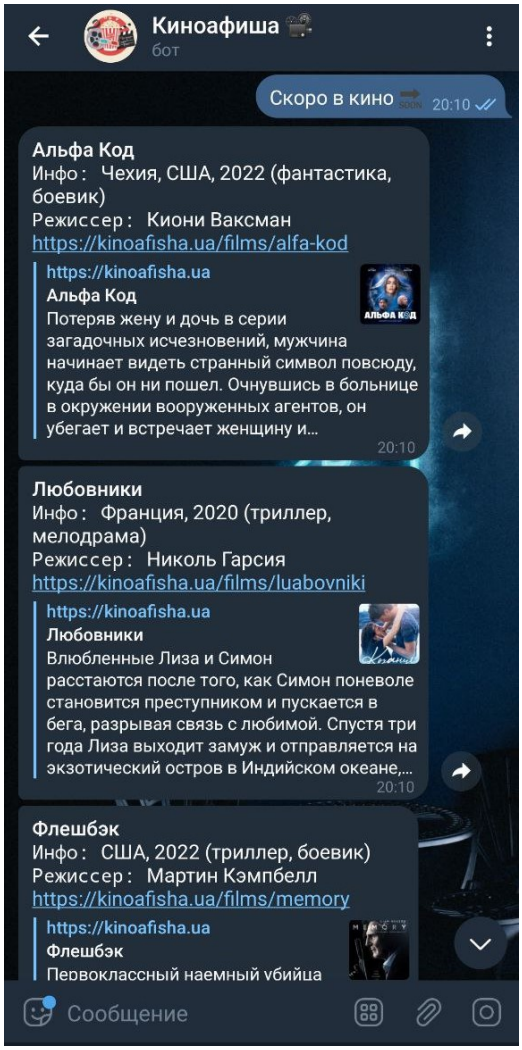

Рисунок Д.8 – Перегляд майбутніх прем’єр