Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

РОЗРОБЛЕННЯ БОТА В TELEGRAM ДЛЯ АВТОМАТИЗАЦІЇ ПРОЦЕСІВ ПЛАНУВАННЯ ТА ПУБЛІКАЦІЇ КОНТЕНТУ В SMM

Виконав:
здобувач 4 року навчання,
групи ІТІНФ-22-3
Дам-Васильєва Ч. А.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник проф. Шафроненко А. Ю.
(посада, прізвище, ініціали)

Допускається до захисту

Завідувач кафедри інформатики _____ Кобилін О. А.
(підпис) (прізвище, ініціали)

2026 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту

Кафедра Інформатики

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Дам-Васильєвій Чанг Анжеліці
(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення бота в Telegram для автоматизації процесів планування та публікації контенту в SMM

затверджена наказом університету від 26 травня 2026 року № 435Ст
2. Термін подання здобувачем роботи до екзаменаційної комісії 22 червня 2026 р.
3. Вихідні дані до роботи: науково-методична та науково-технічна література з питань цифрового маркетингу та автоматизації SMM-процесів, документація Telegram Bot API, документація мови програмування Go, документація бібліотек і фреймворків, використаних під час розроблення Telegram-бота, матеріали наукових конференцій, електронні інформаційні ресурси мережі Інтернет, результати аналізу сучасних програмних засобів планування та публікації контенту в соціальних мережах.

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд предметної області автоматизації SMM-процесів.
2. Алгоритми та методи автоматизації SMM-процесів.
3. Програмна реалізація Telegram-бота

5. Перелік графічного матеріалу: схема загальної логіки роботи Telegram-бота, схема алгоритму формування контент-плану, схема алгоритму організації нагадувань, схема взаємодії користувача із системою, схема архітектури програмного продукту, фрагменти програмної реалізації основних модулів, скріншоти створення публікацій, генерації ідей для контенту, експорту постів та інших функцій Telegram-бота.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	13.04.2026	
2	Аналіз завдання, підбір літератури	14.04.26-16.04.26	
3	Аналіз літератури з проблеми, що вирішується	17.04.26-19.04.26	
4	Аналіз існуючих методів розпізнавання	20.04.26-22.04.26	
5	Формування кроків вирішення проблеми	23.04.26-05.05.26	
6	Програмна реалізація	26.04.26-20.05.26	
7	Аналіз отриманих результатів	21.05.26-04.06.26	
8	Оформлення пояснювальної записки	05.06.26-09.06.26	
9	Перевірка на нормоконтроль	12.06.26-17.06.26	
10	Перевірка на плагіат	17.06.26-18.06.26	
11	Рецензування	18.06.26-30.06.26	
12	Підготовка презентації та доповіді	20.06.26-30.06.26	
13	Занесення роботи в електронний архів	01.07.2026	
14	Попередній захист кваліфікаційної роботи	01.07.2026	

Дата видачі завдання 13 квітня 2026 р.

Здобувач _____
(підпис)

Керівник роботи _____ проф. Шафроненко А. Ю.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка: 83 с., 5 табл., 11 рис., 27 джерел.

АВТОМАТИЗАЦІЯ, КОНТЕНТ-ПЛАН, СОЦІАЛЬНІ МЕРЕЖІ, ЧАТ-БОТ, GO, SMM, TELEGRAM-БОТ.

Об’єкт роботи - процес планування та публікації контенту в соціальних мережах.

Мета роботи - розроблення Telegram-бота для автоматизації планування та публікації контенту.

Під час роботи використано: методи аналізу предметної області, порівняльний аналіз програмних засобів, алгоритмізація, GO, Telegram Bot API, aiogram.

Практична цінність полягає у створенні Telegram-бота для ведення контент-плану та організації нагадувань.

Проаналізовано SMM-процеси та існуючі засоби автоматизації. Визначено вимоги до системи, розроблено алгоритми роботи та реалізовано Telegram-бот для керування контентом.

Область застосування: цифровий маркетинг, SMM-діяльність підприємств, планування та організація контенту в соціальних мережах, автоматизація процесів підготовки та публікації контенту, підтримка роботи SMM-фахівців, маркетологів, контент-менеджерів і власників малого бізнесу

AUTOMATION, CHATBOT, CONTENT PLAN, GO, SOCIAL MEDIA, SMM, TELEGRAM BOT.

The object of the study is the process of planning and publishing content on social media platforms.

The purpose of the work is to develop a Telegram bot for automating content planning and publishing.

The following methods and technologies were used in the study: domain analysis methods, comparative analysis of software tools, algorithm design, Go programming language, Telegram Bot API, and aiogram.

The practical value of the work lies in the development of a Telegram bot for managing a content plan and organizing publication reminders.

SMM processes and existing automation tools were analyzed. System requirements were defined, operational algorithms were developed, and a Telegram bot for content management was implemented.

Applications: digital marketing, enterprise SMM activities, planning and organization of social media content, automation of content preparation and publishing processes, support for SMM specialists, marketers, content managers, and small business owners.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

Контент-план – структурований перелік запланованих публікацій із зазначенням часу та параметрів розміщення

Чат-бот – програмний засіб для автоматизованого обміну повідомленнями з користувачем

AI – Artificial Intelligence (штучний інтелект)

API – Application Programming Interface (програмний інтерфейс застосунку)

JSON – JavaScript Object Notation (текстовий формат обміну даними)

Go – мова програмування загального призначення

SMM – Social Media Marketing (маркетинг у соціальних мережах)

SQLite – вбудована реляційна система керування базами даних

Telegram Bot API – програмний інтерфейс для створення та керування Telegram-ботами

Telegram-бот – програмний агент у месенджері Telegram, що забезпечує автоматизовану взаємодію з користувачем

ВСТУП

У сучасних умовах розвитку інформаційних технологій та цифрового середовища соціальні мережі відіграють важливу роль у просуванні товарів, послуг і формуванні іміджу бренду. Соціальні платформи стали основним каналом комунікації між компаніями та споживачами, що зумовило активний розвиток напряму SMM (Social Media Marketing). Використання SMM дозволяє не лише інформувати аудиторію, але й формувати довготривалі відносини з клієнтами, підвищувати рівень довіри та стимулювати продажі.

Однією з ключових складових ефективної SMM-діяльності є створення та регулярна публікація контенту. Контент виступає основним засобом взаємодії з аудиторією, тому його якість, актуальність та своєчасність мають вирішальне значення. Водночас процес планування та публікації контенту є досить складними та потребують значних витрат часу. Вони включають формування контент-плану, підготовку матеріалів, визначення оптимального часу публікації та контроль виконання.

У практичній діяльності SMM-спеціалісти часто стикаються з проблемами, пов’язаними з необхідністю виконання великої кількості рутинних операцій, відсутністю зручних інструментів для планування та ризиком пропуску запланованих публікацій. Це знижує ефективність роботи та може негативно впливати на результати просування.

У зв’язку з цим актуальним є використання засобів автоматизації, які дозволяють оптимізувати процеси планування та публікації контенту. Одним із перспективних напрямів є застосування чат-ботів, зокрема Telegram-ботів. Вони забезпечують зручну взаємодію з користувачем, можливість організації нагадувань, збереження інформації та виконання інших функцій без необхідності використання складного програмного забезпечення.

1 ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ АВТОМАТИЗАЦІЇ SMM-ПРОЦЕСІВ

1.1 Характеристика SMM як предметної області

У сучасному інформаційному суспільстві соціальні мережі займають провідне місце серед засобів комунікації, обміну інформацією та просування товарів і послуг. Їх широке поширення та доступність зумовили формування нового напряму маркетингової діяльності - маркетингу в соціальних мережах (Social Media Marketing, SMM), який є невід'ємною складовою цифрового маркетингу [1].

SMM як предметна область охоплює сукупність методів, інструментів і технологій, спрямованих на просування брендів, продуктів або послуг за допомогою соціальних платформ. Важливою перевагою SMM є можливість персоналізації маркетингових повідомлень. Використовуючи дані про інтереси, поведінку та вподобання користувачів, компанії можуть адаптувати контент до потреб конкретних груп споживачів. Такий підхід дозволяє підвищити ефективність комунікацій та забезпечити більш високий рівень залученості аудиторії. Персоналізований контент сприймається користувачами як більш корисний і релевантний, що позитивно впливає на результати маркетингових кампаній [1].

Сучасний розвиток цифрових технологій суттєво змінив підходи до маркетингової діяльності підприємств. Якщо традиційний маркетинг переважно орієнтувався на односторонню передачу інформації від компанії до споживача, то SMM базується на принципах інтерактивності, відкритості та постійної комунікації. Соціальні мережі створюють середовище, у якому користувачі можуть не лише споживати інформацію, але й активно брати участь у її створенні та поширенні. У результаті аудиторія стає важливим

учасником комунікаційного процесу, а її думка безпосередньо впливає на формування репутації бренду [2].

Основною метою SMM є встановлення ефективної взаємодії з цільовою аудиторією, формування позитивного іміджу компанії, підвищення рівня довіри споживачів, а також стимулювання попиту та продажів [2].

Однією з ключових особливостей SMM є інтерактивний характер комунікації. На відміну від традиційних маркетингових каналів, соціальні мережі забезпечують двосторонній зв’язок між брендом і користувачем. Це дозволяє не лише поширювати інформацію, але й отримувати зворотний зв’язок, аналізувати реакцію аудиторії та адаптувати стратегію просування відповідно до її потреб [3].

Важливою характеристикою SMM є його динамічність. Соціальні платформи постійно змінюють алгоритми відображення контенту, впроваджують нові формати та функціональні можливості. У зв’язку з цим SMM-спеціалісти повинні постійно відстежувати ці зміни та оперативно адаптувати свої підходи до створення і розповсюдження контенту [4].

Центральним елементом SMM є контент, який виступає основним засобом комунікації з аудиторією. Контент може бути представлений у різних формах: текстові повідомлення, зображення, відео, сторіс, інтерактивні елементи тощо. Вибір формату залежить від специфіки платформи, характеристик цільової аудиторії та поставлених маркетингових цілей [5].

У межах SMM контент виконує декілька функцій. По-перше, він інформує користувачів про діяльність компанії, її продукти або послуги. По-друге, контент формує емоційний зв’язок із аудиторією, що сприяє підвищенню лояльності. По-третє, він стимулює активність користувачів, що проявляється у вигляді лайків, коментарів, поширень та інших форм взаємодії [6].

Особливого значення в сучасному SMM набуває візуальний контент. Дослідження показують, що повідомлення, які містять зображення або

відеоматеріали, привертають значно більше уваги користувачів порівняно зі звичайними текстовими публікаціями. Це пояснюється особливостями сприйняття інформації людиною, оскільки візуальні образи обробляються швидше та викликають сильніший емоційний відгук [5].

Крім того, поширення коротких відеоформатів суттєво змінило підходи до створення контенту. Такі формати дозволяють швидко донести необхідну інформацію до аудиторії та забезпечити високий рівень взаємодії. У зв’язку з цим SMM-фахівці повинні враховувати особливості різних типів контенту та обирати найбільш ефективні формати для досягнення поставлених цілей [6].

Ефективність SMM значною мірою залежить від правильного визначення цільової аудиторії. Для цього здійснюється аналіз демографічних, соціальних і поведінкових характеристик користувачів. Результати такого аналізу використовуються для створення релевантного контенту, який відповідає інтересам і потребам аудиторії [7].

Ще одним важливим аспектом SMM є аналітика. Сучасні соціальні платформи надають широкий спектр інструментів для оцінки ефективності контенту. До основних показників належать охоплення, кількість переглядів, рівень залученості, кількість підписників та інші. Аналіз цих даних дозволяє оцінити результативність SMM-стратегії та внести необхідні корективи [8].

Не менш важливим напрямом розвитку SMM є використання автоматизованих інструментів управління контентом. Зі збільшенням кількості соціальних платформ і обсягів інформації ручне керування публікаціями стає все менш ефективним. Саме тому сучасні компанії активно впроваджують спеціалізовані програмні рішення, які дозволяють автоматизувати процеси планування, підготовки та контролю контенту [9].

