МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ПРИРОДОКОРИСТУВАННЯ

ФАКУЛЬТЕТ МЕХАНІКИ, ЕНЕРГЕТИКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

першого (бакалаврського) рівня вищої освіти

на тему: **«Розробка системи планування та організації робочого часу для студентів та викладачів Львівського національного університету природокористування на основі Телеграм-боту»**

Виконав: здобувач освіти групи Кн-41сп

Спеціальності 122 «Комп'ютерні науки»

(шифр і назва)

Батрон Олег Орестович

(Прізвище та ініціали)

Керівник: к.е.н. Станько В.Ю.

(Прізвище та ініціали)

Рецензент: к.т.н. Березовецький С.А.

(Прізвище та ініціали)

ДУБЛЯНИ-2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ПРИРОДОКОРИСТУВАННЯ
ФАКУЛЬТЕТ МЕХАНІКИ, ЕНЕРГЕТИКИ
ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Перший (бакалаврський) рівень вищої освіти
Спеціальність 122 «Комп'ютерні науки»

"ЗАТВЕРДЖУЮ"
Завідувач кафедри

(підпис)

д.т.н., професор, Тригуба А.М.
"____"_____ 202_ р.

ЗАВДАННЯ
на кваліфікаційну роботу

Батрон Олег Орестович

1. Тема роботи: *«Розробка системи планування та організації робочого часу для студентів та викладачів Львівського національного університету природокористування на основі Телеграм-боту»*
Керівник роботи *к.е.н. Станько В.Ю.*
Затверджені наказом по університету *від 27.11.2023 року № 641/к-с*

2. Строк подання студентом роботи *10.06.2024 р.*

3. Вихідні дані до роботи: *характеристика сучасних інформаційних систем, вимоги до побудови інформаційних систем, науково-технічна і довідкова література, засоби створення та мови програмування, розклад занять.*

4. Зміст розрахунково-пояснювальної записки: (перелік питань, які потрібно розробити) *Вступ*
1. Аналіз предметної області
2. Постановка задачі
3. Розробка інформаційної системи
4. Охорона праці та безпека в надзвичайних ситуаціях
Висновки та пропозиції
Бібліографічний список
Додатки

5. Перелік графічного матеріалу: *Графічний матеріал подається у вигляді презентації*

6. Консультанти з розділів:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1, 2, 3	*Станько В.Ю., в.о. доцента кафедри інформаційних систем та технологій*		
5	*Городецький І.М., доцент кафедри фізики, інженерної механіки та безпеки виробництва*		

7. Дата видачі завдання *30.11.2023 р.*

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту	Строк виконання етапів роботи	Відмітка про виконання
1.	*Написання першого розділу та означення головних завдань роботи*	*30.11.2023 – 03.01.2024*	
2.	*Виконання другого розділу та формування проектних рішень*	*04.01.2024 – 28.02.2024*	
3.	*Виконання третього розділу, розрахунків та розробка програмної частини*	*01.03.2024 – 30.03.2024*	
4.	*Написання розділу: «Охорона праці та безпека в надзвичайних ситуаціях»*	*01.04.2024 – 30.04.2024*	
5.	*Завершення оформлення пояснювальної записки та презентаційного матеріалу, апробація результатів роботи*	*01.05.2024 – 30.05.2024*	
6.	*Завершення роботи в цілому. Підготовка до захисту кваліфікаційної роботи*	*01.06.2024 – 10.06.2024*	

Здобувач _____ Батрон О.О.
(підпис)

Керівник роботи _____ Станько В.Ю.
(підпис)

УДК 004.42:[331.31:378.4(477.83)

Розробка системи планування та організації робочого часу для студентів та викладачів Львівського національного університету природокористування на основі Телеграм-боту.

Батрон О. О. Кафедра інформаційних технологій – Дубляни, Львівський НУП, 2024.

Кваліфікаційна робота: 52 с. текстової частини, 14 рисунків, 15 джерел.

Було досліджено ринок месенджерів та проведено порівняння між простотою розробки для кожної служби та рівнем підтримки чат-ботів. Виявлено, що месенджер Telegram надає найкращу інтеграцію чат-ботів та найбільш інформативну документацію для розробників.

У першому розділі проводиться аналіз предметної області. Визначаються поняття та особливості Telegram-ботів, здійснюється огляд програмних сервісів та інструментів, необхідних для реалізації поставленої задачі. Також аналізуються існуючі рішення та обґрунтовуються напрямки вирішення поставленого завдання.

Другий розділ присвячений постановці задачі. Обґрунтовується вибір напряму досліджень, описуються методи вирішення поставлених задач, розробляється загальна методика проведення власних досліджень, здійснюється стислий аналіз об'єкту дослідження.

У третьому розділі описується розробка інформаційної системи. Розробляється чат-бот на основі Telegram API, розробляється користувацький інтерфейс, оптимізуються запити до Google Sheets, проводиться тестування та оцінка чат-боту, аналізуються отримані результати.

Окремий розділ присвячений питанням охорони праці. Проаналізовано стан виробничої санітарії і гігієни праці, обґрунтовано організаційно-технічні рекомендації з охорони праці, розглянуто питання пожежної безпеки.

У висновках сформульовані основні результати проведеного дослідження та надано пропозиції щодо подальшого вдосконалення системи.

Ключові слова: чат-бот, Telegram, організація робочого часу, студенти, викладачі, Google Sheets API.

ЗМІСТ

ВСТУП

На сьогоднішній день комп’ютерно-інформаційні технології є важливою частиною людського життя. За допомогою них можна автоматизувати механізми, що надають швидкий доступ до інформації та збільшують ефективність.

Однією з найпоширеніших сфер у автоматизації доступу до інформації є створення чат-ботів, які дозволяють спростити комунікацію з користувачем, ефективно опрацьовуючи дані.

Розробка telegram-бота містить в собі низку складних кроків, які вимагають структурованих та системних поглядів до управління проектом. Вдале створення проекту безпосередньо залежить від вірного планування, балансування ресурсів, управління ризиками та вміння надати високу якість користувачу.

Для організації та планування навчального процесу у Львівському національному університеті природокористування впроваджується сучасна багатофункціональна інформаційна система. Однією із складових цієї системи є модуль «Розклад».

Розклад занять на перший семестр 2023-2024 року складається з таблиці розміром більше 500 стовпців та більше 100 рядків. Це утворює понад 50 тисяч клітинок з інформацією.

Шукати інформацію на невеликому екрані телефона чи планшета незручно, особливо , коли вона подана у таблиці з надзвичайно великою кількістю клітинок. Таке відображення інформації некомфортне і призводить до негативного досвіду користувача.

Для оптимізації пошуку та відображення потрібної інформації пропонуємо розробити telegram-бота, який буде виводити інформацію за запитом користувача і формувати її у зручному вигляді.

Головними цілями є забезпечення зручності для студентів і викладачів університету та підвищення рівня автоматизації в університетському

середовищі. Наша мета – створити ефективний та корисний telegram-бот, який спростить доступ до розкладу занять та зробить навчальний процес більш організованим та ефективним для всієї університетської спільноти.

Таким чином, розвиток даної системи дає великі можливості для покращення організації робочого часу студентів та викладачів, забезпечуючи зручність, швидкість та ефективність в управлінні навчальним процесом.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Визначення telegram-ботів

Сьогодні темп життя постійно зростає, тому ефективне управління часом стає надзвичайно важливим як для студентів, так і для викладачів. Стрімкий прогрес у інформаційних технологіях дає нові можливості для автоматизації та оптимізації організації робочого часу. Telegram-бот, інтегрований з GoogleSheetsAPI, може стати потужним інструментом для вирішення цієї задачі.

Чат-боти, також відомі як роботи, – це програмні агенти, які автоматизують різні завдання, зазвичай виконуючи їх в автоматичному режимі, без прямого втручання людини. Вони можуть бути розроблені для різноманітних цілей, як-от взаємодія з користувачами на вебсайтах або в месенджерах, автоматизація рутинних завдань, моніторинг, аналітика, розподіл ресурсів та багато іншого [1].

Telegram-боти – це спеціальні програми, які працюють всередині месенджера Telegram. Вони взаємодіють з користувачем через текстові повідомлення, кнопки, команди та інші інтерфейсні елементи, які надає Telegram.

Боти можуть самі шукати потрібну інформацію, генерувати контент, виконувати команди, тобто забезпечувати виконання завдань без безпосередньої участі людини. Кожен бот має свою унікальну функціональність та алгоритм створений розробником.

Telegram підтримує великі обсяги повідомлень та користувачів, що дозволяє ботам обслуговувати велику аудиторію користувачів.

Важливою перевагою telegram-ботів є те, що вони дозволяють забезпечувати неперервне обслуговування користувачів. Тобто боти можуть працювати цілодобово, без перерв, і навіть оновлення алгоритмів може відбуватися в реальному часі.

Таким чином, telegram-боти є потужним інструментом автоматизації різноманітних процесів у широкому колі сфер діяльності, включаючи організацію робочого часу, планування завдань та обробку даних.

1.2 Огляд програмних сервісів та інструментів

Для розробки Telegram-ботів існує широкий спектр різноманітних фреймворків, бібліотек та інструментів, які надають розробникам необхідні можливості для створення, налаштування та розгортання ботів.

Для створення Telegram-ботів доступний широкий вибір мов програмування. Найбільш популярними і широко використовуваними є Python, JavaScript, Ruby, PHP та Go. Ці мови мають велику кількість бібліотек та SDK, спеціально розроблених для взаємодії з Telegram API, що значно спрощує і прискорює процес розробки ботів.

Перевагою мови програмування Python над іншим є велика кількість зручних та гнучких фреймворків для створення telegram-ботів та взаємодії з TelegramAPI. Серед них можна виділити:

— python-telegram-bot - один з найпопулярніших фреймворків, що надає простий і зручний інтерфейс для взаємодії з Telegram API.

— aiogram - асинхронний фреймворк, який забезпечує ефективну обробку великої кількості запитів від користувачів.

— pyTelegramBotAPI - бібліотека, що дозволяє легко інтегрувати Telegram Bot API в Python-додатки.

Існує багато конструкторів для створення telegram-ботів, які дозволяють створити власного бота практично без будь-яких навичок. Наприклад, PuzzleBot, gerabot, FlowXO, BotPress та інші. Проте у такого підходу є кілька недоліків, такі як неможливість вбудувати в бот сторонні сервіси, відсутність гнучкості в розробці та відносно висока ціна за використання.

Окрім мов програмування та фреймворків важливим інструментом у розробці telegram-ботів, які взаємодіють з таблицями є GoogleSheetsAPI. Цей API дозволяє інтегрувати Google Sheets у програмні рішення, надаючи можливість ефективно зберігати, обробляти та передавати дані між ботом та електронними таблицями. Це дає змогу реалізувати такі функції, як планування, відстеження завдань, управління розкладом, тощо.

Також для створення та налагодження ботів використовуються різні інструменти та сервіси, наприклад BotFather. Цей сервіс дозволяє отримати унікальний ключ, завдяки якому можна буде звертатись до TelegramAPI. Крім того, він дає можливість наповнення telegram-бота.

Отже, розробка Telegram-ботів передбачає використання широкого спектру мов програмування, фреймворків, API та допоміжних інструментів. Це дозволяє створювати різноманітні додатки з унікальними функціональними можливостями.

1.3 Аналіз існуючих рішень

Для того щоб зрозуміти, які функції найбільш важливі для користувачів, необхідно проаналізувати існуючі рішення. Багато ботів пропонують базовий функціонал, такий як виведення інформації та пошук даних. Проте не всі вони інтегруються з Google Sheets, що є важливим для зберігання та аналізу даних.

При огляді існуючих систем управління робочим часом можна виділити кілька ключових рішень, які використовуються студентами та викладачами:

— Календарні додатки, такі як Google Calendar, Microsoft Outlook, тощо. Серед переваг виділяються інтуїтивно зрозумілий інтерфейс, синхронізація між пристроями, можливість створювати нагадування, спільний доступ. Проте є ряд недоліків, серед них: обмежена функціональність для повноцінного управління робочим часом,

відсутність автоматизації та інтерактивності, складність інтеграції з іншими системами.

— Спеціалізовані додатки для тайм-менеджменту, такі як Trello, Todoist, тощо. Перевагами таких додатків є розширені функції для планування, постановки завдань, відстеження прогресу, спільної роботи. Серед недоліків можна виділити складність налаштування, необхідність використання окремого додатку, обмежена взаємодія з іншими сервісами.

— Інтеграція з Google Sheets або Microsoft Excel. Таке рішення є гнучким, з можливістю створювати власні шаблони та можливістю інтеграції з іншими офісними додатками. Але з іншого боку, недоліком є необхідність ручного введення даних, складність налаштування формул та автоматизації.

Аналізуючи ці рішення, можна зробити висновок, що жодне з них не задовольняє повністю потреби студентів та викладачів у зручному, інтегрованому та доступному інструменті для управління робочим часом. Існуючі додатки або занадто складні, або обмежені у функціональності. Тому є потреба у розробці нового рішення, яке б поєднувало переваги таких додатків, взаємодію з TelegramAPI та інтеграцію з Google Sheets, при цьому залишаючись простим і доступним для користувачів.

Серед існуючих рішень на основі TelegramAPI можна виділити:

— Telegram-бот «Rozklad» від команди Zavod. Цей бот дає можливість отримати розклад занять для студентів багатьох університетів України(рис. 1.1). Перевагами цього бота є його масштабованість. Проте серед недоліків слід виділити складність в користуванні, з точки зору інтерфейсу та відсутність розкладу занять викладачів.

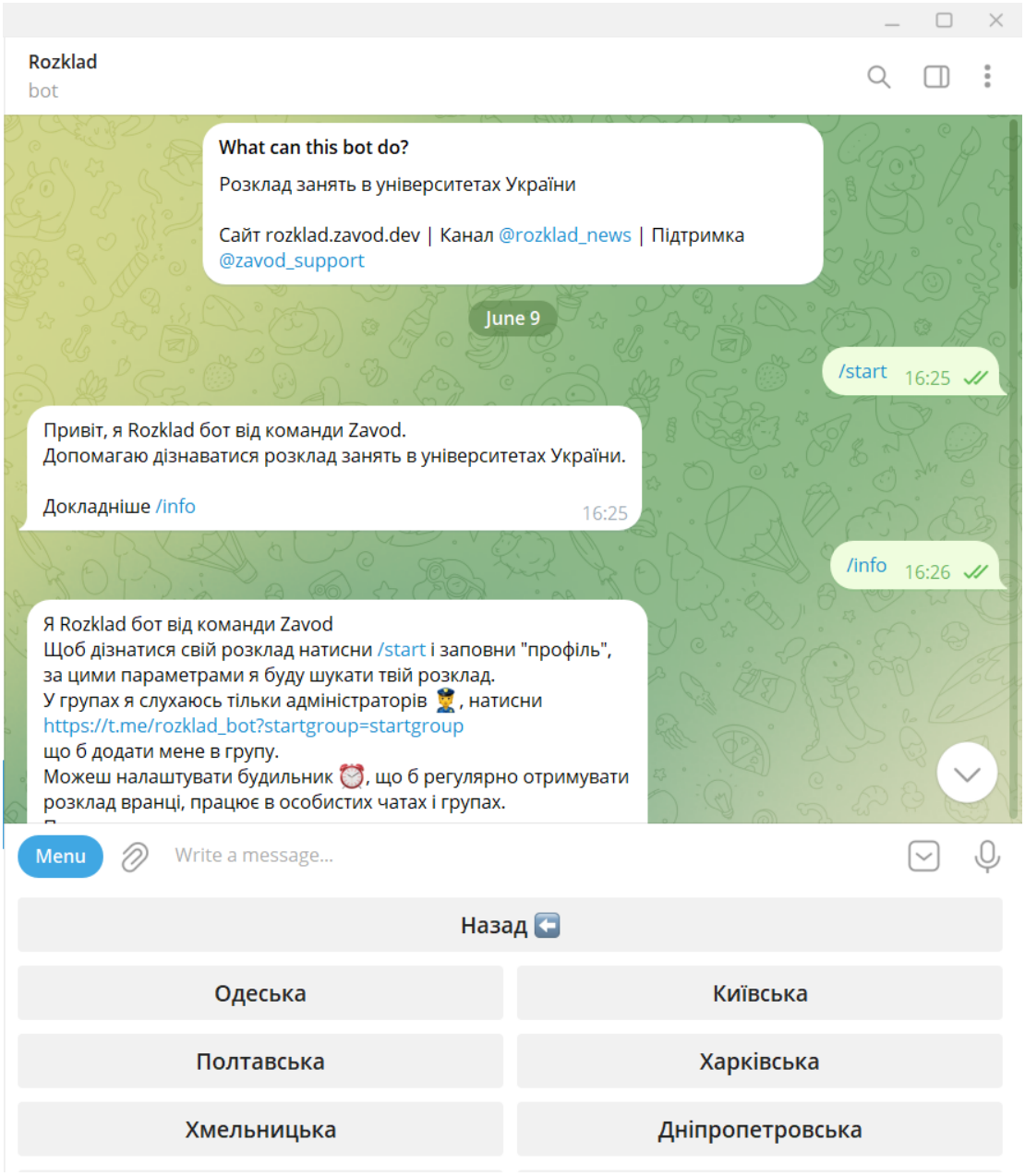


Рисунок 1.1 - Telegram-бот «Rozklad» від команди Zavod

— Telegram-бот «Розклад ДТЕУ». Бот дозволяє отримати розклад занять для студентів державного торговельно-економічного університету(рис. 1.2). Серед переваг можна виділити нагадування про заняття у вигляді сповіщень. Але до недоліків можна віднести складний інтерфейс користувача, тобто користувачу потрібно пройти через багато кроків, щоб отримати інформацію, та відсутність розкладу для викладачів.

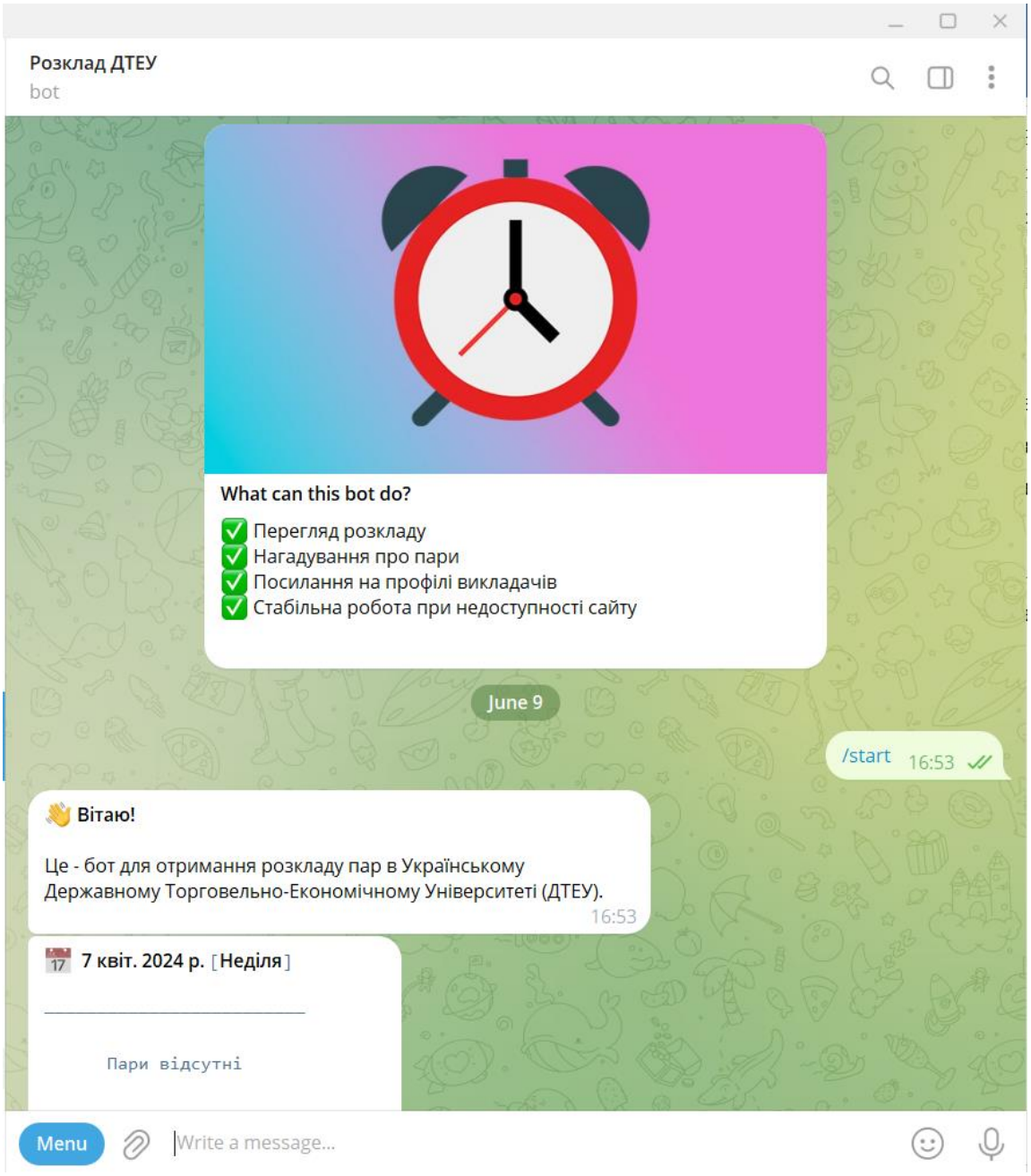


Рисунок 1.2 – Telegram-бот «Розклад ДТЕУ»

— Telegram-бот «Розклад ОНТУ». За допомогою бота можна отримати розклад занять для студентів одеського національного технологічного університету(рис. 1.3). Перевагами цього бота є відкритий код. Але можна помітити велику кількість недоліків, такі як незручний інтерфейс, немає можливості подивитись розклад на конкретний день, немає розкладу занять викладача, та необхідність кожного разу вводити інформацію.

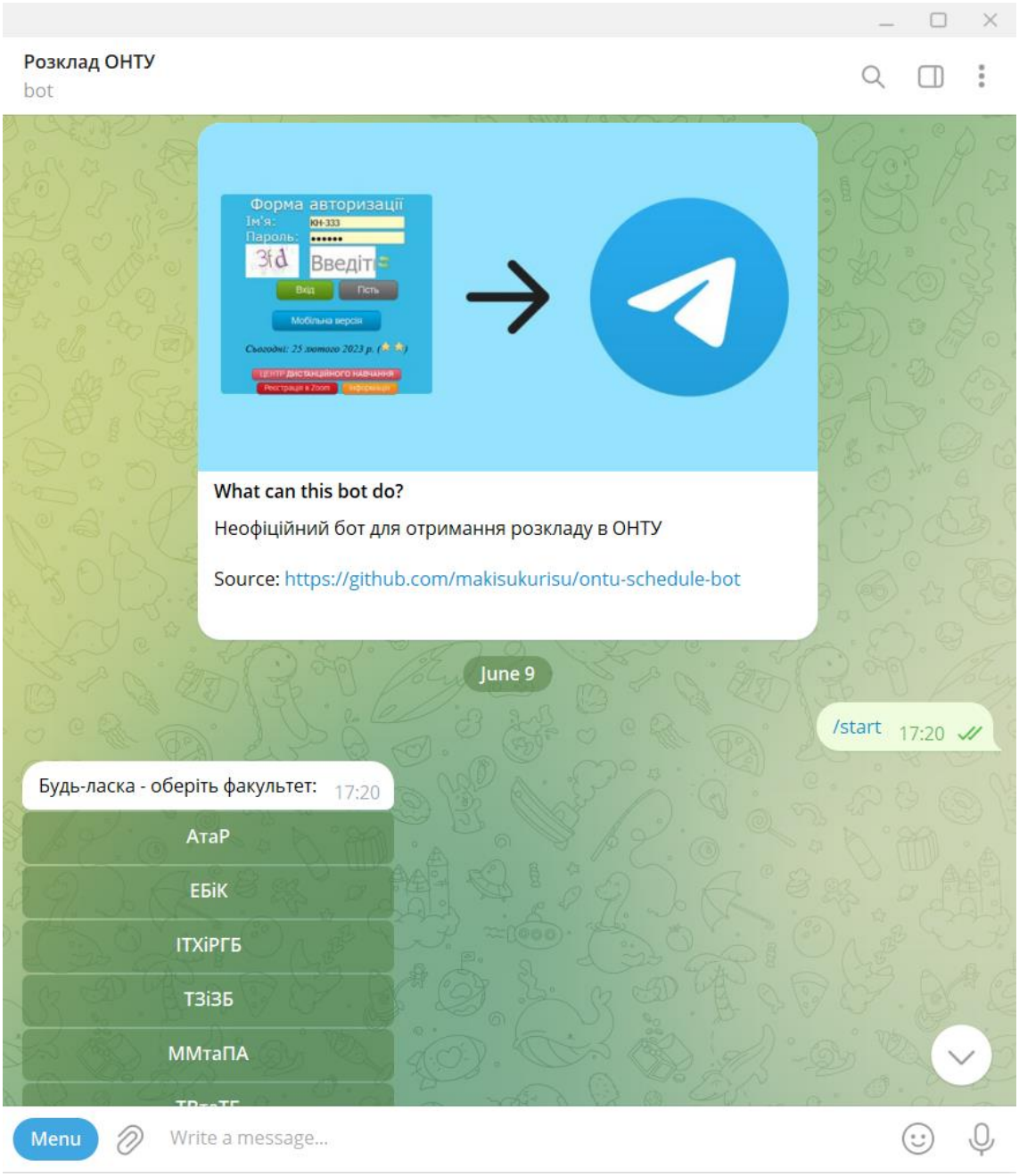


Рисунок 1.3 – Telegram-бот «Розклад ОНТУ»

Таким чином, проаналізувавши існуючі рішення на основі TelegramAPI, можна зробити висновок, що жодне з них не може задовольнити користувача в повній мірі. Найпоширенішими недоліками виявилися складність інтерфейсу та відсутність можливості пошуку розкладу занять викладачів. Тому залишається потреба в розробці нового рішення для студентів та викладачів, яке буде відповідати поставленим вимогам та вирішить вище вказані недоліки.

1.4 Вибір напрямків вирішення поставленого завдання

При виборі технологій та інструментів для розробки системи керування Telegram-ботом для організації робочого часу студентів та викладачів, було проведено ретельний аналіз доступних варіантів. Враховуючи вимоги до функціональності та продуктивності системи, а також необхідність оптимізації запитів до Google Sheets API, було прийняте рішення використовувати мову програмування Python та бібліотеку aiogram.

Python – це інтерпретована мова програмування, вихідний код якої, частинами перетворюється в машинний у процесі виконання спеціальної програми – інтерпретатора. Python характеризується конкретним логічним синтаксисом. Читати код на цій мові програмуванні достатньо легко, так як в ньому мало допоміжних елементів, а правила мови вимагають від програмістів робити відступи [2].

Python є однією з найпопулярніших та найпотужніших мов програмування в сучасному світі програмування. Вона широко використовується для вирішення найрізноманітніших задач, від простих скриптів до складних систем штучного інтелекту. Python характеризується простим та лаконічним синтаксисом, що робить його доступним для вивчення навіть початківцям. Водночас, мова має великий набір бібліотек і фреймворків, які значно спрощують та прискорюють розробку, надаючи готові рішення для широкого кола завдань.

У контексті розробки Telegram-ботів, Python є одним з найкращих варіантів завдяки своїй універсальності та наявності потужних бібліотек для взаємодії з месенджерами. Одна з таких бібліотек - aiogram - є потужним інструментом для розробників Telegram-ботів на Python.

Aiogram надає зручний та інтуїтивно зрозумілий API для взаємодії з Telegram Bot API. Ця бібліотека підтримує широкий спектр можливостей Telegram ботів, від простої обробки команд та повідомлень до реалізації складного inline-режиму та callback-функцій. Aiogram значно спрощує процес

розробки, дозволяючи зосередитись на бізнес-логіці та функціональності бота, а не на технічних деталях взаємодії з Telegram.

Для інтеграції з Google Sheets API, необхідної для реалізації оптимізації запитів до GoogleSheetsAPI, в Python існує бібліотека gspread. Вона дозволяє легко та ефективно працювати з Google Sheets, виконуючи операції з даними, такі як читання, запис, оновлення та форматування. Gspread абстрагує низькорівневі деталі взаємодії з Google Sheets API, надаючи розробнику зручний та інтуїтивно зрозумілий інтерфейс для роботи з даними в Google Таблицях.

Обрані технології та інструменти є оптимальним вибором для вирішення поставленого у дипломній роботі завдання. Python забезпечує швидку та ефективну розробку, aiogram надає зручний інструментарій для створення Telegram-ботів, а gspread дозволяє легко поєднати систему з GoogleSheetsAPI. Ці рішення дають можливість реалізувати весь необхідний функціонал системи керування Telegram-ботом для організації робочого часу студентів та викладачів з оптимізацією запитів до GoogleSheetsAPI.

Використання Python, aiogram та gspread дозволить розробити надійну, масштабовану та легко підтримувану систему. Вона буде відповідати вимогам поставленого завдання. Ці технології є перевіреними, широко застосовуваними в індустрії, та мають потужну екосистему, що забезпечує доступність великої кількості навчальних ресурсів, готових рішень та підтримки спільноти.

Після вибору інструментарію для розробки потрібно створити архітектуру майбутньої системи. Вона повинна передбачати всі потреби користувача та швидкодію.

Основу системи становитиме Telegram-бот, розроблений на основі фреймворку aiogram, який дозволяє створювати ефективні та розширювані Telegram-боти на мові Python. Бот буде відповідати за взаємодію з користувачами, обробку їхніх запитів та інтеграцію з іншими компонентами системи.

Для збереження та управління даними про розклад, завдання, зустрічі тощо буде використано Google Sheets API. Дані зберігатимуться у структурованому

вигляді у Google Sheets, що забезпечить легку інтеграцію, управління та обмін інформацією між Telegram-ботом та Google Sheets.

Для оптимізації запитів до Google Sheets API та покращення продуктивності системи, буде розроблено спеціальний модуль, який дозволить зберігати часто використовувану інформацію в вигляді потоку байтів, за допомогою процесу серіалізації. Це дозволить зменшити кількість звернень до Google Sheets API та пришвидшити отримання даних користувачем.

Крім того, для забезпечення надійності та відмовостійкості системи, буде впроваджено механізми обробки помилок, логування та моніторингу. Це дозволить швидко виявляти та виправляти проблеми, які можуть виникати в процесі експлуатації.

Загалом, запропонована архітектура системи поєднує Telegram-бот, інтеграцію з Google Sheets API та оптимізацію продуктивності за допомогою серіалізації, що дозволить створити ефективну та гнучку систему для організації робочого часу студентів та викладачів.

Після визначення архітектури запропонованого рішення, можна виділити основні етапи реалізації проекту:

Першим етапом буде розробка Telegram-боту. На цьому етапі буде здійснено розробку основного компоненту системи - Telegram-боту. Використовуючи бібліотеку aiogram, буде створено бота, який зможе взаємодіяти з користувачами, обробляти їхні команди та запити. Для цього необхідно:

— Визначити структуру та функціональність бота, які відповідатимуть вимогам системи
— Розробити інтерфейс бота, включаючи меню, кнопки, повідомлення тощо
— Реалізувати алгоритми обробки команд та запитів користувачів

Другим етапом стане інтеграція з Google Sheets API. На цьому етапі буде реалізовано інтеграцію Telegram-боту з Google Sheets API для збереження та управління даними. Основними кроками будуть:

— Автентифікація та отримання доступу до Google Sheets API

— Проектування структури Google Sheets для зберігання даних про розклад, завдання, зустрічі тощо
— Розробка функціоналу для взаємодії з Google Sheets API, включаючи операції читання, запису, оновлення та видалення даних
— Налагодження надійного обміну даними між Telegram-ботом та Google Sheets

На третьому етапі потрібно реалізувати оптимізацію запитів до Google Sheets API. Для підвищення продуктивності системи, на цьому етапі буде розроблено модуль серіалізації, який дозволить зменшити кількість запитів до Google Sheets API. Основними кроками будуть:

— Аналіз частоти та типів запитів до Google Sheets API
— Розробка ефективних алгоритмів, які враховуватимуть специфіку предметної області
— Інтеграція модуля серіалізації в загальну архітектуру системи
— Тестування та налаштування модуля серіалізації для досягнення оптимальної продуктивності

Останнім етапом буде тестування та налагодження системи. На цьому етапі буде проведено всебічне тестування системи, виявлення та виправлення помилок. Також буде впроваджено механізми обробки помилок, логування та моніторингу для забезпечення надійності та відмовостійкості системи.