Автоматизація забезпечує можливість централізованого управління інформаційними потоками, зменшує ризик виникнення помилок та дозволяє більш ефективно використовувати робочий час спеціалістів. Особливо актуальним це є для малого бізнесу та індивідуальних підприємців, які часто не мають окремих команд для ведення соціальних мереж. Використання

програмних засобів автоматизації дозволяє підтримувати регулярність публікацій та забезпечувати стабільну присутність бренду в інформаційному просторі [10].

Додатково слід зазначити, що розвиток месенджерів відкрив нові можливості для організації взаємодії з користувачами. Такі платформи, як Telegram, поступово перетворюються не лише на засіб спілкування, але й на повноцінне середовище для реалізації інформаційних сервісів. Завдяки підтримці ботів користувачі отримують можливість працювати з інформацією без встановлення додаткових програм, що робить такі рішення доступними та зручними у використанні [11].

З точки зору організації роботи, SMM передбачає системний підхід до створення та публікації контенту. Це включає планування публікацій, визначення їх частоти, часу розміщення та тематичного наповнення. Для цього використовується контент-план, який дозволяє структуровано організувати роботу та забезпечити регулярність публікацій [9].

Однак на практиці реалізація SMM-процесів пов’язана з рядом труднощів. Зокрема, це значні витрати часу на підготовку контенту, необхідність постійного контролю за публікаціями, а також потреба у використанні різних інструментів для виконання окремих задач. Усе це ускладнює роботу та знижує її ефективність [10].

У зв’язку з цим зростає актуальність автоматизації SMM-процесів. Використання програмних засобів дозволяє оптимізувати рутинні операції, зменшити витрати часу та підвищити ефективність роботи. Особливий інтерес у цьому контексті становлять чат-боти, які можуть виконувати функції планування, нагадування та взаємодії з користувачем [1].

Таким чином, SMM як предметна область характеризується складністю, багатофункціональністю та високим рівнем динамічності. Вона поєднує креативні, аналітичні та технологічні аспекти, що обумовлює необхідність використання сучасних інструментів автоматизації. Це створює

передумови для розробки програмних рішень, зокрема Telegram-ботів, які дозволяють оптимізувати процеси планування та публікації контенту.

1.2 Процеси планування та публікації контенту в SMM

Процеси планування та публікації контенту є основою ефективної діяльності в сфері соціальних медіа та визначають якість взаємодії між брендом і його аудиторією. У сучасних умовах високої конкуренції в цифровому середовищі недостатньо лише створювати контент - важливо робити це системно, регулярно та з урахуванням поведінкових особливостей користувачів [1]. Саме тому процеси планування та публікації контенту розглядаються як структурована послідовність дій, спрямованих на досягнення конкретних маркетингових цілей.

Планування контенту починається з визначення стратегічних цілей SMM-діяльності. До таких цілей належать підвищення впізнаваності бренду, збільшення аудиторії, залучення користувачів, стимулювання продажів або підтримка комунікації з клієнтами. Чітке формулювання цілей дозволяє визначити загальний напрям створення контенту та сформувати основу для подальших дій [7].

Наступним етапом є аналіз цільової аудиторії. Цей процес передбачає дослідження характеристик користувачів, їхніх інтересів, поведінкових моделей та активності в соціальних мережах. Результати аналізу дозволяють адаптувати контент до потреб аудиторії, що значно підвищує його ефективність [2]. Наприклад, для молодої аудиторії доцільно використовувати короткі відео та інтерактивні формати, тоді як для більш дорослої-інформативні текстові матеріали.

Аналіз цільової аудиторії є безперервним процесом, оскільки інтереси користувачів, їхня поведінка та інформаційні потреби можуть змінюватися під впливом зовнішніх факторів. Для отримання актуальних даних

використовуються результати аналітики соціальних мереж, статистика переглядів, взаємодій та переходів за посиланнями. Отримана інформація дозволяє визначити найбільш популярні теми, формати контенту та оптимальний стиль комунікації з аудиторією [3].

Крім того, важливим елементом є сегментація аудиторії. Розподіл користувачів на окремі групи за віком, професією, інтересами або іншими характеристиками дозволяє створювати більш персоналізований контент. Такий підхід сприяє підвищенню ефективності комунікації та забезпечує більш високий рівень залученості користувачів до взаємодії з брендом [2].

На основі отриманих даних формується контент-стратегія, яка визначає основні напрями створення контенту. Контент-стратегія включає вибір тематики, форматів, стилю подачі інформації, а також визначення частоти публікацій. Важливим є баланс між різними типами контенту, такими як інформаційний, розважальний, рекламний та навчальний, що дозволяє підтримувати інтерес аудиторії та уникати одноманітності [6].

Ключовим інструментом реалізації контент-стратегії є контент-план. Контент-план являє собою структурований перелік запланованих публікацій на певний період часу. Він містить інформацію про дату та час публікації, тему, формат контенту, текст повідомлення, а також додаткові елементи, такі як хештеги або посилання. Використання контент-плану дозволяє систематизувати роботу, забезпечити регулярність публікацій та уникнути хаотичного розміщення контенту [9].

Контент-план виконує не лише організаційну, але й стратегічну функцію. Його використання дозволяє забезпечити рівномірний розподіл контенту за темами та форматами, уникнути дублювання матеріалів і підтримувати систематичність інформаційної діяльності. У практиці SMM контент-плани можуть формуватися на тиждень, місяць або навіть квартал залежно від особливостей діяльності організації та масштабів маркетингових кампаній [9].

При формуванні контент-плану важливо враховувати сезонність, святкові події, професійні заходи та інші фактори, які можуть впливати на інтереси аудиторії. Завдяки цьому компанія отримує можливість своєчасно реагувати на актуальні інформаційні приводи та підвищувати ефективність взаємодії з користувачами [7].

У сучасних умовах контент-план також використовується як інструмент координації роботи команди. Він дозволяє розподіляти обов'язки між спеціалістами, контролювати виконання завдань та забезпечувати узгодженість маркетингових активностей у різних каналах комунікації [6].

Після етапу планування відбувається створення контенту. Цей процес включає генерацію ідей, підготовку текстових матеріалів, створення зображень або відео, а також їх редагування. Створення контенту є одним із найбільш трудомістких етапів, оскільки потребує як креативного підходу, так і технічних навичок [5]. Крім того, необхідно враховувати специфіку кожної соціальної платформи, що впливає на формат і спосіб подачі інформації.

Після підготовки контенту здійснюється його публікація. Важливим аспектом цього етапу є вибір оптимального часу розміщення. Активність користувачів у соціальних мережах залежить від багатьох факторів, зокрема часу доби та дня тижня, тому правильний вибір часу публікації може суттєво вплинути на охоплення та рівень залученості [8].

У сучасній практиці широко використовується відкладена публікація контенту. Вона передбачає попереднє створення та планування публікацій із подальшим автоматичним розміщенням у визначений час. Такий підхід дозволяє оптимізувати робочий процес, зменшити навантаження на користувача та забезпечити стабільність SMM-активності [4].

Після публікації контенту здійснюється моніторинг його ефективності. Це включає аналіз таких показників, як кількість переглядів, лайків, коментарів, поширень та інших форм взаємодії. Отримані результати дозволяють оцінити ефективність контенту та внести зміни до стратегії [3].

Моніторинг результативності контенту є необхідною умовою ефективного управління SMM-діяльністю. Аналіз показників дозволяє оцінити не лише популярність окремих публікацій, але й загальну ефективність обраної контент-стратегії. На основі отриманих результатів визначаються сильні та слабкі сторони інформаційної діяльності, що створює основу для прийняття подальших управлінських рішень [8].

Особливого значення набуває аналіз показників залученості аудиторії, оскільки саме вони відображають рівень взаємодії користувачів із контентом. Високий рівень залученості свідчить про актуальність і цінність опублікованих матеріалів для аудиторії. Натомість низькі показники можуть вказувати на необхідність перегляду тематики, формату або часу публікації контенту [3].

Результати аналітики використовуються для коригування контент-плану, визначення найбільш ефективних форматів та оптимізації процесів публікації. Таким чином забезпечується безперервне вдосконалення SMM-діяльності та підвищення її результативності [1].

Окремим етапом є взаємодія з аудиторією. Користувачі можуть залишати коментарі, ставити запитання або надсилати повідомлення, на які необхідно оперативно реагувати. Це сприяє формуванню довіри та підвищенню рівня залученості [2].

Незважаючи на структурованість процесів планування та публікації контенту, їх реалізація на практиці пов’язана з рядом труднощів. До основних проблем належать значні витрати часу, необхідність виконання великої кількості рутинних операцій, складність контролю за виконанням контент-плану, а також ризик пропуску запланованих публікацій [1]. Усе це знижує ефективність роботи та створює додаткове навантаження на SMM-спеціаліста.

У зв’язку з цим виникає потреба в автоматизації зазначених процесів. Автоматизація дозволяє оптимізувати виконання рутинних завдань, зменшити ймовірність помилок та забезпечити стабільність роботи. Зокрема,

автоматизовані системи можуть виконувати функції збереження контент-плану, нагадування про публікації, обробки даних та взаємодії з користувачем [11].

Перспективним інструментом автоматизації є використання Telegram-ботів. Вони забезпечують зручний інтерфейс для роботи з контентом, можливість організації нагадувань та інтерактивну взаємодію з користувачем [12]. Завдяки цьому Telegram-бот може виступати ефективним засобом підтримки процесів планування та публікації контенту.

Таким чином, процеси планування та публікації контенту в SMM є складною системою взаємопов’язаних етапів, що потребують системного підходу до організації. Їх автоматизація є важливим напрямом підвищення ефективності SMM-діяльності та створює передумови для розробки спеціалізованих програмних рішень.

1.3 Аналіз існуючих рішень автоматизації SMM-процесів

У сучасних умовах розвитку цифрового маркетингу автоматизація процесів SMM є невід’ємною складовою ефективної роботи з соціальними мережами. Зростання кількості платформ, обсягів контенту та необхідність постійної взаємодії з аудиторією зумовлюють потребу у використанні спеціалізованих програмних засобів, які дозволяють оптимізувати робочі процеси та підвищити їх ефективність [1].

На сьогодні існує значна кількість програмних рішень, які забезпечують автоматизацію SMM-процесів. Більшість із них належать до категорії систем управління соціальними мережами (Social Media Management Tools). До найбільш відомих і широко використовуваних сервісів належать Hootsuite, Buffer, Sprout Social, Later та інші [2].

Сервіс Hootsuite є одним із найбільш популярних інструментів для управління соціальними мережами. Він надає можливість централізовано

керувати декількома акаунтами, планувати та публікувати контент, відстежувати активність аудиторії та аналізувати ефективність публікацій. Однією з ключових переваг цього сервісу є наявність функцій соціального моніторингу, які дозволяють відстежувати згадки бренду та аналізувати поведінку користувачів [13]. Крім того, Hootsuite забезпечує інтеграцію з великою кількістю соціальних платформ та інструментів аналітики.

Важливою перевагою Hootsuite є підтримка роботи з різними соціальними платформами в межах єдиного інтерфейсу. Це дозволяє користувачам централізовано керувати маркетинговими активностями та контролювати процес публікації контенту без необхідності переходу між декількома сервісами. Крім того, система надає інструменти для планування довгострокових кампаній, що особливо актуально для компаній із великою кількістю інформаційних матеріалів [13].

Разом із тим використання Hootsuite пов’язане з певними обмеженнями. Частина функціональних можливостей доступна лише у платних тарифних планах, а для повноцінного використання системи необхідне попереднє навчання користувачів. Це може створювати додаткові труднощі для невеликих організацій та індивідуальних спеціалістів, які не потребують складних інструментів управління соціальними мережами [4].