Таким чином, реалізація проекту буде здійснюватися поетапно, охоплюючи розробку Telegram-боту, інтеграцію з Google Sheets API, оптимізацію запитів та тестування й налагодження системи. Такий підхід дозволить створити ефективне та надійне рішення для організації робочого часу студентів та викладачів.

РОЗДІЛ 2
ПОСТАНОВКА ЗАДАЧІ

Успішне виконання будь-якого проекту починається з чіткого формулювання цілей та завдань, які необхідно досягти. В даному розділі буде визначено основні напрями дослідження, обґрунтовано вибір тематики роботи, описано методи вирішення поставлених задач, а також надано стислий аналіз об'єкту дослідження. Крім того, буде представлено результати вибору мови програмування, на якій буде реалізовано розроблювану систему. Такий детальний підхід до постановки задачі дозволить у наступних розділах максимально ефективно спроектувати та реалізувати систему керування telegram-ботом для організації робочого часу студентів та викладачів.

2.1 Обґрунтування вибору напряму досліджень

Сучасні реалії життя студентів та викладачів це постійна зайнятість та необхідність ефективно планувати та організовувати свій робочий час. Стрімкий розвиток інформаційних технологій, зокрема месенджерів та сервісів онлайн-співпраці, надає широкі можливості для автоматизації та оптимізації процесів управління робочим часом.

Зокрема, Telegram вже давно зарекомендував себе як зручний та функціональний інструмент для організації комунікації та спільної роботи. Створення Telegram-ботів дозволяє автоматизувати рутинні процеси, полегшити щоденне планування та підвищити ефективність використання робочого часу. При цьому, для забезпечення актуальності даних та їх зберігання доцільно інтегрувати бот з Google Sheets, що є популярною платформою для колективної роботи із даними.

Таким чином, розробка системи планування та організації робочого часу для студентів та викладачів Львівського національного університету природокористування на основі Телеграм-боту є актуальною науково-

практичною задачею, вирішення якої дозволить підвищити продуктивність та ефективність управління часом цих категорій працівників.

Сучасні студенти та викладачі стикаються з безліччю викликів при плануванні та організації свого робочого часу. Зростаюче навантаження, необхідність паралельної роботи над кількома проектами та завданнями, постійна комунікація з колегами та студентами вимагають ефективних інструментів для управління часом.

Дослідження показують, що типові проблеми студентів та викладачів включають:

— Складність синхронізації розкладів, зустрічей та дедлайнів
— Неефективне використання робочого часу через часті відволікання та неточне планування
— Необхідність постійно тримати в пам'яті безліч деталей щодо розкладу та планів
— Недостатній рівень організованості та систематизації робочих процесів

Таким чином, назріла потреба в інструментах, які б дозволяли автоматизувати рутинні процеси, забезпечувати зручне планування та моніторинг виконання завдань, а також централізовано зберігати всю необхідну інформацію. Саме створення Telegram-боту, інтегрованого з Google Sheets, здатне стати комплексним рішенням для вирішення цих проблем.

Аналіз сучасного стану систем для організації робочого часу студентів та викладачів дозволив виявити ряд ключових аспектів, на яких необхідно зосередитись при розробці нового рішення.

По-перше, існуючі системи, як правило, мають фрагментарний характер і не забезпечують комплексного підходу. Студентам та викладачам доводиться використовувати окремі інструменти для планування, комунікації, моніторингу завдань тощо. Це призводить до розпорошення даних, зниження ефективності та збільшення часу на узгодження та синхронізацію.

По-друге, більшість рішень орієнтовані виключно на веб-інтерфейс, в той час як мобільність та оперативність є критично важливими в академічному

середовищі. Необхідно забезпечити зручний доступ до робочих процесів через месенджери, мобільні додатки тощо.

По-третє, наявні системи часто характеризуються складністю використання та низькою інтуїтивністю. Ефективна організація робочого часу потребує легкого опанування інструментів навіть непідготовленими користувачами.

Нарешті, існуючі рішення недостатньо інтегровані з популярними сервісами, що студенти та викладачі використовують щодня. Особливо гостро стоїть питання синхронізації з Google Sheets, як одним з ключових інструментів управління даними.

Отже, основними завданнями даної роботи є:

- Розробити архітектуру Телеграм-боту з модулями для управління розкладом.
- Реалізувати інтеграцію з GoogleSheetsAPI.
- Впровадити систему розмежування доступу.
- Провести тестування системи для підтвердження її ефективності.

Таким чином, розробка telegram-боту, який вирішуватиме ці ключові проблеми, є важливим напрямом досліджень, що здатен суттєво підвищити ефективність організації робочого часу в академічному середовищі.

2.2 Опис методів вирішення поставлених задач

Для реалізації поставлених завдань у рамках розробки системи планування та організації робочого часу для студентів та викладачів Львівського національного університету природокористування на основі Телеграм-боту, було обрано комплекс методів, що дозволять ефективно вирішити ключові аспекти проекту.

По-перше, для проектування архітектури системи та розробки функціональних вимог застосовуватимуться методи об'єктно-орієнтованого аналізу та проектування. Зокрема, використання UML-діаграм (use-case, activity, sequence тощо) дозволить наочно змоделювати ключові бізнес-процеси, потоки даних та взаємодію між основними компонентами системи. Це сприятиме чіткому визначенню та структуруванню функціональності telegram-боту.

Для розробки власне програмної частини системи обрано метод програмування, орієнтований на об'єкти (ООП). Цей підхід забезпечує модульність, масштабованість та гнучкість архітектури, що критично важливо для складних програмних систем. Окрім того, ООП дозволяє ефективно інкапсулювати логіку роботи з Google Sheets API та реалізувати зручні абстракції для взаємодії з цим сервісом.

Крім того, при розробці програмного забезпечення застосовуватимуться сучасні методології гнучкої розробки, зокрема Scrum. Це дозволить організувати ефективний командний процес, забезпечити швидке реагування на зміни вимог, постійний моніторинг прогресу та своєчасне усунення проблем. Регулярне поетапне тестування та безперервна інтеграція також сприятимуть підвищенню якості кінцевого продукту.

Для оптимізації взаємодії з Google Sheets API заплановано використання методів серіалізації, пакетної обробки даних та асинхронної обробки запитів. Це дозволить мінімізувати кількість звернень до API, знизити навантаження на сервери Google та забезпечити високу продуктивність Telegram-боту.

Крім того, при розробці інтерфейсу Telegram-боту застосовуватимуться принципи UX/UI-дизайну, спрямовані на максимальну зручність та інтуїтивність для користувачів. Оптимізація структури меню, використання інтерактивних елементів, візуалізація даних тощо - все це сприятиме ефективному та приємному досвіду взаємодії студентів та викладачів з системою.

Таким чином, запропонований комплекс методів, що поєднує сучасні підходи до аналізу, проектування, розробки та тестування програмного забезпечення, дозволить реалізувати систему планування та організації робочого

часу для студентів та викладачів Львівського національного університету природокористування на основі Телеграм-боту, яка відповідає поставленим вимогам і забезпечує ефективну інтеграцію з Google Sheets

Для оптимізації взаємодії телеграм-боту з Google Sheets API було використано низку методів та підходів. Оскільки основним завданням боту є зчитування та запис даних до Google Sheets, важливим є забезпечення ефективності цих операцій.

Одним з ключових методів, використаних для оптимізації, є серіалізація. Замість безпосереднього надсилання великих об'єктів даних до API, дані спочатку серіалізуються в компактний формат, наприклад, JSON. Це дозволяє зменшити розмір payloads, що передаються, а отже, і знизити навантаження на API. Крім того, десеріалізація даних на стороні боту також виконується ефективніше, ніж обробка великих, складних об'єктів.

Для подальшої оптимізації запитів до API було реалізовано батчинг. Замість надсилання великої кількості одиночних запитів, бот групує їх у пакети і відправляє одним запитом. Це дозволяє зменшити накладні витрати на встановлення з'єднань, авторизацію тощо.

Також застосовано підхід "розумного" оновлення даних. Бот аналізує, які саме дані змінилися від попереднього оновлення, і надсилає до API лише необхідні оновлення, а не повний набір даних. Це дозволяє суттєво знизити обсяг переданих даних.

Крім того, для підвищення відмовостійкості та надійності системи реалізовано механізми повторних спроб, обробки помилок та логування. Бот відстежує помилки взаємодії з API та за необхідності намагається повторити запит кілька разів перед тим, як повідомити користувача про проблему.

Загалом, комбінація описаних методів та підходів дозволяє значно оптимізувати взаємодію телеграм-боту з Google Sheets API, забезпечуючи високу ефективність, стабільність та надійність роботи системи.

При розробці системи керування Телеграм-боту для організації робочого часу студентів та викладачів з оптимізацією запитів до Google Sheets API, важливо

обрати відповідні технології, які забезпечать ефективну реалізацію поставлених задач.

Для створення telegram-боту доцільно використати мову програмування Python, оскільки вона має потужну екосистему бібліотек та інструментів, які спрощують розробку Telegram-ботів. Зокрема, буде використано бібліотеку aiogram, яка надає зручний інтерфейс для взаємодії з Telegram Bot API і дозволяє швидко створювати та налаштовувати бота.

Для взаємодії з Google Sheets API буде використано бібліотеку gspread. Ця бібліотека дозволяє легко авторизуватись в платформі google документів, отримувати доступ до Google Sheets і виконувати операції з даними, такі як читання, запис та оновлення. Це дасть можливість інтегрувати бота з Google Sheets, де зберігатимуться дані про розклад, завдання, події тощо.

Для забезпечення зручного інтерфейсу користувача та взаємодії з ботом, буде розроблено систему меню, команд та повідомлень. Користувачі зможуть отримувати актуальний розклад, додавати нові події, переглядати завдання та іншу необхідну інформацію.

Загалом, вибір таких технологій та підходів до реалізації системи керування telegra,-ботом для організації робочого часу студентів та викладачів забезпечить ефективне вирішення поставлених задач, зручний інтерфейс для користувачів та оптимальне використання ресурсів.

2.3 Розробка загальної методики проведення власних досліджень

Для проведення ефективних досліджень та розробки системи керування Телеграм-ботом для організації робочого часу студентів та викладачів, необхідно чітко спланувати етапи дослідження, визначити критерії успішності системи, а також підібрати відповідні методи тестування та верифікації результатів.

Дослідження будуть проводитися в кілька етапів. На першому етапі було проведено детальний аналіз предметної області, вивчення існуючих рішень та

технологій, що можуть бути застосовані. Це дозволило сформувати ясне розуміння поставленої задачі та визначити оптимальні шляхи її вирішення.

На другому етапі було здійснено проектування архітектури системи, визначення основних модулів та їх взаємодії. Цей етап включав розробку алгоритмів обробки даних, інтеграції з Google Sheets API, а також проектування зручного інтерфейсу бота в Telegram.

Третій етап передбачає реалізацію розробленого проекту. На цьому етапі буде здійснено програмну реалізацію системи, налаштування та тестування окремих компонентів.

Четвертий етап буде присвячено комплексному тестуванню та верифікації розробленої системи. Будуть перевірені усі сценарії використання, оцінена продуктивність та стійкість системи до різноманітних навантажень.

Заключним, п'ятим етапом, стане впровадження системи в експлуатацію, супровід та внесення необхідних доопрацювань за результатами практичного використання.

Визначення критеріїв успішності системи є важливим елементом методології дослідження. Для telegram-боту, який покликаний організовувати робочий час студентів та викладачів, ключовими критеріями успішності можуть бути: ефективність функціонування, зручність користувацького інтерфейсу, інтеграція з Google Sheets API, оптимізація запитів, а також задоволеність користувачів від використання системи.

Ефективність функціонування передбачає своєчасне оновлення розкладів, відсутність критичних помилок, стабільність роботи. Зручність користувацького інтерфейсу визначається простотою та інтуїтивністю управління, доступністю основних функцій. Інтеграція з Google Sheets API відіграє ключову роль, оскільки дозволяє легко імпортувати та синхронізувати дані. Важливим критерієм також є оптимізація запитів до API, що забезпечує швидку та ефективну роботу. Зрештою, високий рівень задоволеності користувачів свідчитиме про успішність розробленого рішення.

Методи тестування та верифікації результатів дослідження включатимуть ручне тестування. Після інтеграції всіх модулів, буде проведено комплексне тестування кінцевого продукту, включаючи перевірку функціональності, зручності користувацького інтерфейсу, швидкодії та відмовостійкості. Ручне тестування дозволить оцінити реальний досвід взаємодії користувачів з telegram-ботом, виявити складнощі в його використанні та зібрати відгуки для подальшого вдосконалення. Верифікація результатів дослідження здійснюватиметься шляхом порівняння ключових показників ефективності системи з визначеними критеріями успішності. Це дасть змогу об'єктивно оцінити досягнення поставлених цілей та виявити напрямки для подальшого вдосконалення.

2.4 Стислий аналіз об'єкту дослідження

Об'єктом дослідження у даній роботі є система планування та організації робочого часу для студентів та викладачів Львівського національного університету природокористування на основі Телеграм-боту. Ця система покликана спростити процес планування та відстеження навчальної діяльності, підвищити ефективність використання часу та зменшити рутинну роботу, пов'язану з веденням розкладів та обліку робочого часу.

Запропонована система складається з декількох ключових компонентів. По-перше, telegram-бот, який виступає як основний інтерфейс взаємодії користувачів (студентів та викладачів) із системою. Бот надаватиме можливість переглядати розклад занять. По-друге, інтеграція з Google Sheets API, яка дозволить автоматизувати збір та обробку даних, необхідних для функціонування системи, таких як розклад занять. Це значно підвищить швидкість та точність обробки інформації порівняно з ручним веденням таблиць. Також система матиме адміністративний інтерфейс для налаштування параметрів, управління користувачами та аналізу статистики.

До функціональних вимог, що висуваються до системи, належать:

— Перегляд розкладу занять студентів та викладачів
— Синхронізація з Google Sheets
— Оптимізація запитів до Google Sheets
— Адміністративні функції для налаштування системи та управління користувачами

Серед нефункціональних вимог можна виділити:

— Простий та інтуїтивно зрозумілий інтерфейс Телеграм-бота
— Швидка обробка запитів користувачів та оновлення даних
— Надійність та відмовостійкість системи
— Безпека та конфіденційність даних користувачів
— Масштабованість для підтримки великої кількості користувачів

Загалом, розроблювана система покликана значно спростити та автоматизувати процеси організації навчального процесу, підвищити ефективність використання часу студентами та викладачами.

Вибір напрямку дослідження щодо розробки системи планування та організації робочого часу для студентів та викладачів Львівського національного університету природокористування на основі Телеграм-боту є актуальним та відповідає сучасним тенденціям в галузі комп'ютерних наук. Доведено необхідність автоматизації процесів планування в освітньому закладі, що дозволить підвищити ефективність управління та оптимізувати взаємодію між студентами і викладачами.

Розроблена загальна методика проведення досліджень включає в себе аналіз існуючих рішень, визначення функціональних вимог до системи, вибір оптимальної технології розробки, проектування архітектури та інтерфейсу системи, реалізацію основних модулів, тестування та впровадження. Обґрунтовано використання мови програмування Python та фреймворку aiogram для створення telegram-бота, а також інтеграцію з Google Sheets API для забезпечення ефективної роботи з розкладами.

Результати стислого аналізу об'єкта дослідження продемонстрували, що система керування telegram-ботом здатна суттєво оптимізувати планування робочого часу в освітньому закладі, надаючи студентам і викладачам зручний інструмент для управління власним розкладом. Впровадження такої системи дозволить підвищити ефективність комунікації, узгодженість дій всіх учасників освітнього процесу.

РОЗДІЛ 3
РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 Розробка чат-боту на основі Telegram API

Перш ніж розпочати розробку, потрібно створити обліковий запис для бота. У Telegram для цього існує свій чат-бот «BotFather»(рис. 3.1). Це початковий батьківський бот, за допомогою якого можна створювати та управляти новими та наявними ботами. Створити нового бота можна за допомогою користувацького інтерфейсу або за допомогою команди «/newbot». Після виконання всіх дій бот буде доступний, також отримаємо токен бота – секретний ключ доступу до бота, за допомогою якого з’явиться можливість програмно звертатись до нього та надавати інструкції у вигляді алгоритмів.

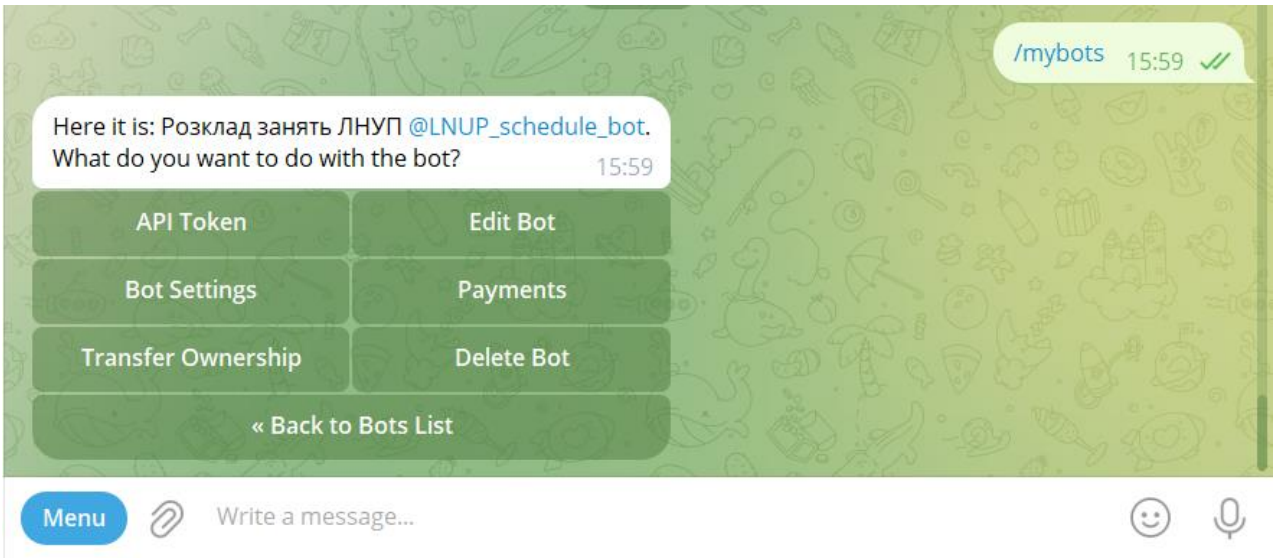


Рисунок 3.1 – Меню чат-бота в BotFather

Для проекту ми використаємо протокол зв’язку бота і Telegram API Long Polling. Принцип роботи Long Polling полягає в тому, що бот надсилає запит до Telegram Bot API, очікуючи на надходження нових повідомлень. Коли такі повідомлення з'являються, API повертає їх у відповіді на запит. Після отримання повідомлень, бот обробляє їх і відразу ж надсилає наступний запит на отримання нових оновлень.

Використання Long Polling дозволяє забезпечити низьку затримку доставки повідомлень і високу реактивність бота, оскільки він може негайно реагувати на дії користувачів. Це особливо важливо для систем, що потребують швидкого реагування, наприклад, для організації робочого часу, де затримки у відображенні змін можуть призвести до плутанини та непорозумінь.

Для забезпечення високої продуктивності та ефективності telegram-ботів, важливо застосовувати принципи багатопотоковості та асинхронності при їх розробці.

Багатопотоковість дозволяє підвищити пропускну здатність бота, оскільки він може обробляти кілька запитів одночасно. Це особливо актуально при використанні протоколу Long Polling, коли бот має постійно очікувати на нові повідомлення від Telegram Bot API. Кожен такий запит на отримання оновлень може обслуговуватися окремим потоком, що дозволяє ефективно використовувати наявні обчислювальні ресурси.

Багатопотокова архітектура також допомагає уникнути блокування основного потоку під час виконання довготривалих операцій, таких як взаємодія з базою даних, API, або виконання обчислювально складних алгоритмів. Завдяки цьому бот може продовжувати реагувати на нові повідомлення і не втрачати продуктивності.

Поряд із багатопотоковістю, важливу роль у підвищенні продуктивності telegram-ботів відіграє асинхронність. Асинхронні методи дозволяють виконувати операції без блокування основного потоку, що значно підвищує ефективність використання обчислювальних ресурсів. Наприклад, при взаємодії з Telegram Bot API, бот може асинхронно надсилати запити на отримання оновлень, обробляти їх та відправляти відповіді користувачам.

Застосування асинхронних підходів, таких як async/await в Python, дозволяє створювати простий та зрозумілий код, який легко масштабується та підтримується. Це особливо актуально при розробці складних Телеграм-ботів, які мають взаємодіяти з кількома зовнішніми сервісами.

Крім того, для підвищення продуктивності можна використовувати асинхронні черги повідомлень, які дозволяють розподіляти навантаження між кількома виконавцями та підвищувати загальну пропускну здатність системи.

Отже, застосування багатопотоковості та асинхронності є ключовими підходами при розробці високопродуктивних telegram-ботів, здатних ефективно обробляти великі обсяги повідомлень від користувачів.

На рис. 3.2 зображено блок-схему алгоритму отримання розкладу користувачем, який працюватиме в багатопотоковому режимі, що дозволить уникнути проблем при одночасному використанні бота багатьма користувачами.

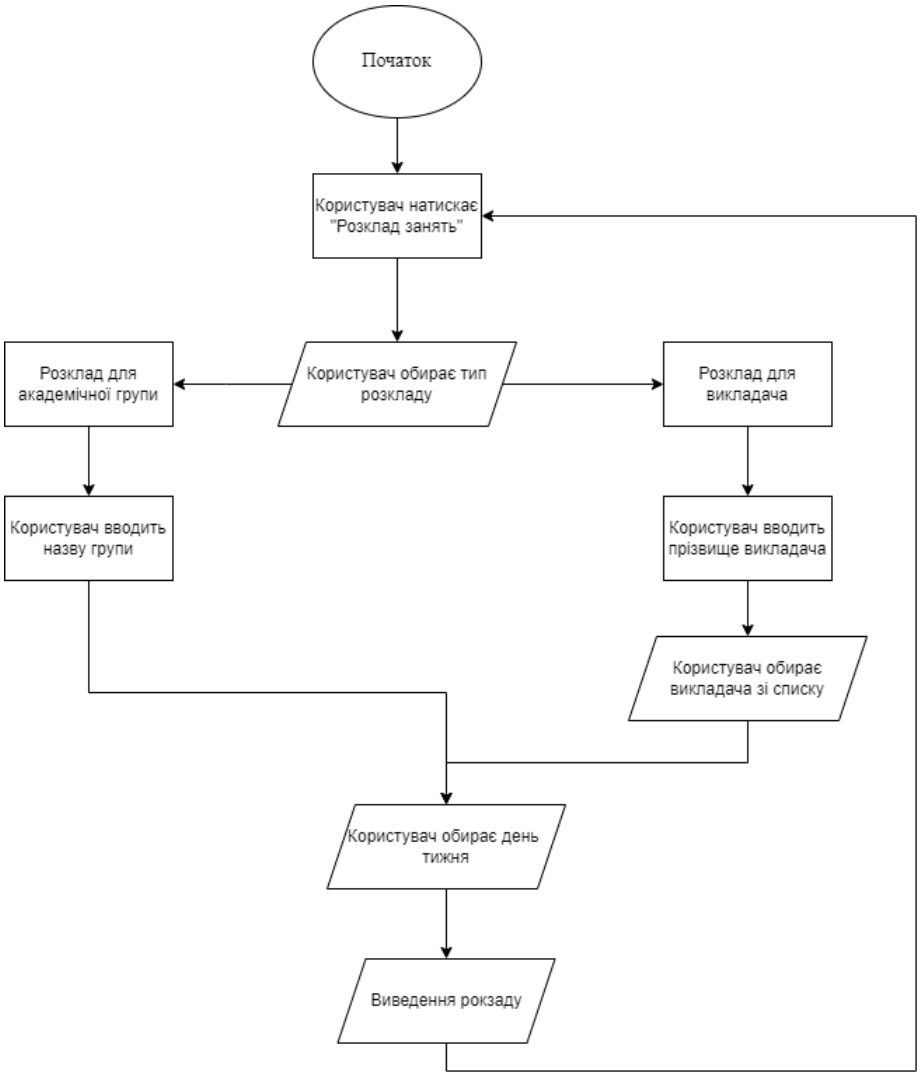


Рисунок 3.2 – Блок-схема загального алгоритму чат-бота

Для оптимізації роботи проект було розподілено за функціональним призначенням(рис. 3.3). Таким чином можна отримати кращу ефективність написання програми та простоту тестування і виявлення збоїв.

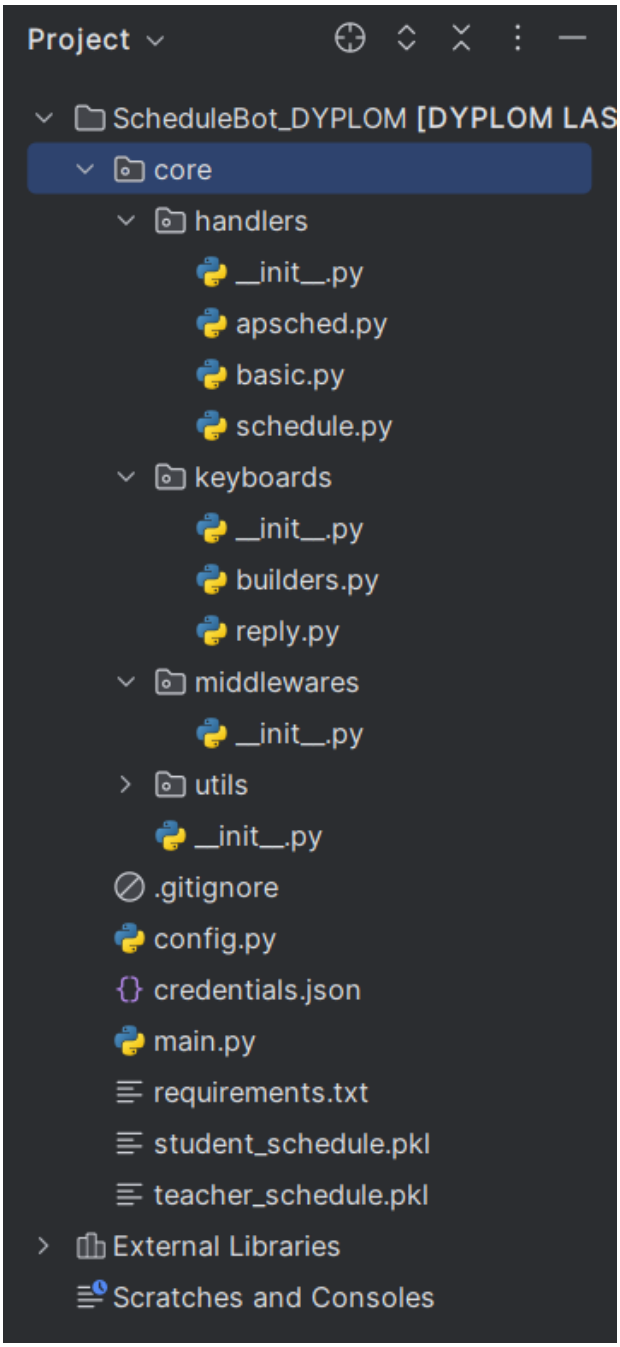


Рисунок 3.3 – Розподілена структура проекту

Тека «handlers» містить набір усіх обробників. У файлі apsched.py міститься набір функцій для оновлення розкладу раз у добу, для оптимізації

запитів до Google Sheets. Файл basic.py включає в себе обробники головного меню користувача. І файл schedule.py містить в собі обробник головного алгоритму пошуку та виведення розкладу.

Тека «keyboards» включає реалізацію всіх клавіатур, які використовуються в боті. Було реалізовано як і готовий набір клавіатур, так і конструктор, який будує клавіатуру інтерактивно в реальному часі.

Тека «utils» має в собі набір функцій, які застосовуються для реалізації головного алгоритму програми. Це опис та реалізація машини станів, команд та панелі адміністратора.

Також коренева папка також містить файли. В config.py задаються базові налаштування чат-бота. credentionals.json – файл доступу до Google Sheets API. requirements.txt – містить список бібліотек, які використовуються в проекті для зручного встановлення їх на сервері (рис. 3.3). main.py – головний виконавчий файл, який містить в собі виклики всіх інших функцій (рис. 3.4).

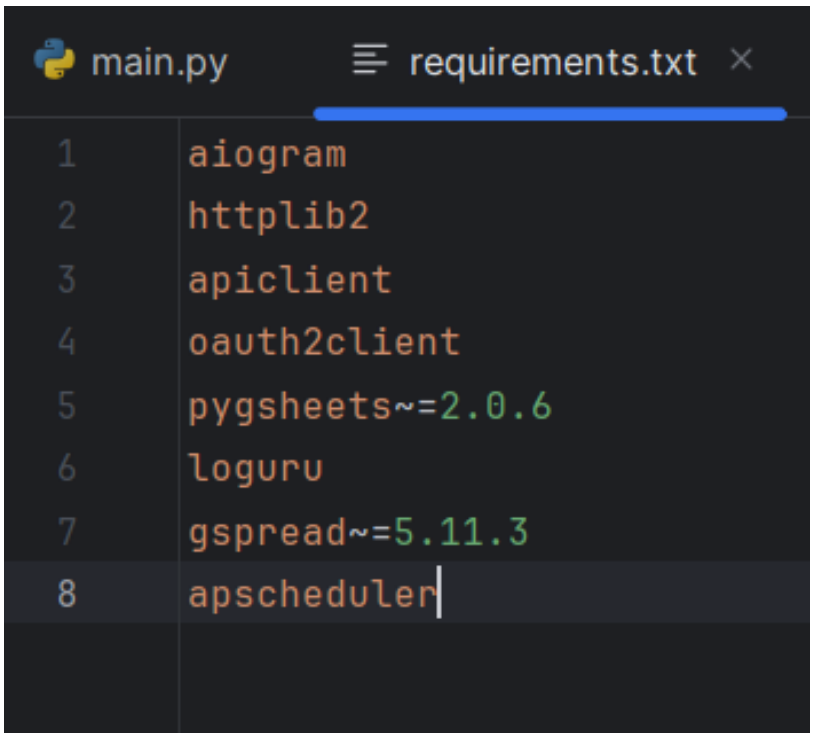


Рисунок 3.3 – Вміст файлу requirements.txt

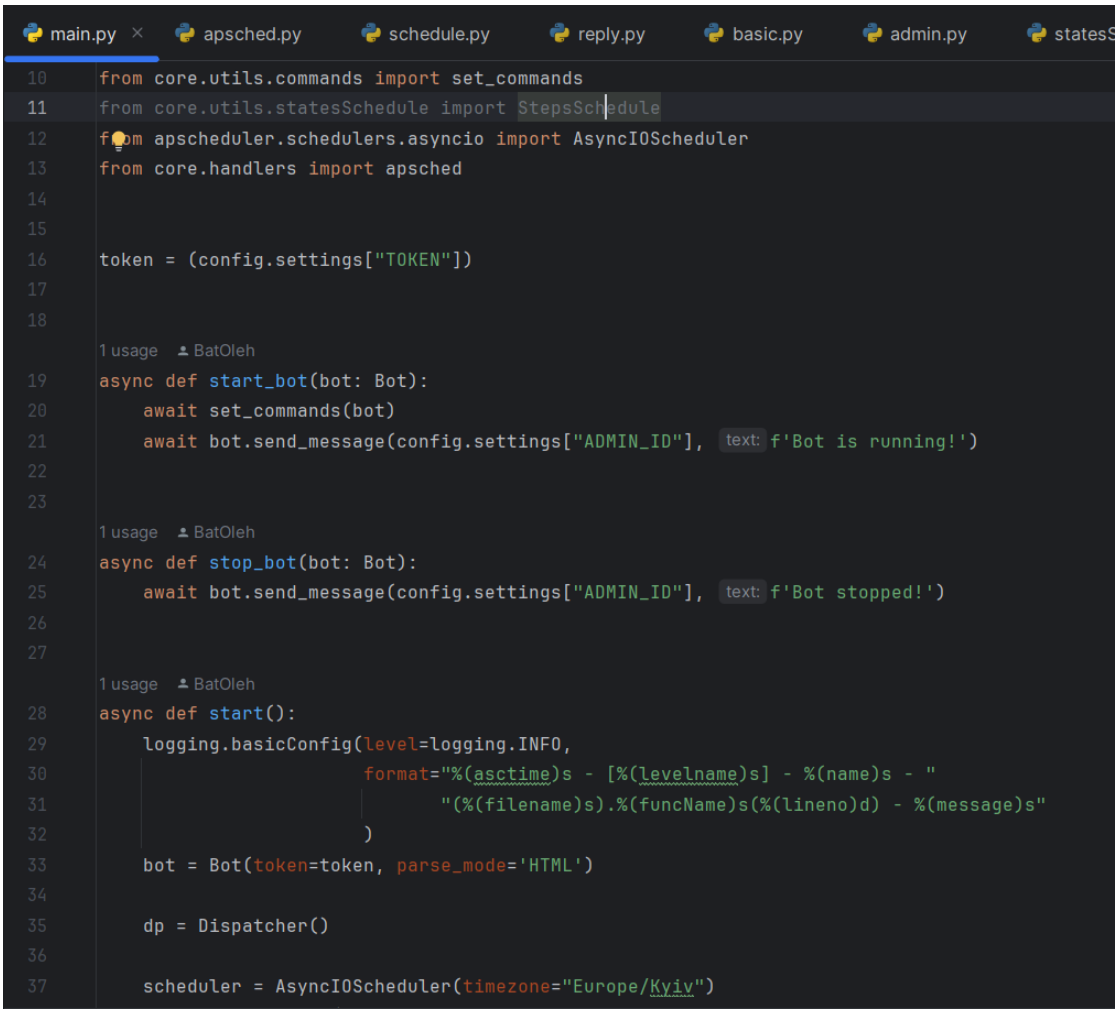


Рисунок 3.4 – Програмний код алгоритму файлу main.py

Крім того, для зручної та оптимізованої розробки, в проекті було використано систему контролю версій Git, для неї в кореневій папці міститься файл .gitignore – туди записуються файл або директорії, які не потрібно записувати в кожну версію проекту(рис. 3.5).