Іншим популярним інструментом є Buffer, який орієнтований на простоту використання та ефективне планування контенту. Основною функцією цього сервісу є можливість створення черги публікацій та їх автоматичного розміщення у визначений час. Buffer також дозволяє взаємодіяти з аудиторією та використовувати допоміжні інструменти для створення контенту [14]. Завдяки своїй простоті цей сервіс є популярним серед індивідуальних користувачів та невеликих команд.

Sprout Social є більш комплексним рішенням, яке поєднує функції планування, аналітики та управління взаємодією з аудиторією. Особливістю цього сервісу є розширені аналітичні можливості та інструменти для командної роботи. Він дозволяє створювати детальні звіти, аналізувати

поведінку користувачів та оцінювати ефективність маркетингових кампаній [15].

Сервіси Later та Planoly орієнтовані переважно на візуальні соціальні мережі, такі як Instagram. Вони дозволяють створювати візуальні контент-плани, планувати публікації та автоматично їх розміщувати. Основною перевагою таких сервісів є зручний інтерфейс, який дозволяє працювати з контентом у форматі календаря або стрічки [16].

Окремої уваги заслуговують сучасні сервіси автоматизації, орієнтовані на аналітичну підтримку маркетингової діяльності. Такі рішення дозволяють не лише здійснювати публікацію контенту, але й аналізувати ефективність окремих кампаній, визначати найкращий час для розміщення матеріалів та оцінювати поведінку аудиторії. Використання аналітичних інструментів дозволяє приймати більш обґрунтовані управлінські рішення та підвищувати результативність SMM-діяльності [8].

Проте наявність великої кількості функцій не завжди є перевагою. У багатьох випадках користувачеві необхідний лише базовий набір можливостей, пов’язаний із плануванням контенту та отриманням нагадувань. Надмірна складність інтерфейсу може негативно впливати на швидкість роботи та вимагати додаткових ресурсів для освоєння системи [5].

Загалом більшість сучасних інструментів автоматизації SMM-процесів мають спільні функціональні можливості, зокрема планування та відкладену публікацію контенту, управління декількома соціальними акаунтами, аналітику ефективності публікацій, взаємодію з аудиторією та підтримку командної роботи [6].

Проте, незважаючи на широкий функціонал, існуючі рішення мають ряд суттєвих недоліків. По-перше, більшість із них є платними або мають обмежену безкоштовну версію. Це ускладнює їх використання для малих бізнесів, фрилансерів або студентських проєктів [4]. По-друге, багато сервісів мають складний інтерфейс, що потребує часу на освоєння та навчання.

Крім того, універсальні платформи часто містять надлишковий функціонал, який не використовується у повсякденній роботі. Це ускладнює взаємодію з системою та знижує її зручність. Наприклад, для користувача, якому необхідно лише планувати публікації та отримувати нагадування, використання складних багатофункціональних систем є недоцільним [5].

Ще однією проблемою є залежність від сторонніх сервісів. Використання готових платформ передбачає передачу даних третім сторонам, що може створювати ризики з точки зору конфіденційності та безпеки [11]. Крім того, такі сервіси можуть змінювати умови використання або обмежувати функціональність.

Не менш важливим фактором є питання адаптації програмних рішень до індивідуальних потреб користувачів. Більшість популярних платформ створюються як універсальні продукти, орієнтовані на широкий спектр завдань. У результаті користувачі змушені працювати з функціоналом, який може не відповідати специфіці їхньої діяльності. Це знижує рівень гнучкості та ускладнює впровадження таких систем у невеликих проєктах [11].

Розвиток месенджерів та бот-платформ створює альтернативний підхід до автоматизації SMM-процесів. На відміну від традиційних веб-сервісів, Telegram-боти можуть забезпечувати доступ до основних функцій через звичний для користувача інтерфейс месенджера. Це суттєво спрощує процес взаємодії та зменшує вимоги до технічної підготовки користувача [12].

Крім того, Telegram-боти мають високу гнучкість налаштування та можуть бути адаптовані під конкретні потреби організації або окремого спеціаліста. Завдяки відкритому програмному інтерфейсу Telegram Bot API розробники отримують можливість створювати рішення, які реалізують лише необхідний функціонал без перевантаження системи зайвими можливостями [12].

Важливо також зазначити, що більшість існуючих рішень орієнтовані на веб-інтерфейс або мобільні додатки, тоді як інтеграція з месенджерами реалізована обмежено. У той же час месенджери, зокрема Telegram, активно

використовуються як основний засіб комунікації, що відкриває нові можливості для автоматизації [12].

У цьому контексті перспективним напрямом є використання Telegram-ботів як інструменту автоматизації SMM-процесів. Telegram-боти дозволяють реалізувати базові функції планування та управління контентом у простому та доступному форматі. Вони не потребують встановлення додаткового програмного забезпечення, забезпечують швидку взаємодію з користувачем та можуть бути адаптовані під конкретні задачі.

Таким чином, проведений аналіз показав, що існуючі рішення автоматизації SMM-процесів мають широкий функціонал, але не завжди відповідають потребам користувачів з точки зору доступності, простоти використання та гнучкості. Це створює передумови для розробки альтернативних інструментів, зокрема Telegram-ботів, які дозволяють ефективно автоматизувати основні процеси планування та публікації контенту.

1.4 Постановка задачі автоматизації планування та публікації контенту

У сучасних умовах розвитку цифрового маркетингу процеси планування та публікації контенту в соціальних мережах є важливою складовою діяльності підприємств, організацій та окремих фахівців у сфері SMM. Постійне зростання обсягів контенту, необхідність дотримання графіків публікацій та контролю ефективності комунікації з аудиторією обумовлюють потребу в автоматизації відповідних процесів. Існуючі програмні рішення часто характеризуються складністю використання, надлишковим функціоналом або обмеженнями безкоштовних версій, що знижує їх доступність для широкого кола користувачів [2, 4].

Тому ставиться завдання створити Telegram-бот для автоматизації процесів планування та публікації контенту, який забезпечуватиме зручне

керування контент-планом, організацію нагадувань про майбутні публікації та підтримку основних SMM-процесів.

Об’єктом роботи є процес планування та публікації контенту в соціальних мережах.

Метою роботи є розроблення Telegram-бота для автоматизації планування та публікації контенту, що забезпечує створення та керування контент-планом, збереження інформації про заплановані публікації та організацію системи нагадувань користувачу.

Для досягнення мети необхідно вирішити такі завдання:

– проаналізувати особливості SMM як предметної області та процеси планування контенту в соціальних мережах;

– дослідити існуючі програмні засоби автоматизації SMM-процесів та визначити їх переваги й недоліки;

– сформулювати функціональні та технічні вимоги до програмного продукту;

– розробити алгоритми формування контент-плану, організації нагадувань та взаємодії користувача з Telegram-ботом;

– спроєктувати архітектуру програмного продукту та структуру збереження даних;

– реалізувати Telegram-бот засобами мови програмування Go з використанням Telegram Bot API;

– виконати тестування розробленої системи та оцінити коректність реалізації основних функцій.

1.5 Вимоги до програмного продукту

Розробка програмного продукту для автоматизації процесів планування та публікації контенту в SMM передбачає визначення чітких вимог, які забезпечують його ефективність, зручність використання та відповідність потребам користувачів. Вимоги формуються на основі аналізу предметної

області, існуючих рішень та поставленої задачі, що дозволяє створити функціонально завершений і практично орієнтований програмний продукт [13].

Усі вимоги до системи доцільно поділити на функціональні та нефункціональні. Функціональні вимоги визначають основні можливості програмного продукту та описують дії, які система повинна виконувати. У межах даної роботи Telegram-бот повинен забезпечувати введення інформації про заплановані публікації. Користувач має мати можливість додавати нові записи, що містять текст публікації, дату та час її розміщення, а також додаткові примітки.

Важливою функціональною вимогою є можливість формування та ведення контент-плану. Система повинна забезпечувати збереження інформації про заплановані публікації та надавати користувачу можливість переглядати їх у зручному форматі. Крім того, необхідно передбачити можливість редагування та видалення записів, що забезпечує гнучкість у керуванні контентом. Однією з ключових функцій програмного продукту є організація нагадувань. Telegram-бот повинен автоматично надсилати повідомлення користувачу перед запланованим часом публікації. Це дозволяє уникнути пропусків та забезпечити регулярність розміщення контенту. Система також повинна забезпечувати ефективну взаємодію з користувачем. Це передбачає підтримку команд, обробку запитів та надання відповідей у текстовому форматі. Взаємодія може здійснюватися через текстові повідомлення або за допомогою інтерактивних кнопок, що підвищує зручність використання. Такі можливості відповідають загальній логіці роботи Telegram Bot API, що надає інструменти для обробки повідомлень, команд і взаємодії з користувачем через інтерфейс месенджера [14]. Ще однією важливою функціональною вимогою є забезпечення збереження даних. Усі введені користувачем записи повинні зберігатися у системі та бути доступними для подальшого використання. Це може бути реалізовано за допомогою бази даних або інших засобів зберігання інформації. Для

невеликих програмних систем доцільним є використання легких вбудованих баз даних, зокрема SQLite, яка дозволяє зберігати структуровані дані без необхідності налаштування окремого серверного середовища [15].

Нефункціональні вимоги визначають якісні характеристики системи та умови її експлуатації. Однією з основних нефункціональних вимог є зручність використання. Інтерфейс Telegram-бота повинен бути простим, інтуїтивно зрозумілим та не вимагати спеціальної підготовки від користувача.

Важливою характеристикою є надійність системи. Програмний продукт повинен стабільно працювати, коректно обробляти запити користувачів та не допускати втрати даних. Надійність є критичною для забезпечення безперервності роботи та довіри користувачів.

До нефункціональних вимог належить також продуктивність. Telegram-бот повинен швидко реагувати на запити користувача та забезпечувати своєчасне виконання функцій, зокрема надсилання нагадувань. Затримки у роботі можуть негативно вплинути на ефективність системи.

Ще одним важливим аспектом є масштабованість. Програмний продукт повинен бути спроєктований таким чином, щоб у майбутньому можна було розширювати його функціональні можливості без суттєвих змін у структурі системи. Це дозволяє адаптувати продукт до нових вимог і потреб користувачів.

Необхідно також враховувати вимоги до безпеки. Хоча система не обробляє критично важливі дані, необхідно забезпечити базовий рівень захисту інформації користувача та обмежити доступ до неї сторонніх осіб.

Окремо слід зазначити вимогу сумісності. Програмний продукт повинен коректно працювати на різних пристроях та операційних системах, оскільки доступ до Telegram здійснюється через різні платформи. Це забезпечує універсальність і доступність системи.

Важливою перевагою розроблюваного програмного продукту є відсутність необхідності встановлення додаткового програмного

забезпечення. Telegram-бот функціонує у межах месенджера, що значно спрощує його використання та робить доступним для широкого кола користувачів.

Таким чином, сформульовані вимоги визначають основні характеристики, функціональні можливості та принципи роботи Telegram-бота для автоматизації процесів планування та публікації контенту. Вони охоплюють як функціональні аспекти системи, пов’язані зі створенням, редагуванням, збереженням і переглядом контент-плану, так і нефункціональні вимоги щодо зручності використання, надійності, продуктивності та простоти взаємодії з користувачем.

Визначення вимог на початковому етапі проєктування дозволяє сформувати цілісне уявлення про майбутній програмний продукт, встановити межі його функціональності та забезпечити узгодженість між поставленими цілями і засобами їх досягнення. Крім того, чітко сформульовані вимоги виступають основою для розроблення алгоритмів роботи системи, проєктування її архітектури, структури даних та інтерфейсу взаємодії з користувачем.

Важливим результатом проведеного аналізу є визначення ключових функцій, необхідних для ефективної підтримки SMM-процесів, зокрема організації контент-плану, автоматичного нагадування про майбутні публікації, збереження інформації про контент та швидкого доступу до неї через інтерфейс Telegram. Реалізація зазначених вимог дозволить створити програмний продукт, орієнтований на потреби сучасних користувачів та придатний для практичного використання у сфері цифрового маркетингу.