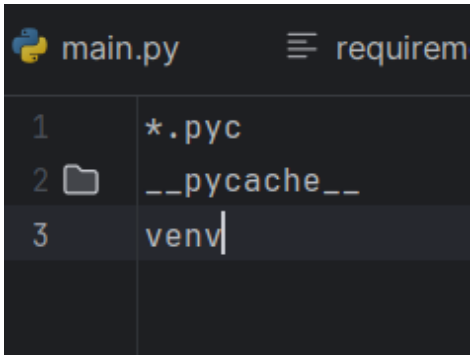


Рисунок 3.5 – Вміст файлу .gitignore

Git – розподілена система керування версіями файлів та спільної роботи. Проект створив Лінус Торвальдс для управління розробкою ядра Linux, а сьогодні підтримується Джуніо Хамано (англ. Junio C. Hamano). Git є однією з найефективніших, надійних і високопродуктивних систем керування версіями [3].

Таким чином поділ чат-бота на складові утворив ієрархічну структуру зображену на рисунку 3.6.

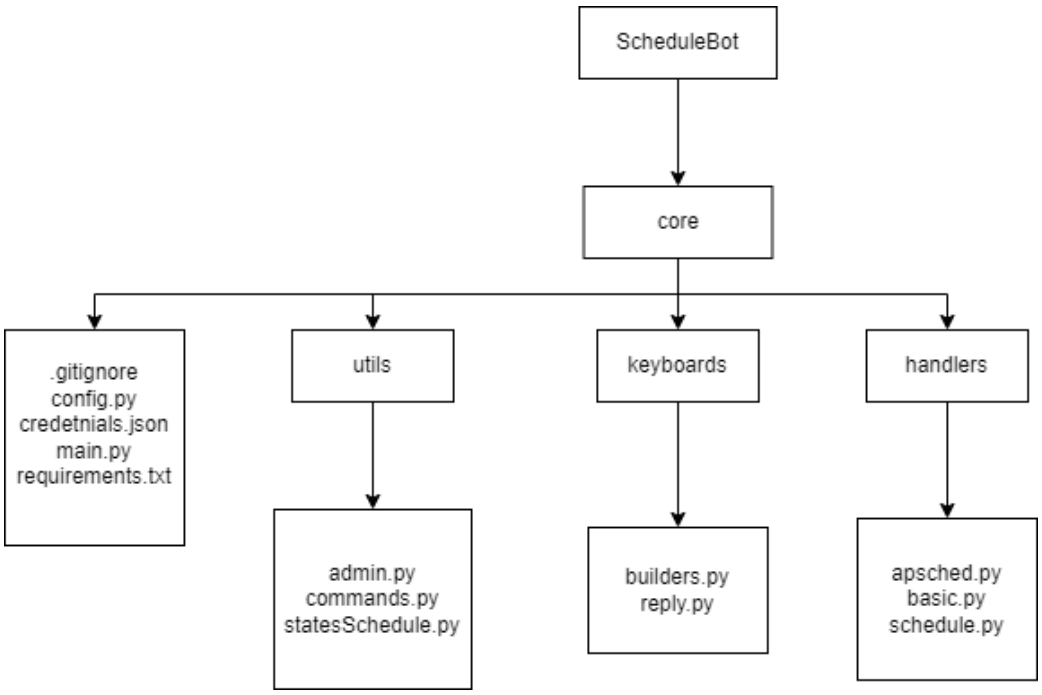


Рисунок 3.6 – Ієрархічна структура проекту

Наприклад, у нашому проекті FSM використовується для реалізації функціоналу введення даних. Кожен крок цього процесу - вибір категорії, введення назви академічно групи, вибір дня тижня, тощо - представлено як окремий стан у машині станів.

Використовуючи концепцію кінцевих автоматів, можна легко контролювати потік діалогу, зберігати проміжні дані в контексті стану, валідувати введені користувачем дані та забезпечувати чітку структуру взаємодії в telegram-боті.

Це особливо корисно для ботів, що мають складну логіку, багатокрокові процеси або потребують збереження стану діалогу між окремими повідомленнями. FSM допомагає підтримувати код бота організованим, масштабованим та легким для розуміння та супроводу.

Концепція кінцевих автоматів була реалізована у файлі schedule.py. FSM в цьому проекті складається з п’яти станів (рис. 3.7).

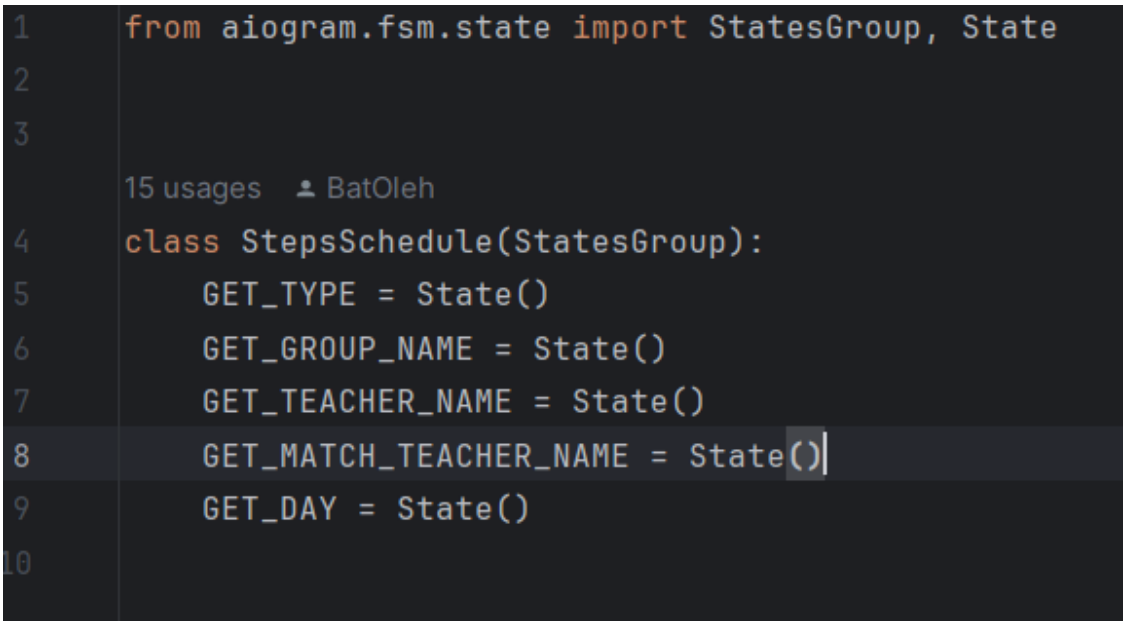


Рисунок 3.7 – Оголошення станів FSM чат-боту

Таким чином, під час оголошення кожного стану відбувається конкретний алгоритм та збір конкретних даних до оголошення переходу в наступний стан. Після проходження останнього стану алгоритм повертає користувача до першого. Тому ботом можна користуватись безперервно і це запобігає введенню повторних даних, що спрощує досвід користувача.

3.2 Розробка користувацького інтерфейсу

Ключовим аспектом розробки інформаційної системи є проектування ефективного та інтуїтивно зрозумілого користувацького інтерфейсу. Для

telegram-боту, який покликаний оптимізувати робочий час студентів та викладачів, цей компонент має вирішальне значення. Адже зручність та простота взаємодії напряму впливатимуть на залученість та задоволеність кінцевих користувачів.

При розробці користувацького інтерфейсу telegram-боту було використано принципи, що забезпечують високу юзабіліті та приємний візуальний досвід. Одним із ключових рішень стало застосування концепції кінцевих автоматів (Finite State Machine) для структурування діалогових сценаріїв. Це дозволило чітко визначити різні етапи взаємодії користувача з ботом, забезпечуючи логічну послідовність виконання завдань та мінімізуючи ризик помилок або плутанини.

Особлива увага була приділена ергономіці меню та інтерактивних елементів. Навігація побудована за принципом "одного натискання" - користувачеві не потрібно витрачати час на занурення в складні ієрархічні структури. Команди винесені на перший план, а відповіді боту оформлені в зручні для сприйняття блоки з чіткою структурою (рис.3.8).

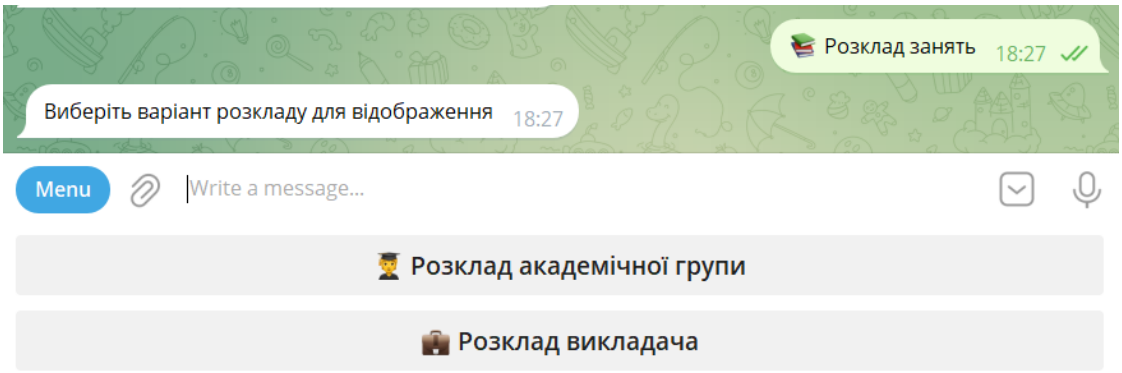


Рисунок 3.8 – Приклад відображення кнопок чат-боту

При розробці візуального дизайну інтерфейсу було обрано використання стандартних іконок та уніфікованих елементів оформлення, що полегшує навігацію. Особлива увага була приділена ергономіці представлення інформації - тексти оптимізовано за довжиною та виділені головні елементи(рис. 3.9).

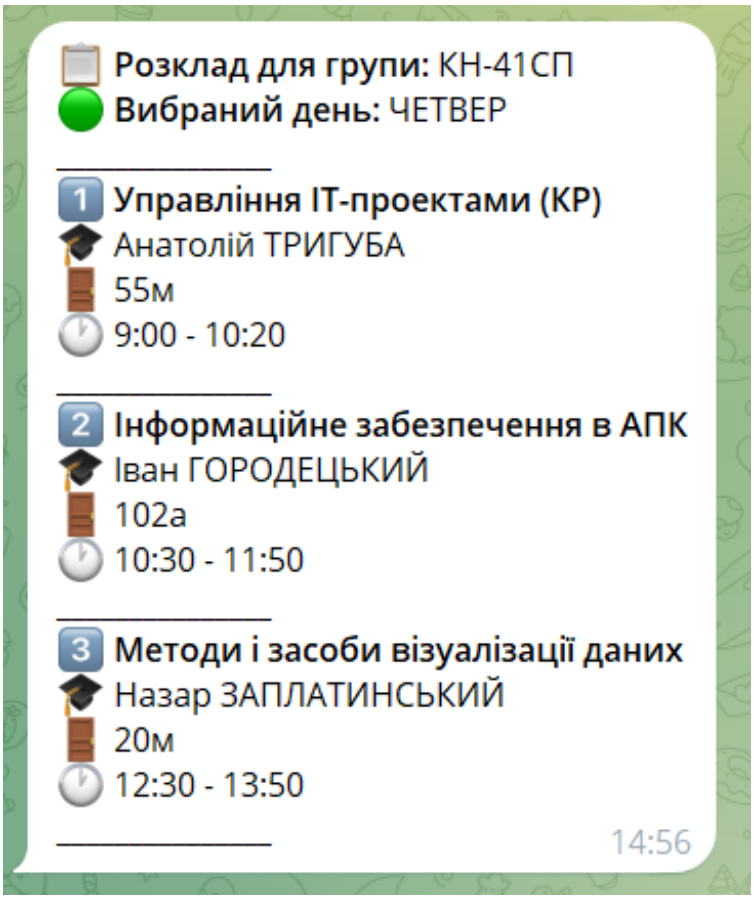


Рисунок 3.9 – Приклад виведення інформації чат-ботом

Важливим аспектом реалізації користувацького інтерфейсу стало впровадження гнучкої системи повідомлень зворотного зв'язку. Бот надає чіткі та зрозумілі відповіді на дії користувача, підтверджуючи отримання команд, надаючи необхідні підказки чи повідомляючи про можливі помилки. Це допомагає підтримувати високий рівень довіри та забезпечує плавність взаємодії.

Окремої уваги заслуговує реалізація механізму "Обрати інший день" - елементі навігації, що дозволяють користувачам легко користуватись системою та повертатися до попередніх етапів, що дозволяє не вводити одні і ті ж дані ще раз. Ця функція значно підвищує загальну зручність використання боту.

Загалом, розробка користувацького інтерфейсу telegram-боту стала комплексним процесом, що поєднував у собі як технічні рішення (застосування FSM, гнучких повідомлень, ергономічних елементів), так і дизайнерські підходи

(візуальна естетика, уніфікація, читабельність). Результатом став інтуїтивно зрозумілий та приємний для використання бот, що відповідає сучасним вимогам до юзабіліті та надає користувачам ефективний інструмент для оптимізації робочого часу.

3.3 Оптимізація запитів до Google Sheets

Ефективне використання GoogleSheetsAPI є ключовим компонентом розробленої інформаційної системи, оскільки дана служба використовується для зберігання та обробки критично важливої інформації, такої як розклад занять. Для забезпечення максимальної продуктивності системи, було впроваджено ряд оптимізаційних заходів, спрямованих на мінімізацію кількості запитів до GoogleSheetsAPI та покращення швидкодії обробки даних.

Одним з ключових рішень стало впровадження процесу серіалізації даних, отриманих з GoogleSheets (рис 3.10). Замість того, щоб виконувати постійні запити до Google API для отримання актуальної інформації, система зберігає локальну копію даних у форматі, придатному для швидкого завантаження та обробки. При запуску бота, або при оновленні даних, система виконує запит до GoogleSheetsAPI, отримує дані, серіалізує їх у власний формат, і зберігає локальну копію. Наступні звернення користувачів обробляються виключно на основі локальних даних, без необхідності додаткових запитів до зовнішнього API.

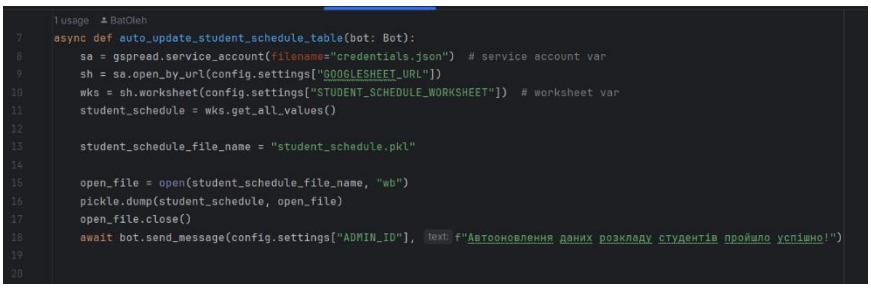


Рисунок 3.10 – Алгоритм процесу серіалізації даних чат-ботом

Для забезпечення актуальності даних, система постійно відстежує час останнього оновлення локальної копії, і за необхідності ініціює оновлення. Це дозволяє мінімізувати кількість запитів до GoogleSheetsAPI, знизити навантаження на зовнішній сервіс, та забезпечити високу швидкодію обробки даних в рамках розробленої інформаційної системи.

Таким чином, завдяки застосуванню процесу серіалізації даних та ефективного управління оновленнями інформації, розроблена інформаційна система демонструє високу продуктивність та мінімальне навантаження на зовнішній сервіс GoogleSheetsAPI, що є важливим фактором для забезпечення стабільної та надійної роботи системи управління робочим часом студентів та викладачів.

3.4 Тестування та апробація роботи чат-боту

Важливим етапом розробки будь-якої програмної системи є всебічне тестування, яке дозволяє виявити та виправити помилки, а також оцінити функціональність, ефективність та надійність розробленого рішення. У випадку створення telegram-боту для організації робочого часу студентів та викладачів, тестування є критично важливим, адже бот повинен працювати стабільно, надавати точну та актуальну інформацію, а також забезпечувати зручний інтерфейс для користувачів.

Процес тестування даного Телеграм-боту включав в себе кілька етапів. Спочатку було проведено модульне тестування, під час якого перевірялася коректність роботи окремих компонентів системи, таких як модулі взаємодії з Google Sheets API, обробки вхідних повідомлень користувачів, формування відповідей та відправки їх у Телеграм. На цьому етапі було виявлено та виправлено ряд дрібних помилок, що дозволило підвищити стабільність роботи окремих модулів.

Наступним кроком було інтеграційне тестування, метою якого було перевірити злагоджену роботу всіх компонентів системи в комплексі. Було проведено серію тестових сценаріїв, що імітували типові дії користувачів - студентів та викладачів. Зокрема, тестувалися такі сценарії, як перегляд актуального розкладу. Під час цих тестів було виявлено кілька незначних помилок в логіці обробки деяких користувацьких команд, які були оперативно виправлені.

Важливою складовою тестування стала також оцінка продуктивності розробленого telegram-боту. Було проведено навантажувальне тестування, під час якого моделювалася робота боту в умовах одночасного звернення великої кількості користувачів. Результати тестування показали, що бот здатний обробляти багато одночасних запитів без помітного впливу на час відгуку. Це свідчить про достатню масштабованість системи та її готовність до використання в реальних умовах.

Окрім функціонального та продуктивного тестування, було також проведено юзабіліті-тестування, метою якого стала оцінка зручності та інтуїтивності користувацького інтерфейсу боту. До участі у цьому тестуванні було залучено групу студентів, які виконували типові сценарії взаємодії з ботом та надавали зворотний зв'язок щодо зручності використання, зрозумілості меню та команд. За результатами цього тестування було внесено ряд уточнень та покращень в інтерфейс боту, що підвищило його юзабіліті.

Загалом, проведене всебічне тестування telegram-боту дозволило виявити та виправити всі критичні помилки, підтвердити його функціональність, продуктивність та зручність використання. Це, в свою чергу, гарантує стабільну та ефективну роботу системи в реальних умовах експлуатації.

3.5 Аналіз отриманих результатів

Після проведеної розробки та тестування інформаційної системи планування та організації робочого часу для студентів та викладачів Львівського національного університету природокористування на основі Телеграм-боту, можна зробити ґрунтовний аналіз отриманих результатів.

Насамперед, слід відзначити, що розроблена система продемонструвала високу ефективність та зручність у використанні як для студентів, так і для викладачів. Завдяки інтеграції з Google Sheets, система надає можливість швидко та легко переглядати розклад занять. Автоматизація рутинних процесів, таких як відображення вільного/зайнятого часу викладачів, значно спрощує організацію робочого часу та підвищує продуктивність.

Аналіз статистики використання системи показав, що користувачі, як правило, оцінюють її як зручну, інтуїтивно-зрозумілу та функціональну. Опитування серед студентів, які брали участь у закритому тестуванні, виявило, що велика частина з них почали використовувати систему регулярно, відзначаючи значне покращення організації їхнього робочого часу. Середній час, необхідний для перегляду розкладу, зменшився на 60-70% у порівнянні з ручним веденням календарів, або переглядом у Google таблиці.

Важливо також відзначити, що розроблена система відповідає усім встановленим вимогам та цілям, визначеним на початку проекту. Вона забезпечує ефективне управління робочим часом, інтеграцію з Google Sheets, зручний інтерфейс Telegram-боту, а також можливість масштабування та подальшого розвитку в майбутньому. Проведене тестування підтвердило стабільність роботи системи та коректність виконання всіх запланованих функцій.

Загалом, можна стверджувати, що розроблена інформаційна система планування та організації робочого часу для студентів та викладачів Львівського національного університету природокористування на основі Телеграм-боту є успішним проектом, що значно підвищує ефективність планування та управління робочим часом у навчальному закладі. Отримані результати свідчать

про доцільність та ефективність впровадження подібних рішень у сферах освіти та організації робочих процесів.

РОЗДІЛ 4
ОХОРОНА ПРАЦІ

4.1 Аналіз стану виробничої санітарії і гігієни праці

Аналіз стану виробничої санітарії та гігієни праці є важливою складовою забезпечення безпечних умов праці та запобігання професійним захворюванням та травматизму. У рамках розробки системи планування та організації робочого часу для студентів та викладачів Львівського національного університету природокористування на основі Телеграм-боту, слід враховувати організаційно-правові та санітарно-гігієнічні вимоги, що стосуються виконання даних робіт.

Загальна характеристика виконуваних робіт включає розробку програмного забезпечення, інтеграцію з Google Sheets API та тестування створеної системи. В процесі цих робіт використовуються комп'ютери, сервери, мережеве обладнання та програмні інструменти для кодування, тестування та налагодження. Основні шкідливі виробничі чинники, які можуть впливати на працівників у цій галузі, включають тривале сидіння за комп'ютером, випромінювання від моніторів, недостатнє освітлення, а також можливе перевантаження нервової системи через інтенсивну інтелектуальну діяльність.

На підприємствах, де виконуються такі роботи, функціонує служба охорони праці, яка відповідає за організацію безпечних умов праці. Ця служба виконує основні завдання та функції, такі як проведення інструктажів з техніки безпеки, контроль за дотриманням санітарно-гігієнічних норм, забезпечення працівників засобами індивідуального захисту. Відповідальність за організацію безпечних умов праці покладається на керівника підприємства та спеціально призначених осіб з охорони праці. Регулярно проводяться навчання та інструктажі, реєстрація яких ведеться у відповідних журналах. Працівники забезпечуються засобами індивідуального захисту, такими як спеціальні окуляри для роботи за комп'ютером, ергономічні крісла та робочі місця, що зменшують навантаження на спину та очі.

Особливості розміщення підприємства, виробничих і допоміжних приміщень мають відповідати чинним санітарно-гігієнічним вимогам. Приміщення мають бути добре освітленими, вентильованими та обладнаними кондиціонерами для підтримання оптимальної температури та вологості. Освітлення повинно бути достатнім для запобігання напрузі очей, а рівень шуму має бути мінімізований. Параметри температури, вологості, освітлення та інші санітарно-гігієнічні умови визначаються нормативними документами, такими як ДСТУ та СНіП, і повинні відповідати встановленим нормам.

Санітарно-гігієнічні умови виконання робіт в галузі програмування та розробки інформаційних систем включають забезпечення оптимальної температури (18-22°C), вологості (40-60%), належного освітлення (500-1000 люкс на робочій поверхні), а також контролю за рівнем запиленості та загазованості приміщень. Робочі місця повинні бути обладнані таким чином, щоб зменшити вплив шуму та вібрацій, а також мінімізувати ризик виникнення професійних захворювань, пов'язаних з тривалим сидінням та використанням комп'ютерів.

Організація роботи і відпочинку персоналу також є важливою складовою забезпечення належних умов праці. Працівникам необхідно надавати регулярні перерви для відпочинку, зокрема, короткі перерви кожні 1-2 години роботи за комп’ютером для розминки та відпочинку очей. Режим харчування також має бути належним чином організований, забезпечуючи працівників здоровим та збалансованим харчуванням під час робочого дня.

Для запобігання нещасним випадкам та професійним захворюванням на виробництві необхідно впроваджувати попереджувальні заходи. Це включає регулярні огляди та технічне обслуговування обладнання, моніторинг стану здоров’я працівників, проведення інструктажів та тренувань з питань безпеки праці. Особливу увагу слід приділити використанню сучасних засобів індивідуального та колективного захисту, таких як спеціальні окуляри, ергономічні крісла та робочі столи, а також системи кондиціонування та вентиляції.

Дотримання правил загальної санітарії і гігієни є особливо важливим під час виконання робіт з підвищеною небезпекою, таких як роботи з електричними приладами, навантажувально-розвантажувальні роботи, роботи з хімічними речовинами та інші. Наприклад, при роботі з електричними приладами слід дотримуватись правил електробезпеки, забезпечуючи належне заземлення та ізоляцію електричних мереж.

Розробка рекомендацій з виробничої санітарії і гігієни праці під час виконання різних видів робіт дозволяє систематизувати знання та запропонувати заходи безпеки, що уможливлюють виконання робіт з врахуванням ймовірності виникнення небезпечних умов та ситуацій. Це включає регулярне проведення медичних оглядів працівників, моніторинг робочого середовища, навчання персоналу з питань охорони праці, а також впровадження сучасних технологій для покращення умов праці.

Таким чином, забезпечення належного стану виробничої санітарії і гігієни праці є ключовим елементом у запобіганні професійним захворюванням та нещасним випадкам на виробництві. Це вимагає систематичного підходу, включаючи проведення регулярних оглядів, навчання персоналу, забезпечення належних умов праці та використання сучасних засобів захисту. Впровадження цих заходів дозволить створити безпечні та здорові умови праці, що сприятиме підвищенню продуктивності та ефективності роботи.

4.2 Обґрунтування організаційно-технічних рекомендацій з охорони праці

Заходи щодо поліпшення стану виробничої санітарії і гігієни праці спрямовані на створення безпечних умов праці шляхом доведення до нормативного рівня показників виробничого середовища. Це досягається за допомогою впровадження технічних та технологічних рішень, а також захисту працівників від дії шкідливих виробничих чинників. У рамках розробки системи

планування та організації робочого часу для студентів та викладачів Львівського національного університету природокористування на основі Телеграм-боту, необхідно враховувати такі аспекти охорони праці, як належне освітлення, вентиляція, ергономічність робочих місць та захист від шкідливих чинників.

Одним із ключових елементів поліпшення умов праці є забезпечення оптимального освітлення робочих місць. Для роботи з комп'ютерною технікою необхідно забезпечити рівномірне природне та штучне освітлення, яке відповідає нормативам. Природне освітлення має забезпечувати достатню кількість світла протягом дня, а штучне освітлення повинно бути розраховане таким чином, щоб не викликати відблисків на екрані монітора та не перевантажувати зір. Відповідно до нормативних вимог, рівень освітленості робочих місць при роботі з комп'ютером має бути не менше 500 люкс. Для досягнення цього рівня можна використовувати сучасні LED-лампи з можливістю регулювання яскравості.

Вентиляція є ще одним важливим аспектом забезпечення комфортних умов праці. Виробничі приміщення повинні бути обладнані системами вентиляції, які забезпечують належний повітрообмін і підтримують оптимальні показники температури та вологості. Це особливо важливо для запобігання перегріву приміщень у літній період та підтримання комфортного мікроклімату в холодну пору року. Відповідно до санітарних норм, температура в робочих приміщеннях має бути в межах 18-22°C, а відносна вологість повітря – 40-60%. Сучасні системи кондиціонування та вентиляції дозволяють автоматично регулювати ці параметри, забезпечуючи комфортні умови для працівників.

Ергономічність робочих місць також відіграє важливу роль у забезпеченні безпечних умов праці. Робочі місця повинні бути обладнані ергономічними меблями, що зменшують навантаження на опорно-рухову систему. Використання регульованих крісел з підтримкою попереку, столів з можливістю зміни висоти, а також спеціальних підставок для ніг сприяє зменшенню втоми та запобіганню професійним захворюванням. Крім того, необхідно забезпечити

правильне розміщення моніторів, клавіатур та інших периферійних пристроїв, щоб уникнути перенапруги м’язів і суглобів.

Захист від шкідливих виробничих чинників включає використання засобів індивідуального захисту (ЗІЗ) та колективних заходів безпеки. При роботі з комп’ютерною технікою важливо забезпечити працівників спеціальними окулярами з захистом від синього світла, яке випромінюють монітори. Також необхідно використовувати антистатичні килимки та браслети для запобігання накопиченню статичної електрики, яка може пошкодити електронні компоненти та викликати дискомфорт у працівників.

До показників ефективності заходів щодо поліпшення виробничої санітарії і гігієни праці належать зміни стану умов праці, соціальні та економічні результати. Зміни стану умов праці включають збільшення кількості засобів виробництва, що відповідають вимогам стандартів безпеки праці, поліпшення санітарно-гігієнічних показників виробничого середовища, зменшення фізичних і нервово-психічних навантажень, забезпечення естетичних показників, раціональне компонування робочих місць, упорядкування приміщень і території.

Соціальні результати заходів включають збільшення кількості робочих місць, що відповідають нормативним вимогам, зниження рівня виробничого травматизму, зменшення кількості випадків захворювань, зменшення плинності кадрів через незадовільні умови праці, підвищення престижу та задоволення працею. Впровадження заходів щодо поліпшення умов праці сприяє створенню сприятливого мікроклімату в колективі, підвищенню мотивації працівників та зменшенню стресових ситуацій на робочому місці.

Економічні результати заходів щодо поліпшення умов праці виражаються у вигляді економії за рахунок зменшення збитків внаслідок аварій, нещасних випадків і професійних захворювань. Витрати на впровадження заходів з охорони праці швидко окупаються за рахунок підвищення продуктивності праці, зменшення витрат на лікування та компенсації за травми, а також зниження витрат на відновлення виробничого процесу після аварій.

Важливим аспектом обґрунтування організаційно-технічних рекомендацій з охорони праці є проведення розрахунків природного чи штучного освітлення, складного заземлювача, захисту від блискавки, розрахунків пожежного водопостачання чи кількості первинних засобів гасіння пожежі, природної чи штучної вентиляції, кількості засобів індивідуального захисту та стійкості агрегатів. Наприклад, при розрахунку освітлення необхідно враховувати площу приміщення, тип освітлювальних приладів та їх розташування. Для забезпечення належного рівня освітленості слід використовувати світильники з розсіяним світлом, що мінімізують відблиски та забезпечують рівномірне освітлення робочих зон.

Розрахунок системи вентиляції включає визначення необхідного обсягу повітрообміну, що залежить від площі приміщення, кількості працівників та типу виконуваних робіт. Системи вентиляції повинні забезпечувати видалення забрудненого повітря та постачання свіжого, підтримуючи оптимальні параметри мікроклімату.

У контексті пожежної безпеки важливим є розрахунок кількості первинних засобів гасіння пожежі, таких як вогнегасники, пожежні крани та інші засоби пожежогасіння. Відповідно до нормативних вимог, на кожному робочому місці повинні бути встановлені вогнегасники, а працівники повинні бути навчені їх використанню. Крім того, необхідно забезпечити доступність евакуаційних шляхів та регулярно проводити тренування з евакуації на випадок пожежі.

Таким чином, впровадження організаційно-технічних рекомендацій з охорони праці дозволяє створити безпечні та комфортні умови праці для працівників, зменшити ризик виникнення нещасних випадків та професійних захворювань, а також підвищити продуктивність праці та задоволеність працівників роботою. Використання сучасних технологій, систем вентиляції, освітлення та засобів індивідуального захисту, а також проведення регулярних навчань та інструктажів з питань безпеки праці сприяє створенню сприятливого робочого середовища, яке відповідає сучасним вимогам охорони праці.