Отже, сформований перелік вимог створює основу для подальшого проєктування та реалізації програмного продукту, забезпечує його відповідність поставленій задачі та сприяє створенню ефективного інструменту автоматизації процесів планування і публікації контенту в соціальних мережах.

2 АЛГОРИТМИ ТА МЕТОДИ АВТОМАТИЗАЦІЇ SMM-ПРОЦЕСІВ

2.1 Загальна логіка роботи Telegram-бота

У сучасних умовах автоматизації інформаційних процесів розробка програмних засобів передбачає чітке визначення логіки їх функціонування. Для реалізації задачі автоматизації планування та публікації контенту було обрано підхід, що базується на використанні Telegram-бота як інтерактивної системи взаємодії з користувачем [12].

Telegram-бот у межах даної роботи виступає як програмний агент, який приймає запити користувача, обробляє їх відповідно до заданих алгоритмів та повертає результат у вигляді повідомлень. Загальна логіка роботи системи базується на подієво-орієнтованому підході, де основною одиницею взаємодії є повідомлення користувача.

Подієво-орієнтований підхід є одним із найбільш поширених способів побудови сучасних інформаційних систем, що працюють у режимі реального часу. Його сутність полягає в тому, що виконання певних операцій ініціюється не за заздалегідь визначеним сценарієм, а в результаті виникнення конкретної події. У випадку Telegram-бота такими подіями виступають повідомлення користувача, натискання інтерактивних кнопок, запити до бази даних або настання визначеного часу для надсилання нагадування. Використання такого підходу дозволяє забезпечити гнучкість роботи системи та швидке реагування на дії користувача [17].

У загальному вигляді процес функціонування Telegram-бота можна представити як послідовність взаємопов’язаних етапів. Першим етапом є отримання вхідного запиту від користувача. Запит може бути представлений у вигляді текстової команди або взаємодії з інтерфейсними елементами (кнопками). Після отримання запиту система здійснює його аналіз, визначаючи тип команди та відповідний сценарій обробки.

Другим етапом є обробка запиту. На цьому етапі відбувається визначення необхідних дій, які повинні бути виконані системою. Наприклад, якщо користувач надсилає команду для створення нового запису, система переходить до сценарію додавання інформації до контент-плану. Якщо запит пов’язаний із переглядом даних, здійснюється отримання інформації зі сховища та її підготовка до відображення.

Третім етапом є взаємодія з модулем збереження даних. Усі дані, пов’язані з контент-планом, зберігаються у структурованому вигляді. Це дозволяє забезпечити їх подальше використання, редагування та аналіз. Модуль збереження даних виконує функції запису, читання, оновлення та видалення інформації.

Четвертим етапом є формування відповіді користувачу. Після виконання необхідних операцій система генерує відповідне повідомлення, яке містить результат виконання запиту. Це може бути підтвердження додавання запису, відображення списку публікацій або нагадування про заплановану подію.

Окремим важливим елементом логіки роботи Telegram-бота є система нагадувань. Вона функціонує незалежно від безпосередніх запитів користувача та базується на механізмах планування подій. Система відстежує час запланованих публікацій та надсилає повідомлення користувачу за визначений проміжок часу до їх настання.

Функціонування системи нагадувань передбачає постійний контроль стану запланованих подій та своєчасне виконання відповідних дій. Для кожного запису контент-плану визначається окремий часовий параметр, який використовується для формування повідомлення користувачу. Такий підхід дозволяє забезпечити персоналізований контроль за кожною публікацією та виключити ризик пропуску важливих інформаційних повідомлень.

Особливістю даного механізму є його автономний характер роботи. Після створення запису користувачеві не потрібно додатково контролювати процес формування нагадувань. Усі необхідні перевірки виконуються

системою автоматично, що суттєво зменшує навантаження на користувача та підвищує загальну ефективність роботи із контент-планом. Аналогічний підхід використовується в багатьох сучасних системах автоматизації управління інформаційними процесами, де значна частина операцій виконується у фоновому режимі [11].

Додатковою перевагою такого рішення є можливість подальшого розширення функціоналу. У майбутньому система нагадувань може бути доповнена механізмами повторних сповіщень, пріоритезації окремих записів або автоматичного інформування про зміни у графіку публікацій. Це дозволить підвищити рівень автоматизації та зробити взаємодію користувача із системою ще більш зручною [17].

Важливо зазначити, що життєвий цикл кожного повідомлення в системі складається з декількох послідовних етапів. Спочатку повідомлення надходить через Telegram Bot API до програмного модуля бота. Далі здійснюється його попередня обробка та перевірка. Після цього визначається сценарій виконання, формується відповідний запит до бази даних або інших компонентів системи та генерується відповідь користувачу. Така організація дозволяє розділити процес обробки повідомлень на окремі логічні етапи та забезпечити стабільність функціонування програмного продукту [3].

З точки зору архітектури, Telegram-бот можна розглядати як систему, що складається з декількох логічних компонентів. До основних компонентів належать: інтерфейс взаємодії з користувачем, модуль обробки запитів, модуль управління даними та модуль планування подій. Взаємодія між цими компонентами забезпечує цілісність і ефективність роботи системи.

Інтерфейс взаємодії реалізується через можливості платформи Telegram та забезпечує прийом і відправлення повідомлень. Модуль обробки запитів відповідає за аналіз вхідних даних та вибір відповідного алгоритму обробки. Модуль управління даними забезпечує роботу з інформацією про контент-план, а модуль планування подій відповідає за організацію нагадувань.

Важливою перевагою використання Telegram як платформи для реалізації інтерфейсу є підтримка інтерактивних елементів керування. Зокрема, користувач може взаємодіяти із системою не лише за допомогою текстових команд, але й через кнопки меню, списки вибору та інші елементи інтерфейсу. Це дозволяє спростити роботу з ботом і зробити її більш зрозумілою навіть для користувачів, які не мають спеціальної технічної підготовки [12].

Використання інтерактивних елементів також сприяє зменшенню кількості помилок під час введення команд. Замість ручного введення тексту користувач може обрати необхідну дію зі списку доступних операцій, що підвищує швидкість взаємодії та покращує загальний користувацький досвід.

Кожен із зазначених модулів виконує власні функції та взаємодіє з іншими компонентами системи через визначені програмні інтерфейси. Такий підхід відповідає принципам модульного проєктування програмного забезпечення, згідно з якими окремі частини системи можуть розроблятися та тестуватися незалежно одна від одної [11]. Це підвищує надійність програмного продукту та спрощує його подальшу модернізацію.

Особливістю розроблюваного Telegram-бота є його орієнтація на простоту та зручність використання. Логіка роботи системи побудована таким чином, щоб користувач міг швидко виконувати необхідні дії без необхідності вивчення складних команд або інструкцій. Це досягається за рахунок використання зрозумілих команд та структурованої взаємодії.

Перевагою використання Telegram-бота для автоматизації SMM-процесів є також його кросплатформеність. Користувач може працювати із системою через смартфон, планшет або персональний комп’ютер без встановлення додаткових програмних засобів. Усі функції доступні безпосередньо в середовищі месенджера, що спрощує доступ до системи та знижує вимоги до технічної підготовки користувача [12].

Загальна логіка роботи Telegram-бота може бути представлена у вигляді алгоритму, який включає такі основні кроки: отримання запиту, його

аналіз, визначення сценарію обробки, виконання необхідних операцій, формування відповіді та, у разі необхідності, планування подій для подальшого виконання. Для реалізації подібної логіки доцільно використовувати Python, оскільки ця мова має розвинені засоби для створення прикладних програм, роботи з даними та асинхронної обробки подій [17].

Таким чином, запропонована логіка роботи Telegram-бота забезпечує ефективну автоматизацію процесів планування та публікації контенту, дозволяє оптимізувати взаємодію з користувачем та створює основу для подальшої реалізації функціональних можливостей системи.

2.2 Алгоритм формування контент-плану

Формування контент-плану є одним із ключових етапів автоматизації процесів SMM, оскільки саме він забезпечує системність та впорядкованість публікацій. У межах розроблюваного Telegram-бота алгоритм формування контент-плану реалізується як послідовність дій, спрямованих на введення, обробку та збереження інформації про майбутні публікації [4].

Контент-план у даній системі представляється як структурована множина записів, кожен з яких містить інформацію про окрему публікацію. До складу такого запису входять текст повідомлення, дата та час публікації, а також додаткові параметри, що можуть бути використані для класифікації або уточнення змісту.

У практиці SMM контент-план виконує роль інструменту координації та контролю маркетингової активності. Його використання дозволяє забезпечити регулярність публікацій, дотримання тематичного балансу та своєчасне розміщення інформаційних матеріалів. Відсутність контент-плану часто призводить до хаотичного розміщення контенту, пропуску важливих подій та зниження ефективності взаємодії з аудиторією [2]. Саме тому

автоматизація процесу його формування є одним із ключових завдань розроблюваного програмного продукту.

Алгоритм формування контент-плану починається з ініціації процесу користувачем. Користувач надсилає відповідну команду Telegram-боту, яка сигналізує про необхідність створення нового запису. Після отримання команди система переходить у режим збору даних, у якому поетапно запитує необхідну інформацію. Такий принцип взаємодії відповідає можливостям Telegram Bot API, який дозволяє обробляти повідомлення користувачів, команди та сценарії діалогу [12].

На першому етапі здійснюється введення тексту публікації. Користувач надсилає текстове повідомлення, яке зберігається у тимчасовій змінній. На цьому етапі система може виконувати базову перевірку коректності введених даних, наприклад перевіряти, чи не є текст порожнім.

Під час введення тексту система може виконувати додатковий аналіз повідомлення. Зокрема, може перевірятися наявність мінімального обсягу тексту, відсутність службових символів або інших елементів, які можуть негативно впливати на подальше використання запису. Також на цьому етапі можуть здійснюватися операції попередньої обробки тексту, пов’язані з його форматуванням та підготовкою до збереження [20].

Для підвищення зручності користувача система повинна підтримувати можливість введення текстів різного обсягу. Це дозволяє використовувати Telegram-бот як для коротких інформаційних повідомлень, так і для більш розгорнутих публікацій. Такий підхід забезпечує універсальність використання програмного продукту та розширює сферу його практичного застосування.

Другим етапом є введення дати публікації. Користувач задає дату у визначеному форматі, після чого система перевіряє її коректність. Зокрема, перевіряється, чи відповідає дата встановленому формату та чи не є вона меншою за поточну дату. У разі некоректного введення система запитує повторне введення даних.

Третім етапом є введення часу публікації. Аналогічно до попереднього етапу, здійснюється перевірка правильності формату часу та його логічної узгодженості з датою публікації.

Важливою складовою алгоритму є процедура валідації введених даних. Валідація дозволяє виявити помилки ще до моменту збереження запису в системі. До основних перевірок належать контроль формату дати та часу, перевірка наявності обов’язкових полів, а також перевірка того, що момент публікації знаходиться у майбутньому. Використання механізмів валідації підвищує надійність системи та запобігає виникненню помилок під час роботи з контент-планом [20].

Після введення всіх необхідних параметрів система формує структурований запис, який містить усю інформацію про заплановану публікацію. На цьому етапі може бути реалізована додаткова обробка даних, наприклад перетворення дати та часу у єдиний часовий формат для подальшого використання.

Структура запису контент-плану може бути представлена у вигляді таблиці (табл. 2.1).

Таблиця 2.1 - Структура запису контент-плану

Поле	**Призначення**
ID запису	Унікальний ідентифікатор публікації
Текст публікації	Основний зміст повідомлення
Дата публікації	Запланована дата розміщення
Час публікації	Запланований час розміщення
Час нагадування	Момент надсилання сповіщення
Статус	Активна, виконана або видалена публікація

Наведена структура запису є достатньою для реалізації основних функцій системи та забезпечує можливість подальшого розширення функціоналу. За необхідності до структури можуть бути додані додаткові

атрибути, наприклад категорія публікації, тип контенту, пріоритет повідомлення або перелік пов’язаних файлів. Використання структурованого підходу до організації даних дозволяє адаптувати систему до нових вимог без суттєвої зміни її архітектури [11].

Важливим аспектом є забезпечення цілісності даних під час роботи із контент-планом. Кожен запис повинен мати унікальний ідентифікатор, що дозволяє однозначно визначати його серед інших записів. Це спрощує виконання операцій редагування, пошуку та видалення інформації, а також підвищує надійність функціонування системи в цілому [15].

Після формування структури запису система виконує його підготовку до збереження. На цьому етапі можуть здійснюватися додаткові операції, пов’язані з форматуванням тексту, перевіркою довжини повідомлення або обчисленням параметрів нагадування. Такий підхід дозволяє мінімізувати кількість операцій під час подальшого використання запису.

Наступним кроком є збереження сформованого запису у сховищі даних. Для цього використовується відповідний механізм збереження, який забезпечує доступ до інформації у подальшому. Збереження даних дозволяє реалізувати функції перегляду, редагування та видалення записів. Для таких задач може використовуватися SQLite як легка вбудована база даних, що не потребує окремого серверного налаштування [15].

Після успішного збереження запису система формує повідомлення-підтвердження, яке надсилається користувачу. Це повідомлення містить інформацію про те, що запис було додано до контент-плану.

Окремим етапом алгоритму є інтеграція з модулем нагадувань. На основі введеної дати та часу система визначає момент, коли необхідно надіслати нагадування користувачу. Для цього розраховується час сповіщення, який може бути заданий як певний інтервал до моменту публікації.

Процес інтеграції з модулем нагадувань передбачає передачу до нього всієї необхідної інформації про майбутню публікацію. До таких даних

належать дата та час розміщення контенту, текст повідомлення, а також параметри формування сповіщення. Після отримання інформації модуль нагадувань виконує розрахунок часу надсилання повідомлення та забезпечує його подальший контроль [17].

Слід зазначити, що використання автоматизованого механізму нагадувань суттєво знижує ризик пропуску запланованих публікацій. У традиційних умовах користувач змушений самостійно контролювати графік розміщення контенту, що може призводити до помилок або порушення регулярності публікацій. Автоматизація цього процесу дозволяє підвищити дисципліну ведення соціальних мереж та забезпечити стабільність інформаційної активності [2].

Алгоритм формування контент-плану також передбачає можливість обробки помилок. У разі некоректного введення даних система повинна інформувати користувача про помилку та пропонувати повторне введення. Це забезпечує надійність роботи та підвищує зручність використання.

Крім базового сценарію створення запису, система повинна підтримувати можливість подальшого редагування інформації. Це дозволяє користувачу змінювати дату, час або текст публікації відповідно до поточних потреб. У результаті контент-план залишається актуальним навіть за умов зміни маркетингової стратегії або графіка роботи.

У загальному вигляді алгоритм формування контент-плану можна представити у вигляді такої послідовності дій:

- отримання команди на створення запису;
- введення тексту публікації;
- введення дати публікації;
- введення часу публікації;
- перевірка коректності введених даних;
- формування структурованого запису;
- збереження запису у системі;
- формування підтвердження для користувача;

– передача даних до модуля нагадувань.

Застосування такого алгоритму дозволяє забезпечити впорядкованість процесу створення контент-плану, мінімізувати ймовірність помилок та підвищити ефективність роботи користувача. Крім того, чітка структура алгоритму створює основу для подальшого розширення функціональності системи, наприклад додавання категорій контенту або автоматичної генерації рекомендацій.

Таким чином, алгоритм формування контент-плану є ключовим компонентом системи автоматизації SMM-процесів та забезпечує реалізацію основної функціональності розроблюваного Telegram-бота.

2.3 Алгоритм організації нагадувань та керування публікаціями

Однією з ключових функціональних можливостей розроблюваного Telegram-бота є організація системи нагадувань та керування запланованими публікаціями. Саме ця функція забезпечує своєчасність виконання дій користувача та дозволяє уникнути пропуску важливих публікацій. У зв’язку з цим алгоритм організації нагадувань є важливим елементом загальної логіки функціонування системи.

Основною ідеєю даного алгоритму є автоматичне відстеження часу запланованих подій та надсилання користувачу відповідних повідомлень за певний проміжок часу до моменту публікації. Для реалізації цієї функції використовується підхід, що базується на плануванні подій та періодичній перевірці їх стану.

У практичній діяльності SMM-спеціалістів своєчасність публікацій має суттєве значення, оскільки навіть незначне відхилення від запланованого графіка може вплинути на охоплення аудиторії та ефективність комунікації. Саме тому система нагадувань є важливим інструментом підтримки контент-плану та контролю виконання запланованих дій [4].

Після створення запису у контент-плані, який містить дату та час публікації, система визначає момент, коли необхідно надіслати нагадування. Як правило, нагадування формується заздалегідь, наприклад за одну годину або інший заданий інтервал до запланованого часу. Цей інтервал може бути зафіксованим або визначатися користувачем.

На початковому етапі алгоритму здійснюється розрахунок часу нагадування. Для цього від часу публікації віднімається встановлений інтервал, у результаті чого отримується точка часу, коли має бути відправлене повідомлення. Отримане значення зберігається разом із записом у контент-плані або передається до модуля планування подій.

Наступним етапом є організація механізму відстеження часу. Для цього система використовує циклічну перевірку або планувальник задач, який регулярно аналізує список запланованих подій. Під час кожної перевірки система порівнює поточний час із часом нагадування для кожного запису. Реалізація таких операцій може здійснюватися засобами Python, який має вбудовані інструменти для роботи з датою, часом та асинхронними процесами [17].

Для забезпечення стабільної роботи системи доцільно використовувати спеціальні механізми планування задач. Такі механізми дозволяють автоматично запускати процедури перевірки через певні проміжки часу без втручання користувача. У результаті Telegram-бот здатний працювати автономно та виконувати функції нагадувань навіть за відсутності активної взаємодії з користувачем.

Якщо поточний час відповідає або перевищує запланований час нагадування, система формує повідомлення та надсилає його користувачу. Повідомлення містить інформацію про майбутню публікацію, зокрема її текст, дату та час. Це дозволяє користувачу своєчасно підготуватися до розміщення контенту.

Структура повідомлення може включати такі елементи (табл. 2.2):

Таблиця 2.2 - Інформація, що міститься у повідомленні-нагадуванні

Елемент повідомлення	**Призначення**
Назва нагадування	Ідентифікація події
Текст публікації	Відображення змісту контенту
Дата публікації	Інформування про день розміщення
Час публікації	Інформування про момент розміщення
Додаткова примітка	За потреби уточнення інформації

Наявність структурованого повідомлення дозволяє користувачу швидко отримати всю необхідну інформацію про заплановану подію без необхідності додаткового звернення до контент-плану. Завдяки цьому підвищується оперативність роботи та скорочується час на пошук необхідних відомостей. Особливо важливим це є у випадках, коли користувач одночасно працює з великою кількістю публікацій або веде декілька інформаційних ресурсів.

Крім інформування про майбутню публікацію, повідомлення-нагадування може виконувати функцію контролю виконання завдань. Отримавши нагадування, користувач має можливість перевірити актуальність підготовленого контенту, внести необхідні зміни або скоригувати час розміщення. Таким чином система нагадувань виступає не лише інструментом інформування, а й засобом підтримки процесу управління контентом [9].

Використання автоматичних сповіщень особливо актуальне для малого бізнесу, контент-менеджерів та SMM-фахівців, які самостійно ведуть декілька сторінок у соціальних мережах. У таких умовах контроль великої кількості публікацій вручну є складним завданням, тоді як автоматизоване нагадування дозволяє уникнути пропуску важливих інформаційних повідомлень та маркетингових активностей.

Після надсилання нагадування система може виконувати додаткові дії, наприклад змінювати статус запису або фіксувати факт відправлення

повідомлення. Це дозволяє уникнути повторного надсилання одного й того самого нагадування.

Крім функції нагадувань, алгоритм передбачає керування публікаціями. Це включає можливість перегляду списку запланованих записів, їх редагування та видалення. Для цього користувач надсилає відповідні команди, які обробляються системою за аналогією з іншими запитами.

У процесі керування публікаціями важливим є забезпечення актуальності даних. У разі зміни дати або часу публікації система повинна повторно розрахувати час нагадування та оновити відповідні параметри. Це гарантує коректність роботи механізму сповіщень.

Важливим аспектом роботи системи є підтримка життєвого циклу запису. Кожна публікація проходить декілька станів: створення, збереження, очікування виконання, надсилання нагадування та завершення. Такий підхід дозволяє контролювати всі етапи існування запису та забезпечує впорядкованість роботи системи.

Життєвий цикл запису забезпечує впорядкованість функціонування всієї системи. На етапі створення відбувається введення та перевірка інформації, після чого запис переходить у стан очікування. У цьому стані система виконує моніторинг часу та контролює наближення моменту надсилання нагадування. Після виконання відповідної дії запис отримує статус виконаного або завершеного. Такий підхід дозволяє уникнути дублювання операцій та спрощує контроль над усіма публікаціями.

Для більш наочного представлення життєвого циклу публікації доцільно виділити основні стани запису:

- створення запису;
- перевірка введених даних;
- збереження у базі даних;
- очікування часу нагадування;
- надсилання повідомлення;
- завершення або архівування запису.

Послідовна зміна зазначених станів дозволяє реалізувати логічно завершений процес керування контентом та забезпечує стабільність роботи програмного продукту [11]. Крім того, використання статусів спрощує подальше масштабування системи, оскільки у майбутньому до алгоритму можуть бути додані нові етапи обробки або додаткові типи подій.

Алгоритм також передбачає обробку ситуацій, коли час публікації вже минув. У такому випадку система може або видаляти відповідний запис, або змінювати його статус, наприклад позначати як виконаний. Це дозволяє підтримувати актуальність контент-плану та уникати накопичення застарілих даних.

Окрему увагу слід приділити обробці нестандартних ситуацій. До таких ситуацій належать помилки під час збереження даних, відсутність з’єднання з мережею або некоректне введення параметрів користувачем. У подібних випадках система повинна інформувати користувача про проблему та пропонувати можливі варіанти її вирішення. Наявність механізмів обробки помилок підвищує надійність програмного продукту та покращує користувацький досвід.

З точки зору реалізації, алгоритм організації нагадувань може бути представлений як послідовність таких дій:

отримання даних про заплановану публікацію;

- розрахунок часу нагадування;
- збереження параметрів події;
- періодична перевірка поточного часу;
- порівняння поточного часу з часом нагадування;
- формування та надсилання повідомлення;
- оновлення статусу події.

Важливою особливістю алгоритму є його автономність. Після створення запису система самостійно контролює виконання нагадувань без необхідності постійної участі користувача. Це значно підвищує ефективність роботи та зменшує навантаження.

Крім того, використання автоматизованого механізму нагадувань дозволяє зменшити ризик людського фактору. Користувачу не потрібно самостійно контролювати всі дати та час публікацій, оскільки система виконує цю функцію автоматично. Це особливо важливо при роботі з великою кількістю контенту або одночасному веденні декількох інформаційних ресурсів.

Таким чином, алгоритм організації нагадувань та керування публікаціями забезпечує реалізацію однієї з ключових функцій системи автоматизації SMM-процесів. Він дозволяє своєчасно інформувати користувача про заплановані події, підтримувати актуальність контент-плану та підвищувати ефективність управління контентом.

2.4 Алгоритми взаємодії користувача з ботом

Ефективність роботи Telegram-бота значною мірою залежить від організації взаємодії з користувачем. Саме через інтерфейс взаємодії користувач отримує доступ до всіх функціональних можливостей системи, тому алгоритми взаємодії повинні бути чітко структурованими, логічними та зручними у використанні.

У межах розроблюваного програмного продукту взаємодія користувача з ботом реалізується за допомогою текстових команд та інтерактивних елементів інтерфейсу, таких як кнопки. Основним принципом побудови взаємодії є сценарний підхід, відповідно до якого кожна дія користувача запускає певний сценарій обробки. Такий підхід відповідає можливостям Telegram Bot API, який підтримує обробку повідомлень, команд і взаємодію з елементами інтерфейсу [12].