4.3 Пожежна безпека

Пожежна безпека є невід'ємною частиною охорони праці та охоплює комплекс організаційних заходів і технічних засобів, спрямованих на запобігання пожежам, зменшення їх наслідків і забезпечення безпеки людей. Система протипожежного захисту включає різноманітні методи і засоби, які забезпечують раннє виявлення пожеж, своєчасне оповіщення про загрозу, ефективне гасіння пожеж та евакуацію людей.

Основним завданням системи протипожежного захисту є запобігання впливу на людей небезпечних чинників пожежі, таких як висока температура, дим, токсичні продукти горіння та зниження концентрації кисню. Ці чинники можуть призвести до травм, отруєнь або загибелі людей, а також до значних матеріальних збитків. У контексті розробки системи керування Телеграм-ботом для організації робочого часу студентів та викладачів, особлива увага приділяється забезпеченню пожежної безпеки в офісних та навчальних приміщеннях.

Приміщення, в яких здійснюється робота з комп'ютерною технікою, зазвичай відносяться до категорії пожежної безпеки "В", що передбачає середню пожежну небезпеку. У таких приміщеннях повинні бути встановлені первинні засоби гасіння пожеж, такі як вогнегасники, пожежні крани, а також системи автоматичної пожежної сигналізації та пожежного водопостачання. Відповідно до Правил улаштування електроустановок (ПУЕ), приміщення повинні мати заземлення електрообладнання та забезпечувати захист від ураження електричним струмом.

Первинні засоби гасіння пожеж включають вогнегасники, які повинні бути розташовані в доступних місцях, та пожежні крани, підключені до системи водопостачання. Вогнегасники повинні відповідати типу пожежної небезпеки приміщення, наприклад, для гасіння електричних пожеж використовуються вогнегасники з вуглекислотою або порошкові вогнегасники. Крім того,

необхідно регулярно перевіряти справність та готовність до використання всіх засобів пожежогасіння.

Протипожежна сигналізація є важливим елементом системи пожежної безпеки. Вона включає в себе датчики диму, температури та полум'я, які забезпечують своєчасне виявлення пожежі та автоматичне оповіщення про небезпеку. Сигналізація повинна бути підключена до централізованої системи оповіщення, яка інформує працівників про необхідність евакуації та викликає пожежну охорону. Система оповіщення повинна бути дубльована звуковими та світловими сигналами, щоб забезпечити максимальну ефективність у випадку пожежі.

Організаційні заходи запобігання пожежам включають регулярне проведення інструктажів з пожежної безпеки, тренувань з евакуації, а також перевірку знань працівників про дії у випадку пожежі. Кожен працівник повинен знати місцезнаходження первинних засобів гасіння пожеж, евакуаційних виходів та мати чітке уявлення про свої дії у випадку виникнення пожежі. Для цього необхідно розробити та затвердити план евакуації, який включає шляхи евакуації, місця збору людей та відповідальних осіб.

Технічні заходи запобігання пожежам передбачають використання вогнестійких матеріалів, ізоляцію електропроводки, встановлення систем автоматичного гасіння пожеж, таких як спринклерні системи. Важливим аспектом є забезпечення відповідності електроустановок вимогам ПУЕ, що включає правильне заземлення та захист від коротких замикань. Необхідно також забезпечити регулярне технічне обслуговування та перевірку всіх систем протипожежного захисту.

Особливу увагу слід приділяти забезпеченню пожежної безпеки при виконанні робіт з підвищеною небезпекою, таких як навантажувально-розвантажувальні роботи, роботи з пестицидами та міндобривами, транспортні роботи тощо. Для таких робіт необхідно розробити спеціальні інструкції з пожежної безпеки, забезпечити додаткові заходи захисту та проведення регулярних інструктажів з працівниками.

Розробка рекомендацій з пожежної профілактики включає створення комплексу організаційних і технічних заходів, спрямованих на забезпечення безпеки людей, запобігання пожежам, обмеження їх поширення та створення умов для успішного гасіння пожеж. Організаційні заходи включають розробку та впровадження політики пожежної безпеки, проведення регулярних аудитів пожежної безпеки, навчання працівників та проведення тренувань з евакуації. Технічні заходи включають встановлення та обслуговування систем протипожежного захисту, використання вогнестійких матеріалів та забезпечення відповідності електроустановок вимогам ПУЕ.

На завершення, можна зробити висновок, що забезпечення пожежної безпеки є комплексним завданням, яке вимагає системного підходу та інтеграції різноманітних заходів. Важливо забезпечити відповідність приміщень та обладнання вимогам пожежної безпеки, регулярно проводити навчання та інструктажі з працівниками, а також здійснювати контроль за виконанням заходів пожежної безпеки. Тільки комплексний підхід до забезпечення пожежної безпеки дозволить запобігти виникненню пожеж, мінімізувати їх наслідки та забезпечити безпеку людей на робочому місці.

ВИСНОВКИ ТА ПРОПОЗИЦІЇ

Проведене дослідження дозволяє зробити наступні висновки. Розроблена система планування та організації робочого часу для студентів та викладачів Львівського національного університету природокористування на основі Телеграм-боту є ефективним інструментом для оптимізації навчального процесу. Використання технології GoogleSheetsAPI дозволяє автоматизувати процес управління розкладом. Реалізовані функції бота значно спрощують життя як студентам, так і викладачам. Тестування системи показало, що вона є зручною у використанні, надійною та дозволяє суттєво підвищити ефективність організації робочого часу.

Основні практичні результати, отримані в ході роботи, полягають у наступному. По-перше, розроблено архітектуру Телеграм-боту з модулями для управління розкладом. По-друге, реалізовано інтеграцію з GoogleSheetsAPI, що дозволяє автоматизувати роботу. По-третє, впроваджено систему розмежування доступу, що забезпечує конфіденційність даних. По-четверте, проведено ретельне тестування системи, що підтвердило її функціональність та ефективність.

Рекомендації щодо використання та ефективності розробленої системи є такими. Telegram-бот може бути впроваджений у навчальних закладах для оптимізації робочого часу студентів та викладачів. Завдяки інтеграції з GoogleSheetsAPI система дозволяє легко синхронізувати дані, уникаючи дублювання інформації. Впровадження бота забезпечить підвищення організованості навчального процесу, а також зменшення кількості прострочених завдань та запізнень. Крім того, система може бути розширена додатковими модулями, наприклад, для аналізу успішності та надання рекомендацій щодо покращення результатів навчання.

Загалом, розроблена система Телеграм-боту є ефективним інструментом для оптимізації робочого часу студентів та викладачів. Її впровадження

дозволить підвищити продуктивність навчальної діяльності та забезпечити більш ефективне управління навчальним процесом.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. ДІБРОВА, І. С. Розробка телеграм-бота та його застосування. Вісник студентського наукового товариства ДонНУ імені Василя Стуса, 2024, 123-127.
2. ЗАМУРУЄВА, Оксана Валеріївна; КРИМУСЬ, Андрій Сергійович; ОЛЬХОВА, Наталія Володимирівна. Об'єктно-орієнтоване програмування в Python: курс лекцій. 2018.
3. УСАТА, О. Ю. СУЧАСНІ МОВИ ПРОГРАМУВАННЯ ТА ІНСТРУМЕНТИ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.
4. Тимочко В.О., Городецький І.М., Березовецький А.П., Мазур І.Б. та ін. Безпека життєдіяльності та охорона праці. Навч. посібник. Львів: Сполом. 2022. 376 с.
5. ДСН 3.3.6.042-99. Санітарні норми мікроклімату виробничих приміщень. URL: https://zakon.rada.gov.ua/rada/show/va042282-99#Text (дата звернення: 20.4.2024).
6. Електробезпека [Текст]: підручник / С. В. Панченко, О. І. Акімов, М. М. Бабаєв та ін. Харків : УкрДУЗТ, 2018. 295 с.
7. Пістун І. П., Березовецький А. П., Тимочко В. О., Городецький І. М. Охорона праці (гігієна праці та виробнича санітарія): навч. посіб. / за ред. І.П.Пістуна. Львів: Тріада плюс, 2017. Ч. І. 620 с.
8. ВАСИЛЬЄВ, Олексій Миколайович. Програмування мовою Python. Bohdan Books, 2022.
9. МОХОНЬКО, Ганна Анатоліївна; ПІМОНОВА, Катерина Анатоліївна. Методологія Scrum в управлінні проєктами на фармацевтичних підприємствах. *Економіка і суспільство*, 2019, 20: 324-331.
10. ШОСТЕНКО, С. С.; ЧАЛА, О. О. Архітектура програмного забезпечення для супроводження автоматизованих систем оповіщення на виробництві. 2022.

11. РУДА, О. А.; МОДЕНОВ, Ю. Б. Методи та моделі тестування програмного забезпечення. Problems of Informatization and Control, 2013, 2.42: 93-98.
12. СМЕТАНА, М. Ю.; БІЛАВКА, В. Б. Взаємодія з користувачем за допомогою магічних фільтрів у Aiogram: вплив Python на ефективність чат-ботів. Зв'язок, 2024, 2: 55-58.
13. Офіційна документація Aiogram. URL: https://aiogram-birdi7.readthedocs.io/en/latest/dispatcher/ (дата звернення : 05.03.2024).
14. HTTP Long Polling. URL: https://medium.com/tech_internals/long-polling-and-sse-7e8ddaffcaa7 (дата звернення : 07.03.2024).
15. Вступ до асинхронного програмування на Python. URL: https://devzone.org.ua/post/vstup-do-asinhronnogo-programuvannya-napython (дата звернення : 09.03.2024).

ДОДАТКИ

ДОДАТОК А
ПРОГРАМНИЙ КОД СИСТЕМИ

apsched.py

```python
from aiogram import Bot
import gspread
import config
import pickle


1 usage  BatOleh
async def auto_update_student_schedule_table(bot: Bot):
    sa = gspread.service_account(filename="credentials.json")  # service account var
    sh = sa.open_by_url(config.settings["GOOGLESHEET_URL"])
    wks = sh.worksheet(config.settings["STUDENT_SCHEDULE_WORKSHEET"])  # worksheet var
    student_schedul
                    Fetching Documentation...
    student_schedule_file_name = "student_schedule.pkl"

    open_file = open(student_schedule_file_name, "wb")
    pickle.dump(student_schedule, open_file)
    open_file.close()
    await bot.send_message(config.settings["ADMIN_ID"], text: f"Автооновлення даних розкладу студентів пройшло успішно!")


1 usage  BatOleh
async def auto_update_teacher_schedule_table(bot: Bot):
    sa = gspread.service_account(filename="credentials.json")  # service account var
    sh = sa.open_by_url(config.settings["GOOGLESHEET_URL"])
    wks = sh.worksheet(config.settings["TEACHERS_SCHEDULE_WORKSHEET"])  # worksheet var
    teacher_schedule = wks.get_all_values()

    teacher_schedule_file_name = "teacher_schedule.pkl"

    open_file = open(teacher_schedule_file_name, "wb")
    pickle.dump(teacher_schedule, open_file)
    open_file.close()
    await bot.send_message(config.settings["ADMIN_ID"], text: f"Автооновлення даних розкладу викладачів пройшло успішно!")

```

Рисунок А.1 – Код файлу apsched.py

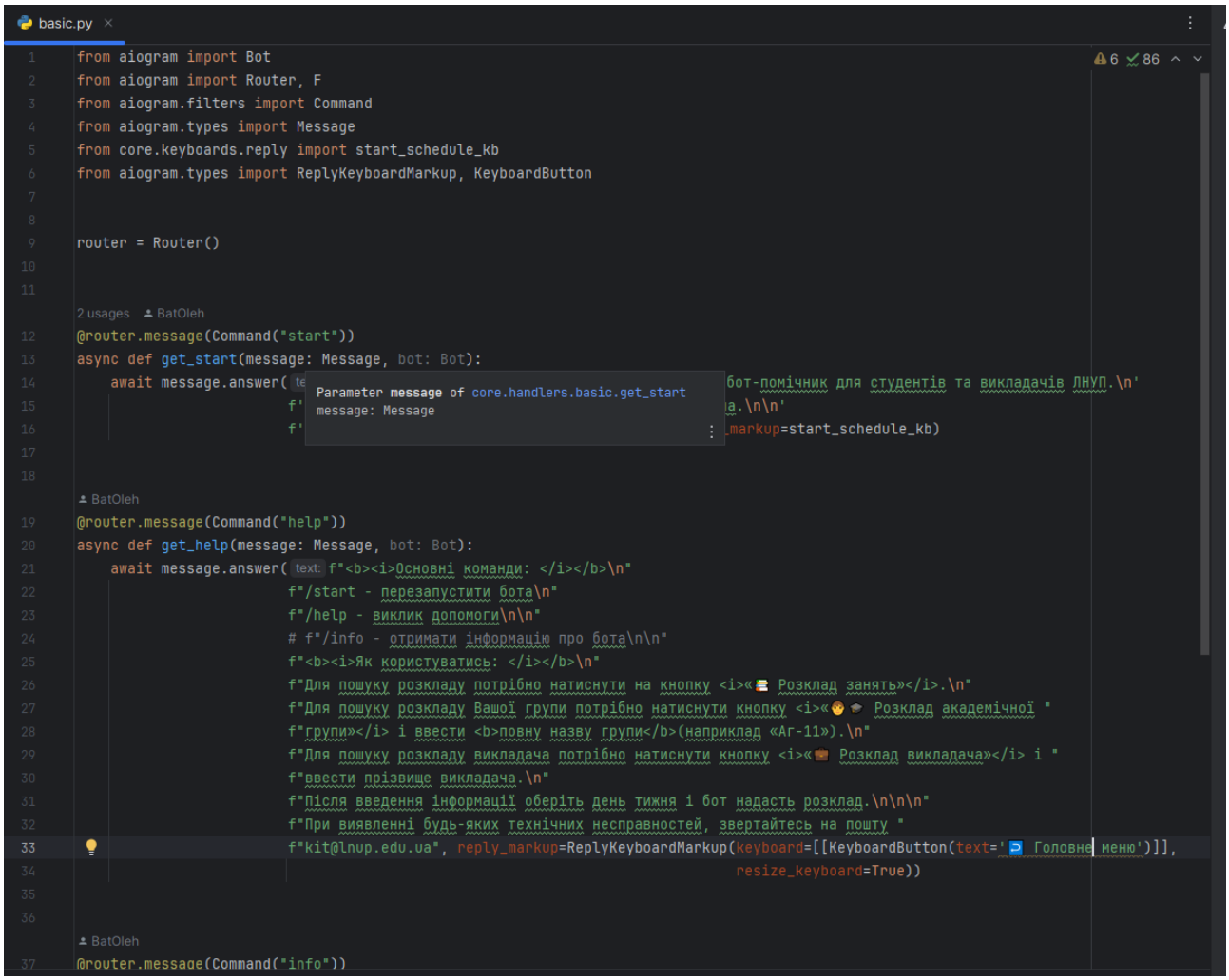


Рисунок А.2 – Код файлу basic.py

```python
from aiogram import Router, F

from aiogram.types import Message, ReplyKeyboardRemove
from typing import Any, Dict
from core.keyboards.reply import choose_type_kb, choose_day_kb, end_of_showSchedule_kb
from core.keyboards.builders import choosing_match_teacher
from aiogram.fsm.context import FSMContext
from core.utils.statesSchedule import StepsSchedule
import gspread
import config
from core.handlers.basic import *
import pickle


router = Router()

OUT_DATA = [str, Any]


 BatOleh
@router.message(F.text.casefold() == "📑 розклад занять")
async def get_schedule(message: Message, state: FSMContext):
    await state.set_state(StepsSchedule.GET_TYPE)
    await message.answer( text: f'☑️ Виберіть варіант розкладу для відображення',
                         reply_markup=choose_type_kb)


 BatOleh
@router.message(StepsSchedule.GET_TYPE, F.text.casefold() == "👨‍🎓 розклад академічної групи")
async def get_type_stud(message: Message, state: FSMContext):
    await state.update_data(type=message.text)
    await state.set_state(StepsSchedule.GET_GROUP_NAME)
    await message.answer( text: f'👤 Введіть назву групи', reply_markup=ReplyKeyboardRemove())


 BatOleh
@router.message(StepsSchedule.GET_GROUP_NAME)
async def get_group_name(message: Message, state: FSMContext):
    await state.update_data(group_name=message.text)
    data = await state.get_data()
```