Організація взаємодії між користувачем і програмною системою є одним із найважливіших аспектів розробки будь-якого програмного продукту. Від якості побудови діалогу залежить швидкість виконання

операцій, зручність використання системи та загальне враження користувача від роботи з програмним забезпеченням. Саме тому під час проєктування Telegram-бота особлива увага приділяється створенню зрозумілого та послідовного механізму взаємодії.

Алгоритм взаємодії починається з ініціалізації роботи бота. Користувач надсилає команду запуску, після чого система формує вітальне повідомлення та надає коротку інформацію про доступні функції. На цьому етапі користувач отримує уявлення про можливості системи та способи взаємодії з нею.

Наступним етапом є вибір користувачем необхідної дії. Це може бути створення нового запису, перегляд контент-плану, редагування або видалення існуючих записів. Для спрощення взаємодії можуть використовуватися меню або кнопки, що дозволяють уникнути необхідності запам’ятовування команд.

Використання кнопок має ряд переваг порівняно із введенням текстових команд вручну. По-перше, зменшується ймовірність помилок під час введення. По-друге, користувач швидше орієнтується у функціональних можливостях системи. По-третє, забезпечується більш інтуїтивна навігація між окремими розділами програми.

У разі вибору створення нового запису запускається сценарій формування контент-плану. Система поетапно запитує у користувача необхідні дані: текст публікації, дату та час. Кожен введений параметр обробляється окремо, після чого система переходить до наступного кроку.

Якщо користувач обирає перегляд контент-плану, система звертається до модуля збереження даних та формує список запланованих публікацій. Інформація відображається у зручному форматі, що дозволяє швидко ознайомитися з наявними записами.

У разі редагування або видалення запису алгоритм передбачає вибір конкретного елемента зі списку. Після вибору система пропонує можливі дії: змінити параметри запису або видалити його. У разі редагування

запускається сценарій повторного введення даних із подальшим оновленням інформації у системі.

Для забезпечення зручності використання доцільно передбачити реалізацію основних команд, наведених у таблиці 2.3.

Таблиця 2.3 - Основні команди Telegram-бота

Команда	**Призначення**
/start	Запуск бота
Додати запис	Створення нового запису
Переглянути записи	Відображення контент-плану
Редагувати запис	Зміна параметрів запису
Видалити запис	Видалення обраного запису
Головне меню	Повернення до початкового меню

Наведений перелік команд формує основу користувацького інтерфейсу системи та забезпечує доступ до всіх основних функцій Telegram-бота. Використання командного підходу дозволяє спростити навігацію між окремими розділами системи та забезпечити швидке виконання типових операцій. При цьому користувачу не потрібно запам’ятовувати складні послідовності дій, оскільки більшість функцій доступна через меню або інтерактивні кнопки.

Важливою перевагою такого підходу є можливість поступового освоєння функціоналу програмного продукту. Новий користувач може працювати лише з базовими командами, тоді як більш досвідчений користувач отримує доступ до всіх можливостей системи без необхідності додаткового навчання. Це позитивно впливає на зручність використання та сприяє швидкому впровадженню програмного продукту у практичну діяльність [9].

Крім того, використання стандартизованих команд дозволяє забезпечити єдність інтерфейсу на всіх етапах роботи. Незалежно від того,

яку функцію виконує користувач, логіка взаємодії залишається однаковою, що зменшує когнітивне навантаження та підвищує ефективність роботи із системою.

Важливою складовою алгоритму взаємодії є обробка помилок. Якщо користувач вводить некоректні дані або надсилає невідому команду, система повинна інформувати його про помилку та пропонувати можливі варіанти дій. Це підвищує зручність використання та запобігає виникненню непорозумінь.

Особливу увагу необхідно приділяти перевірці правильності введення дат і часу. Помилки такого типу можуть призводити до некоректної роботи системи нагадувань та втрати актуальності контент-плану. Саме тому алгоритм передбачає автоматичну перевірку введених даних та повідомлення користувача про виявлені помилки.

Алгоритм також передбачає можливість повернення до попередніх етапів. Користувач повинен мати змогу скасувати поточну дію або повернутися до головного меню. Це забезпечує гнучкість взаємодії та дозволяє уникнути помилок при введенні даних.

Ще однією важливою особливістю є підтримка діалогового режиму роботи. У такому режимі система запам’ятовує поточний стан взаємодії та формує наступні запити відповідно до попередніх відповідей користувача. Це дозволяє реалізувати складні сценарії введення даних без перевантаження інтерфейсу великою кількістю команд.

Діалоговий режим дозволяє організувати взаємодію користувача із системою у формі послідовного обміну повідомленнями. Кожен наступний крок залежить від результатів попереднього етапу, що забезпечує логічність виконання операцій та спрощує процес введення інформації. Такий підхід особливо ефективний під час створення нових записів контент-плану, коли необхідно послідовно отримати декілька взаємопов’язаних параметрів.

Використання діалогових сценаріїв також дозволяє зменшити кількість помилок користувача. Оскільки система запитує лише ту інформацію, яка

необхідна на поточному етапі, знижується ризик пропуску обов'язкових полів або введення даних у неправильному форматі. У разі виявлення помилки Telegram-бот може автоматично повернути користувача до відповідного кроку та запропонувати повторне введення інформації.

Додатковою перевагою є можливість масштабування функціональності без суттєвого ускладнення інтерфейсу. За необхідності до існуючих сценаріїв можуть бути додані нові етапи, пов'язані з категоризацією контенту, вибором типу публікації або налаштуванням параметрів нагадувань. При цьому загальна логіка взаємодії залишатиметься зрозумілою та послідовною для користувача [11].

З точки зору реалізації, взаємодія користувача з ботом базується на обробці подій. Кожне повідомлення користувача розглядається як подія, яка передається до модуля обробки запитів. Система визначає тип події та запускає відповідний сценарій. Реалізація такої логіки може здійснюватися за допомогою бібліотек Python для роботи з Telegram API, зокрема aiogram, яка підтримує асинхронну обробку повідомлень і команд користувача [16, 17].

Використання асинхронної моделі дозволяє одночасно обробляти повідомлення від користувачів та виконувати фонові процеси, наприклад перевірку нагадувань. Завдяки цьому забезпечується швидка реакція системи навіть за значної кількості одночасних запитів.

Алгоритм взаємодії можна представити у вигляді такої послідовності дій:

- отримання повідомлення від користувача;
- визначення типу команди або дії;
- вибір відповідного сценарію обробки;
- виконання необхідних операцій;
- формування відповіді;
- надсилання повідомлення користувачу.

Особливістю запропонованого підходу є орієнтація на простоту та інтуїтивність. Алгоритми взаємодії побудовані таким чином, щоб користувач

міг виконувати всі необхідні дії без додаткових інструкцій. Це досягається за рахунок логічної структури діалогу та використання зрозумілих команд.

Таким чином, алгоритми взаємодії користувача з Telegram-ботом забезпечують ефективний доступ до функціональних можливостей системи, спрощують процес роботи з контентом та підвищують зручність використання програмного продукту.

2.5 Опис обробки даних та сценаріїв використання

У процесі функціонування Telegram-бота важливу роль відіграє організація обробки даних та визначення типових сценаріїв використання системи. Саме ці аспекти забезпечують цілісність роботи програмного продукту, узгодженість його компонентів та зручність взаємодії з користувачем

У межах розроблюваного програмного продукту дані представлені у вигляді структурованих записів, що формують контент-план. Кожен запис містить основні параметри публікації, зокрема текст повідомлення, дату та час запланованого розміщення, а також службову інформацію, необхідну для організації нагадувань і керування записами.

Організація даних є одним із ключових факторів ефективності роботи системи. Від правильності структурування інформації залежить швидкість доступу до неї, можливість виконання операцій пошуку та редагування, а також стабільність роботи програмного продукту в цілому. Саме тому всі записи контент-плану повинні зберігатися у стандартизованому форматі та містити однаковий набір атрибутів.

Обробка даних у системі здійснюється на всіх етапах її функціонування – від моменту введення інформації користувачем до її збереження, оновлення та використання для формування нагадувань. На етапі введення даних система виконує перевірку їх коректності, що дозволяє

уникнути помилок у подальшій роботі. Зокрема, перевіряється формат дати та часу, наявність обов’язкових полів, а також логічна узгодженість параметрів.

Після перевірки дані перетворюються у структурований формат, який забезпечує їх ефективне збереження та обробку. Такий підхід дозволяє реалізувати основні операції з даними, включаючи додавання нових записів, їх редагування, видалення та пошук. Збереження даних може здійснюватися у вигляді таблиці або списку об’єктів, що забезпечує швидкий доступ до інформації. Для збереження структурованих записів доцільним є використання SQLite, оскільки ця база даних є легкою, вбудованою та придатною для невеликих прикладних систем [19].

Приклад структури запису контент-плану наведено у таблиці 2.4.

Таблиця 2.4 - Основні поля запису контент-плану

Поле	**Призначення**
ID запису	Унікальний ідентифікатор
Текст публікації	Основний зміст повідомлення
Дата публікації	День запланованого розміщення
Час публікації	Час розміщення
Час нагадування	Час надсилання сповіщення
Статус	Поточний стан запису

У процесі роботи системи обробка даних також включає взаємодію з модулем нагадувань. На основі збережених параметрів визначаються моменти часу, коли необхідно надсилати повідомлення користувачу. Для цього здійснюється порівняння поточного часу з часом, заданим у записах контент-плану.

Важливим аспектом є оновлення даних. У разі внесення змін до запису система повинна забезпечити актуалізацію всіх пов’язаних параметрів, зокрема часу нагадування. Це дозволяє підтримувати коректність роботи

системи та уникати помилок у виконанні запланованих дій. Окрему увагу необхідно приділяти процесу пошуку інформації. У разі збільшення кількості записів користувачу може бути складно швидко знайти потрібну публікацію. Тому система повинна забезпечувати можливість вибірки даних за певними критеріями, наприклад за датою, статусом або частиною тексту повідомлення. Життєвий цикл запису у системі можна подати як послідовність взаємопов'язаних станів. Спочатку запис створюється користувачем, після чого зберігається у базі даних. Далі він переходить у стан очікування, коли система контролює час до моменту публікації. Після надсилання нагадування запис отримує відповідний статус, а після завершення події може бути позначений як виконаний або архівований.

Окрім обробки даних, важливу роль відіграють сценарії використання системи, які описують типові дії користувача та відповідні реакції системи. Визначення таких сценаріїв дозволяє структурувати взаємодію та забезпечити логічність функціонування програмного продукту.

Одним із основних сценаріїв є сценарій створення нового запису. У цьому випадку користувач ініціює процес додавання публікації, після чого система поетапно запитує необхідні дані. Після введення всіх параметрів запис зберігається у системі та включається до контент-плану.

Іншим поширеним сценарієм є перегляд контент-плану. Користувач надсилає відповідну команду, після чого система формує список запланованих публікацій і відображає його у зручному форматі. Це дозволяє швидко отримати інформацію про майбутні публікації.

Сценарій редагування передбачає зміну параметрів існуючого запису. Користувач обирає потрібний запис і вносить зміни, після чого система оновлює дані та, за необхідності, коригує час нагадування.

Сценарій видалення дозволяє користувачу видалити непотрібний запис із контент-плану. Після підтвердження дії система видаляє відповідні дані та оновлює список публікацій. Для кращого розуміння логіки роботи системи

доцільно виділити основні сценарії використання, наведені у таблиці 2.5.