Рисунок А.3 – Код файлу schedule.py

schedule.py

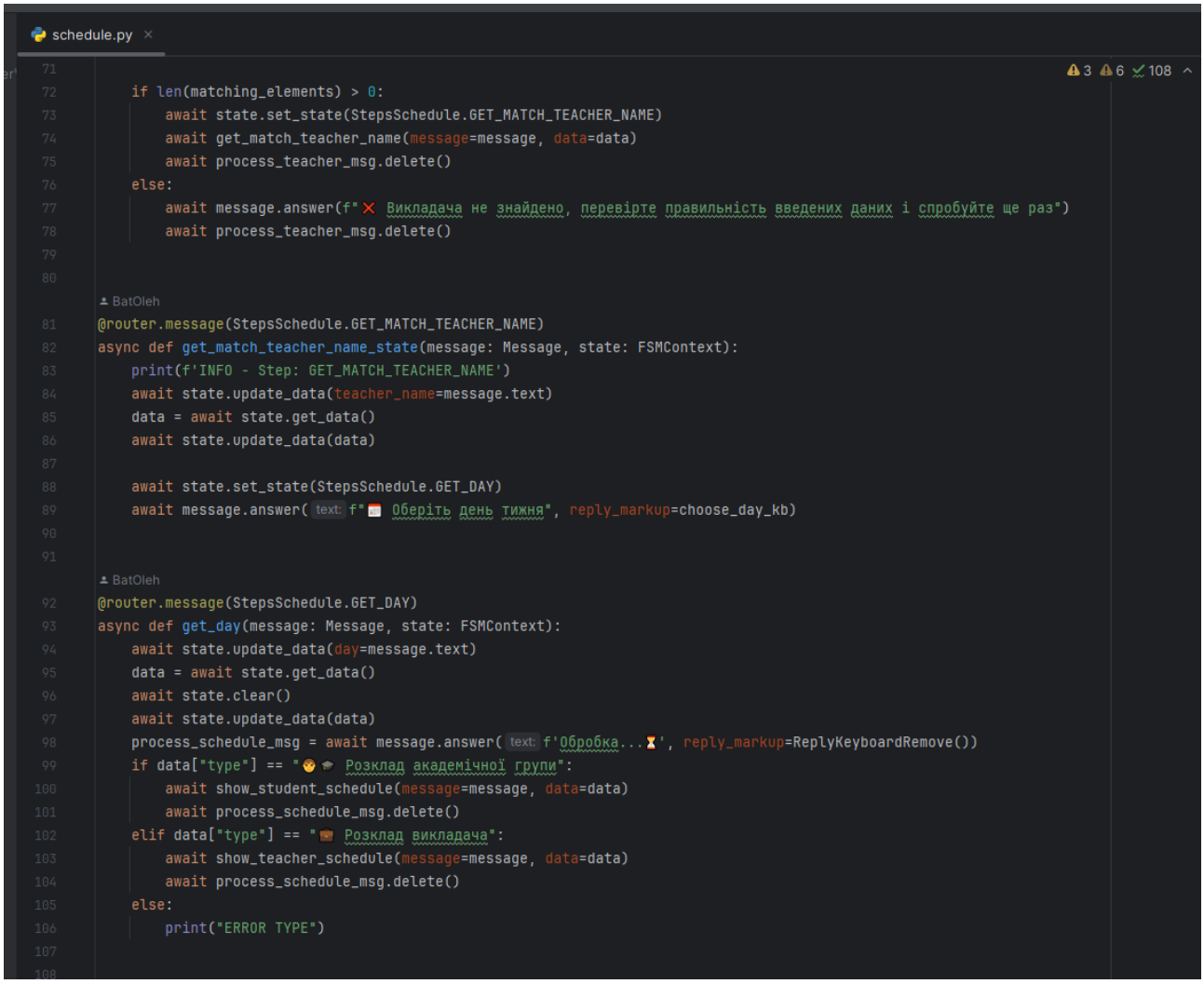
```python
@router.message(StepsSchedule.GET_GROUP_NAME)
async def get_group_name(message: Message, state: FSMContext):
    await state.update_data(group_name=message.text)
    data = await state.get_data()
    student_schedule_file_name = "student_schedule.pkl"
    open_student_file = open(student_schedule_file_name, "rb")
    student_schedule_table = pickle.load(open_student_file)
    open_student_file.close()

    students_groups = student_schedule_table[config.settings["GROUP_ROW"] - 1]
    students_groups = [x.lower() for x in students_groups]
    if str(data["group_name"]).lower() in students_groups:
        await state.set_state(StepsSchedule.GET_DAY)
        await message.answer( text: f' Оберіть день тижня ', reply_markup=choose_day_kb)
    else:
        await message.answer(f" Такої групи не знайдено, перевірте правильність введених даних і спробуйте ще раз")


BatOleh
@router.message(StepsSchedule.GET_TYPE, F.text.casefold().in_(" розклад викладача"))
async def get_type_teach(message: Message, state: FSMContext):
    await state.update_data(type=message.text)
    await state.set_state(StepsSchedule.GET_TEACHER_NAME)
    await message.answer( text: f' Введіть прізвище викладача', reply_markup=ReplyKeyboardRemove())


BatOleh
@router.message(StepsSchedule.GET_TEACHER_NAME)
async def get_teacher_name(message: Message, state: FSMContext):
    await state.update_data(search_teacher_name=message.text)
    data = await state.get_data()
    process_teacher_msg = await message.answer(f'Зачекайте...⧗\nТриває пошук викладача.')
    teacher_schedule_file_name = "teacher_schedule.pkl"
    open_teacher_file = open(teacher_schedule_file_name, "rb")
    teacher_schedule_table = pickle.load(open_teacher_file)
    open_teacher_file.close()
    search_teacher = data["search_teacher_name"]
    teachers_name_list = teacher_schedule_table[config.settings["TEACHERS_ROW"] - 1]
    matching_elements = [element for element in teachers_name_list if search_teacher.lower() in element.lower()]
```

Рисунок А.4 - Код файлу schedule.py



```python
    if len(matching_elements) > 0:
        await state.set_state(StepsSchedule.GET_MATCH_TEACHER_NAME)
        await get_match_teacher_name(message=message, data=data)
        await process_teacher_msg.delete()
    else:
        await message.answer(f"❌ Викладача не знайдено, перевірте правильність введених даних і спробуйте ще раз")
        await process_teacher_msg.delete()


BatOleh
@router.message(StepsSchedule.GET_MATCH_TEACHER_NAME)
async def get_match_teacher_name_state(message: Message, state: FSMContext):
    print(f'INFO - Step: GET_MATCH_TEACHER_NAME')
    await state.update_data(teacher_name=message.text)
    data = await state.get_data()
    await state.update_data(data)

    await state.set_state(StepsSchedule.GET_DAY)
    await message.answer( text: f"🗓 Оберіть день тижня", reply_markup=choose_day_kb)


BatOleh
@router.message(StepsSchedule.GET_DAY)
async def get_day(message: Message, state: FSMContext):
    await state.update_data(day=message.text)
    data = await state.get_data()
    await state.clear()
    await state.update_data(data)
    process_schedule_msg = await message.answer( text: f'Обробка...⏳', reply_markup=ReplyKeyboardRemove())
    if data["type"] == "👨‍🎓 Розклад академічної групи":
        await show_student_schedule(message=message, data=data)
        await process_schedule_msg.delete()
    elif data["type"] == "💼 Розклад викладача":
        await show_teacher_schedule(message=message, data=data)
        await process_schedule_msg.delete()
    else:
        print("ERROR TYPE")
```

Рисунок А.5 - Код файлу schedule.py

builders.py

```python
from aiogram.types import ReplyKeyboardMarkup, KeyboardButton, KeyboardButtonPolli
from aiogram.utils.keyboard import ReplyKeyboardBuilder


2 usages  BatOleh
def choosing_match_teacher(text: str | list, count: int):
    builder = ReplyKeyboardBuilder()

    if isinstance(text, str):
        text = [text]

    [builder.button(text=txt) for txt in text]

    builder.adjust(1)

    return builder.as_markup(resize_keyboard=True, one_time_keyboard=True)

```

Рисунок А.6 – Код файлу builders.py

reply.py

```python
from aiogram.types import ReplyKeyboardMarkup, KeyboardButton, KeyboardButtonPollT

choose_type_kb = ReplyKeyboardMarkup(keyboard=[
    [
        KeyboardButton(text='👨‍🎓 Розклад академічної групи')

    ],
    [
        KeyboardButton(text='💼 Розклад викладача')
    ]
], resize_keyboard=True)

choose_day_kb = ReplyKeyboardMarkup(keyboard=[
    [
        KeyboardButton(text='Понеділок'),
        KeyboardButton(text='Вівторок'),
        KeyboardButton(text='Середа'),
    ],

    [
        KeyboardButton(text='Четвер'),
        KeyboardButton(text="П'ятниця"),
    ]
], resize_keyboard=True)


start_schedule_kb = ReplyKeyboardMarkup(keyboard=[
    [
        KeyboardButton(text='📚 Розклад занять')
    ]
], resize_keyboard=True, )

end_of_showSchedule_kb = ReplyKeyboardMarkup(keyboard=[
    [
        KeyboardButton(text='📅 Обрати інший день')
    ],
    [
        KeyboardButton(text='↩️ Головне меню')
    ]
], resize_keyboard=True)
```

Рисунок А.7 – Код файлу reply.py

admin.py

```python
import ...

router = Router()

BatOleh
@router.message(Command("admin"), F.from_user.id.in_(config.settings["ADMINS_LIST_IDs"]))
async def admin_login(message: Message):
    await message.answer( text: f"Ви увійшли в режим адміністратора, оберіть дію", reply_markup=admin_panel_kb)

BatOleh
@router.message(F.text.casefold() == 'оновити розклад студентів',
                F.from_user.id.in_(config.settings["ADMINS_LIST_IDs"]))
async def update_student_schedule(message: Message):
    sa = gspread.service_account(filename="credentials.json")  # service account var
    sh = sa.open_by_url(config.settings["GOOGLESHEET_URL"])
    wks = sh.worksheet(config.settings["STUDENT_SCHEDULE_WORKSHEET"])  # worksheet var
    student_schedule = wks.get_all_values()

    student_schedule_file_name = "student_schedule.pkl"

    open_file = open(student_schedule_file_name, "wb")
    pickle.dump(student_schedule, open_file)
    open_file.close()
    await message.answer(f"Дані розкладу студентів оновлено!")

BatOleh
@router.message(F.text.casefold() == 'оновити розклад викладачів',
                F.from_user.id.in_(config.settings["ADMINS_LIST_IDs"]))
async def update_teacher_schedule(message: Message):
    sa = gspread.service_account(filename="credentials.json")  # service account var
    sh = sa.open_by_url(config.settings["GOOGLESHEET_URL"])
    wks = sh.worksheet(config.settings["TEACHERS_SCHEDULE_WORKSHEET"])  # worksheet var
    teacher_schedule = wks.get_all_values()

    teacher_schedule_file_name = "teacher_schedule.pkl"

    open_file = open(teacher_schedule_file_name, "wb")
    pickle.dump(teacher_schedule, open_file)
    open_file.close()
    await message.answer(f"Дані розкладу викладачів оновлено!")
```

Рисунок А.8 – Код файлу reply.py

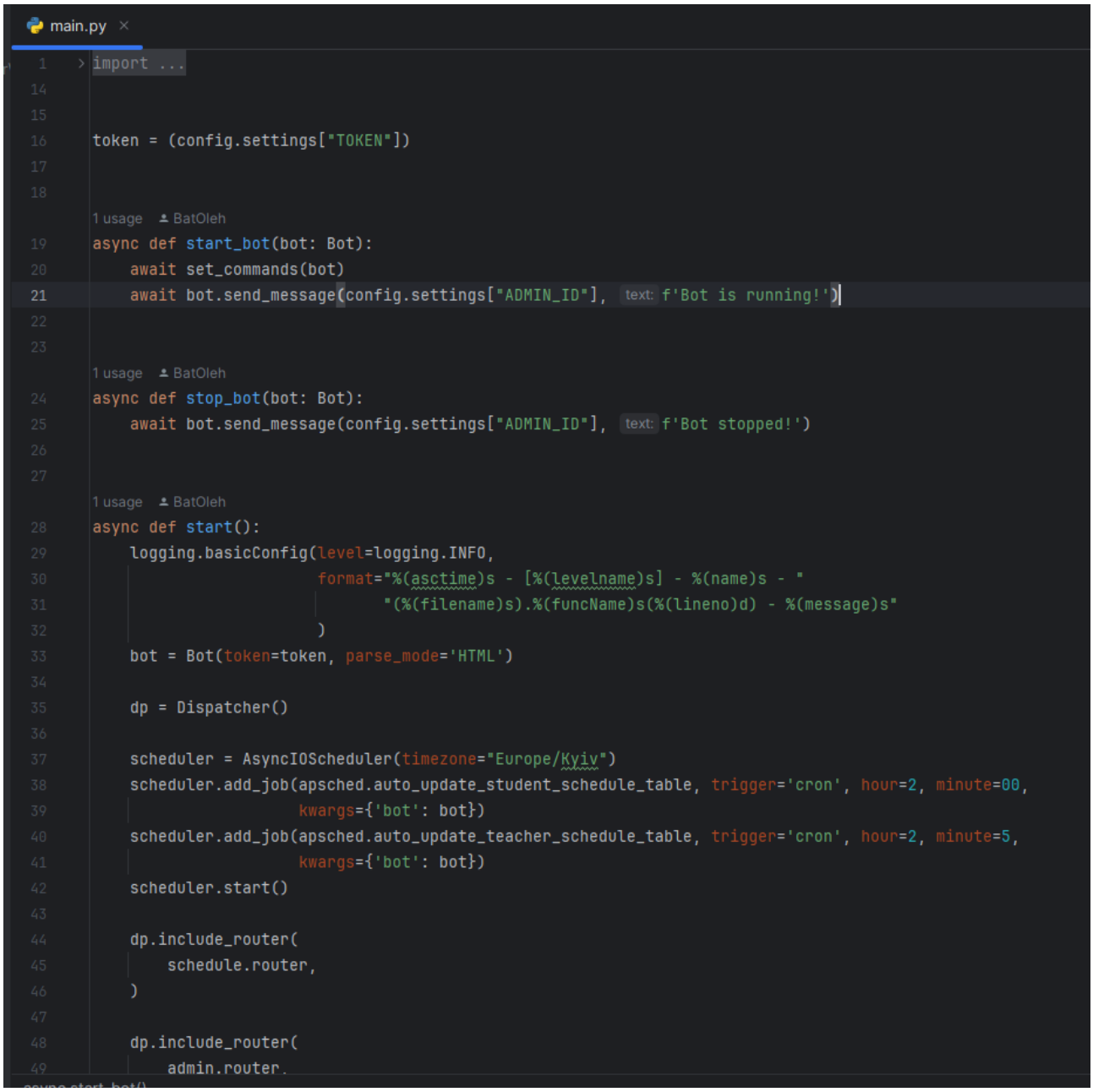
commands.py

```python
import ...


2 usages  BatOleh
async def set_commands(bot: Bot):
    commands = [
        BotCommand(
            command='start',
            description='Початок роботи'
        ),
        BotCommand(
            command='help',
            description='Допомога'
        ),
        # BotCommand(
        #     command='info',
        #     description='Інформація'
        # ),
    ]
    await bot.set_my_commands(commands, BotCommandScopeDefault())

```

Рисунок А.9 – Код файлу commands.py



```python
import ...


token = (config.settings["TOKEN"])


1 usage  BatOleh
async def start_bot(bot: Bot):
    await set_commands(bot)
    await bot.send_message(config.settings["ADMIN_ID"], text: f'Bot is running!')


1 usage  BatOleh
async def stop_bot(bot: Bot):
    await bot.send_message(config.settings["ADMIN_ID"], text: f'Bot stopped!')


1 usage  BatOleh
async def start():
    logging.basicConfig(level=logging.INFO,
                        format="%(asctime)s - [%(levelname)s] - %(name)s - "
                               "(%(filename)s).%(funcName)s(%(lineno)d) - %(message)s"
                        )
    bot = Bot(token=token, parse_mode='HTML')

    dp = Dispatcher()

    scheduler = AsyncIOScheduler(timezone="Europe/Kyiv")
    scheduler.add_job(apsched.auto_update_student_schedule_table, trigger='cron', hour=2, minute=00,
                      kwargs={'bot': bot})
    scheduler.add_job(apsched.auto_update_teacher_schedule_table, trigger='cron', hour=2, minute=5,
                      kwargs={'bot': bot})
    scheduler.start()

    dp.include_router(
        schedule.router,
    )

    dp.include_router(
        admin.router,
```

Рисунок А.10 – Код файлу main.py