Таблиця 2.5 - Основні сценарії використання системи

Сценарій	**Дія користувача**	**Реакція системи**
Створення запису	Введення параметрів публікації	Збереження запису
Перегляд записів	Вибір відповідної команди	Відображення контент-плану
Редагування запису	Внесення змін	Оновлення даних
Видалення запису	Підтвердження видалення	Видалення запису
Отримання нагадування	Дія не потрібна	Надсилання повідомлення

Окремим сценарієм є отримання нагадування. У цьому випадку система самостійно ініціює взаємодію з користувачем, надсилаючи повідомлення про наближення часу публікації. Такий сценарій є прикладом асинхронної взаємодії, коли ініціатором виступає не користувач, а система. Асинхронна обробка подій є доцільною для Telegram-ботів, оскільки дозволяє одночасно реагувати на повідомлення користувачів і виконувати фонові задачі, зокрема перевірку нагадувань [16].

Важливою перевагою сценарного підходу є його передбачуваність. Користувач завжди отримує зрозумілу послідовність дій та може прогнозувати реакцію системи на свої команди. Це позитивно впливає на зручність використання програмного продукту та скорочує час на його освоєння. Усі сценарії використання побудовані за єдиним принципом: отримання запиту, обробка даних, виконання дії та формування відповіді. Це забезпечує узгодженість роботи системи та спрощує її реалізацію.

Таким чином, організація обробки даних та визначення сценаріїв використання є важливими складовими розробки Telegram-бота. Вони забезпечують ефективне функціонування системи, дозволяють оптимізувати взаємодію з користувачем та створюють основу для подальшого розвитку програмного продукту.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ TELEGRAM-БОТА

3.1 Обґрунтування вибору засобів та технологій програмної реалізації

У процесі розробки програмного продукту одним із ключових етапів є вибір технологій та інструментів реалізації. Від правильності цього вибору залежить ефективність роботи системи, її продуктивність, масштабованість, надійність та зручність подальшого супроводу. Саме тому під час проєктування програмного продукту особлива увага приділяється аналізу сучасних програмних засобів і технологій, які здатні забезпечити реалізацію поставлених функціональних вимог.

Для реалізації Telegram-бота було обрано мову програмування Go. Даний вибір обумовлений низкою переваг, серед яких простота синтаксису, висока швидкодія, підтримка конкурентного виконання завдань та наявність розвиненої стандартної бібліотеки. Go є сучасною компільованою мовою програмування, яка широко використовується для створення серверних застосунків, мережевих сервісів та систем автоматизації [17].

Вибір мови програмування є одним із найважливіших етапів проєктування програмного продукту. Для реалізації системи було обрано Go, оскільки ця мова поєднує високу продуктивність, простоту синтаксису та ефективні механізми роботи з мережевими застосунками. Завдяки компіляції у машинний код програми на Go характеризуються швидким виконанням та низькими витратами системних ресурсів, що є важливим для застосунків, які працюють у режимі постійної взаємодії з користувачем [17].

Однією з характерних особливостей Go є підтримка конкурентного виконання завдань за допомогою горутин. Горутини дозволяють виконувати декілька процесів одночасно без значного навантаження на систему. Це особливо важливо для Telegram-ботів, які повинні одночасно приймати

повідомлення користувачів, працювати з базою даних та виконувати фонові задачі, пов’язані з перевіркою нагадувань і формуванням повідомлень [17].

Крім того, Go має розвинену стандартну бібліотеку, яка містить засоби для роботи з мережею, файлами, JSON-даними, часовими інтервалами та HTTP-запитами. Це дозволяє реалізувати значну частину функціоналу без використання великої кількості сторонніх залежностей, що позитивно впливає на стабільність програмного продукту та спрощує його подальшу підтримку [17].

Важливою перевагою використання Go є також висока надійність програмного коду. Мова містить вбудовані механізми контролю типів даних, що дозволяє виявляти значну частину помилок ще на етапі компіляції програми. Завдяки цьому зменшується ймовірність виникнення критичних збоїв під час виконання програмного продукту. Для серверних застосунків та Telegram-ботів така особливість має важливе значення, оскільки навіть короткочасна недоступність сервісу може негативно вплинути на користувацький досвід [17].

Ще однією перевагою Go є автоматичне керування пам’яттю. Розробнику не потрібно самостійно виконувати операції виділення та звільнення пам’яті, оскільки ці функції реалізуються засобами середовища виконання. Це спрощує процес програмування та знижує ризик виникнення помилок, пов’язаних із некоректним використанням ресурсів комп’ютерної системи [17].

Окрім високої продуктивності, важливою перевагою Go є її орієнтованість на створення мережевих сервісів. Значна частина сучасних програмних продуктів працює за клієнт-серверною моделлю, тому підтримка мережевих протоколів є важливим критерієм під час вибору технології розробки. Стандартна бібліотека Go містить велику кількість засобів для роботи з HTTP-запитами, мережевими з’єднаннями та передачі даних між компонентами системи. Завдяки цьому розробник отримує можливість

реалізовувати складні мережеві взаємодії без використання великої кількості сторонніх бібліотек [17].

Важливою характеристикою Go є також кросплатформеність. Скомпільований застосунок може бути розгорнутий на різних операційних системах без суттєвих змін у програмному коді. Це забезпечує універсальність використання програмного продукту та спрощує його подальше впровадження. Для Telegram-ботів така особливість є особливо корисною, оскільки серверна частина може бути розміщена як на локальному комп’ютері розробника, так і на віддалених хмарних серверах [17].

Ще однією важливою причиною вибору Go є можливість ефективної реалізації багатопотокової обробки даних. Завдяки використанню горутин і каналів система здатна швидко реагувати на запити користувачів навіть у випадках одночасної роботи декількох користувачів. Це є важливим фактором для систем, що функціонують у режимі реального часу та повинні забезпечувати мінімальні затримки під час обробки повідомлень [17].

Для взаємодії з платформою Telegram було використано Telegram Bot API, який надає необхідний функціонал для створення та керування ботами. Використання API дозволяє реалізувати прийом і обробку повідомлень, відправлення відповідей, роботу з кнопками, меню та іншими елементами інтерфейсу користувача [12].

Telegram Bot API є офіційним інструментом розробки ботів для платформи Telegram. Його використання забезпечує сумісність програмного продукту із сервісом обміну повідомленнями та надає доступ до всіх необхідних функцій взаємодії з користувачем. API підтримує отримання повідомлень, обробку команд, роботу з файлами, фотографіями, кнопками та інтерактивними елементами інтерфейсу [12].

Для роботи з Telegram Bot API мовою Go доцільно використовувати спеціалізовані бібліотеки, які спрощують взаємодію із сервісом Telegram та дозволяють значно скоротити обсяг програмного коду. Використання

готових бібліотек забезпечує швидшу розробку системи, підвищує її надійність та спрощує реалізацію типових функцій Telegram-бота [12].

Telegram на сьогодні є одним із найпопулярніших месенджерів у світі та активно використовується як інструмент бізнес-комунікацій. Важливою перевагою платформи є підтримка ботів як окремого виду програмних агентів, які можуть автоматизувати виконання значної кількості рутинних операцій. Саме тому Telegram дедалі частіше використовується як середовище для створення сервісів автоматизації, інформаційних систем та програм підтримки користувачів [12].

Використання Telegram як платформи для реалізації програмного продукту має також економічні переваги. Користувачу не потрібно встановлювати окремий мобільний застосунок або реєструватися в новій системі. Всі функції доступні безпосередньо через вже встановлений месенджер, що спрощує впровадження та використання програмного продукту [12].

Для збереження даних було обрано використання SQLite. Дана база даних є вбудованою реляційною системою керування базами даних, яка не потребує встановлення окремого серверного програмного забезпечення та забезпечує достатній рівень продуктивності для невеликих інформаційних систем [19].

SQLite дозволяє зберігати інформацію про контент-план, параметри нагадувань та службові дані користувача у структурованому вигляді. Важливою перевагою даної системи є компактність, простота використання та мінімальні вимоги до ресурсів. Усі дані зберігаються в одному файлі бази даних, що значно спрощує резервне копіювання та перенесення системи між різними середовищами виконання [19].

Застосування SQLite також дозволяє реалізувати основні операції роботи з даними: створення записів, редагування, видалення та пошук інформації. Для невеликих програмних продуктів такий підхід є цілком достатнім і не потребує використання більш складних серверних СКБД [19].

Використання бази даних є необхідною умовою для забезпечення довготривалого збереження інформації про контент-план. Без застосування механізмів постійного зберігання даних усі створені записи втрачалися б після завершення роботи програми. Саме тому реалізація підсистеми збереження інформації є одним із ключових елементів архітектури Telegram-бота.

SQLite підтримує використання мови структурованих запитів SQL, що дозволяє ефективно виконувати пошук, сортування та фільтрацію даних. Це особливо важливо при збільшенні кількості запланованих публікацій, коли швидкість доступу до інформації починає відігравати суттєву роль. Використання реляційної моделі збереження даних забезпечує логічну організацію інформації та спрощує її подальшу обробку [19].

Крім того, SQLite характеризується високою швидкодією під час виконання типових операцій читання та запису даних. Для невеликих програмних систем продуктивності даної СКБД є більш ніж достатньо, що дозволяє уникнути використання складніших серверних рішень [19].

Також важливим аспектом є вибір середовища розробки. Для написання та тестування коду використовувалося середовище Visual Studio Code. Даний інструмент забезпечує зручний інтерфейс, підсвічування синтаксису, підтримку мови Go, інтеграцію із системами контролю версій та засоби налагодження програмного коду [25].

Visual Studio Code є одним із найбільш популярних редакторів коду серед розробників завдяки своїй універсальності та великій кількості розширень. Для роботи з мовою Go доступні спеціальні модулі, які забезпечують автоматичне форматування коду, підказки під час написання програм та інструменти аналізу помилок. Це сприяє підвищенню продуктивності розробки та покращенню якості програмного забезпечення [25].

Під час розробки програмного продукту важливу роль відіграє організація процесу тестування. Використання сучасного середовища

розробки дозволяє оперативно виявляти помилки програмного коду та виконувати їх усунення ще до етапу експлуатації системи. Завдяки інтегрованим засобам налагодження розробник може покроково аналізувати роботу програми, контролювати значення змінних та відстежувати виконання окремих функцій [25].

Додатковою перевагою Visual Studio Code є підтримка роботи з розширеннями для мови Go. Такі розширення забезпечують автоматичне форматування коду відповідно до стандартів мови, перевірку синтаксису, підказки під час написання програм та інші інструменти, що сприяють підвищенню якості програмного продукту [25].

У процесі розробки також використовуються стандартні пакети мови Go для роботи з датою та часом, зокрема пакет time, який забезпечує створення часових міток, обчислення інтервалів та реалізацію механізмів нагадувань. Використання вбудованих засобів дозволяє ефективно реалізувати планування подій та автоматичне надсилання сповіщень користувачам [17].

З точки зору архітектури програмного продукту доцільно використовувати модульний підхід, який передбачає розділення системи на окремі компоненти. Це дозволяє спростити розробку, підвищити читабельність коду та забезпечити можливість подальшого розширення функціоналу. Крім того, модульна архітектура полегшує тестування та супровід програмного продукту протягом усього життєвого циклу [15].

Аналіз обраних технологій показує, що вони повністю відповідають вимогам розроблюваної системи. Поєднання мови програмування Go, Telegram Bot API та SQLite дозволяє створити програмний продукт із високою швидкодією, зручним інтерфейсом користувача та мінімальними вимогами до апаратних ресурсів. Крім того, використання сучасних засобів розробки забезпечує можливість подальшого розвитку системи без необхідності кардинальної зміни її архітектури.

Важливо також зазначити, що всі обрані програмні засоби є вільно доступними та не потребують придбання додаткових ліцензій. Це знижує вартість впровадження програмного продукту та робить його доступним для широкого кола користувачів. Саме тому обраний технологічний стек може вважатися оптимальним для реалізації системи автоматизації процесів планування та публікації контенту в соціальних мережах [15].

Таким чином, вибір мови програмування Go, Telegram Bot API, SQLite та середовища Visual Studio Code обумовлений їх функціональністю, простотою використання, продуктивністю та відповідністю вимогам поставленої задачі. Обрані технології дозволяють реалізувати ефективний, надійний та масштабований програмний продукт для автоматизації процесів планування та публікації контенту в соціальних мережах.

3.2 Архітектура програмного продукту

Архітектура програмного продукту визначає структуру системи, взаємодію її компонентів та принципи організації обробки даних. Для розроблюваного Telegram-бота було обрано модульну архітектуру, яка забезпечує гнучкість, масштабованість та зручність подальшого розширення функціональних можливостей [15].

Модульний підхід передбачає поділ системи на окремі логічні компоненти, кожен з яких виконує визначену функцію. Така організація дозволяє спростити процес розробки, підвищити читабельність коду та забезпечити незалежність окремих частин системи (рис. 3.1).

Модульна архітектура є одним із найбільш поширених підходів до побудови сучасних програмних систем. Її використання дозволяє розподілити функціональні обов’язки між окремими компонентами та уникнути надмірної складності програмного коду. Кожен модуль реалізує власну функціональність і може змінюватися незалежно від інших компонентів. Такий підхід значно спрощує процес підтримки програмного

продукту та дозволяє швидко вносити зміни у функціонал без ризику порушення роботи всієї системи [15].

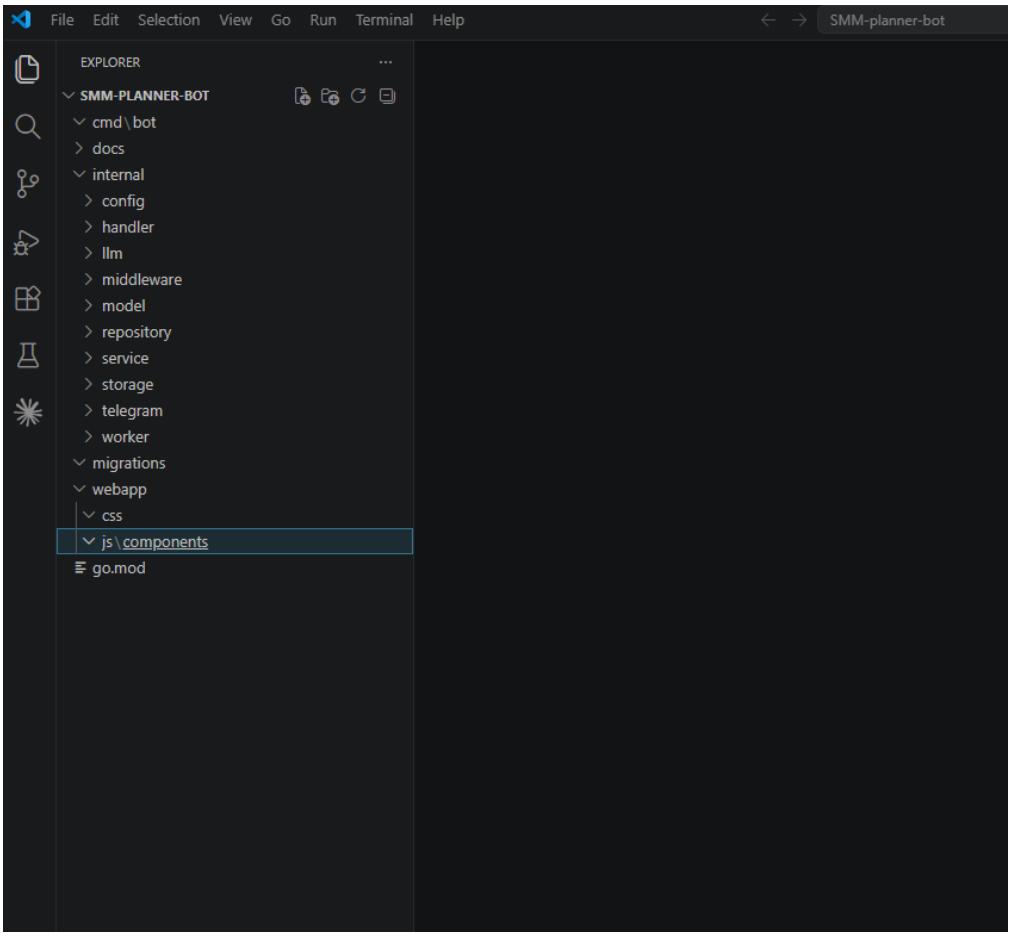

Рисунок 3.1 - Загальна структура програмного проєкту Telegram-бота

Крім того, модульний принцип побудови сприяє повторному використанню програмного коду. Окремі компоненти можуть застосовуватися в інших програмних проєктах або використовуватися під час подальшого розвитку системи. Це дозволяє скоротити витрати часу на розробку та підвищити ефективність використання програмних ресурсів.

У загальному вигляді архітектура Telegram-бота включає такі основні компоненти: модуль взаємодії з користувачем, модуль обробки запитів, модуль управління даними та модуль планування подій (рис. 3.2).

Модуль взаємодії з користувачем відповідає за прийом повідомлень від користувача та відправлення відповідей. Він реалізується за допомогою Telegram Bot API та відповідної бібліотеки, яка забезпечує обробку подій і

взаємодію з інтерфейсом месенджера [12]. Даний модуль виступає точкою входу в систему, через яку проходять усі запити користувача.

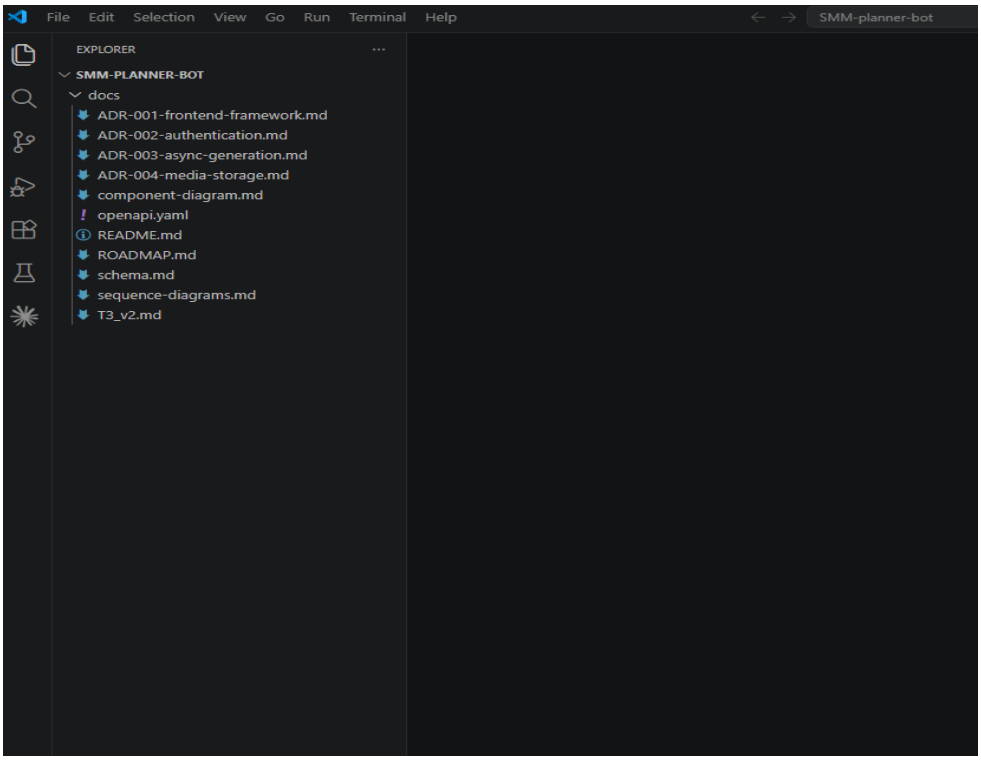

Рисунок 3.2 - Документаційні файли програмного проекту

Модуль обробки запитів є центральним елементом системи, який відповідає за аналіз вхідних даних та визначення сценарію їх обробки. На цьому рівні здійснюється розпізнавання команд користувача, вибір відповідного алгоритму та передача управління до інших компонентів системи.

Модуль управління даними відповідає за збереження, отримання, оновлення та видалення інформації про заплановані публікації. У межах даного модуля реалізується взаємодія зі сховищем даних, що може бути представлене у вигляді бази даних або файлів.

Робота з даними є критично важливою складовою функціонування Telegram-бота. Саме в цьому модулі забезпечується цілісність інформації та підтримується актуальність контент-плану. Під час виконання операцій

використовуються механізми перевірки коректності даних, що дозволяє уникнути дублювання записів та появи помилкової інформації.

Додатковою перевагою централізованого управління даними є можливість реалізації резервного копіювання та відновлення інформації. У випадку виникнення помилок або збою програмного забезпечення користувач не втрачає створений контент-план, що підвищує надійність системи загалом [19].

Представлена структура проєкту демонструє розподіл програмного коду за функціональними підсистемами. Така організація каталогів дозволяє забезпечити логічне групування файлів та спрощує навігацію під час розробки. Розділення коду на окремі пакети відповідає сучасним принципам проєктування програмного забезпечення та сприяє підвищенню його супроводжуваності.

Особливістю даного підходу є можливість незалежного розвитку окремих частин системи. Наприклад, модуль взаємодії з користувачем може бути модифікований без необхідності зміни механізмів роботи з базою даних. Аналогічно модуль генерації контенту або система авторизації можуть розширюватися без впливу на інші компоненти програмного продукту. Така організація особливо важлива для програмних систем, які передбачають подальше вдосконалення та розширення функціональних можливостей [15].

Використання структурованого підходу до збереження інформації дозволяє забезпечити ефективну роботу з контент-планом [19] (рис. 3.3).

Модуль планування подій забезпечує організацію нагадувань та контроль часу виконання запланованих дій. Він відповідає за відстеження часу публікацій, розрахунок моменту надсилання нагадувань та ініціювання відповідних подій.

Даний модуль функціонує незалежно від безпосередніх запитів користувача та реалізує асинхронну обробку подій, що є характерною для сучасних програмних систем [17].

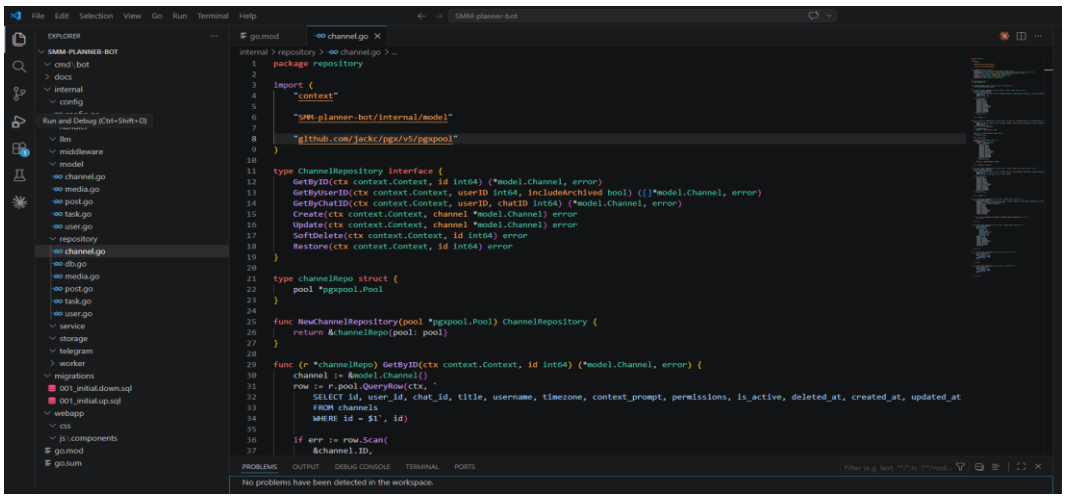

Рисунок 3.3 - Реалізація моделі даних каналу

Взаємодія між модулями здійснюється за принципом передачі даних і виклику відповідних функцій. Наприклад, після отримання запиту користувача модуль взаємодії передає його до модуля обробки запитів, який визначає необхідні дії та, у разі потреби, звертається до модуля управління даними або модуля планування подій (рис. 3.4).

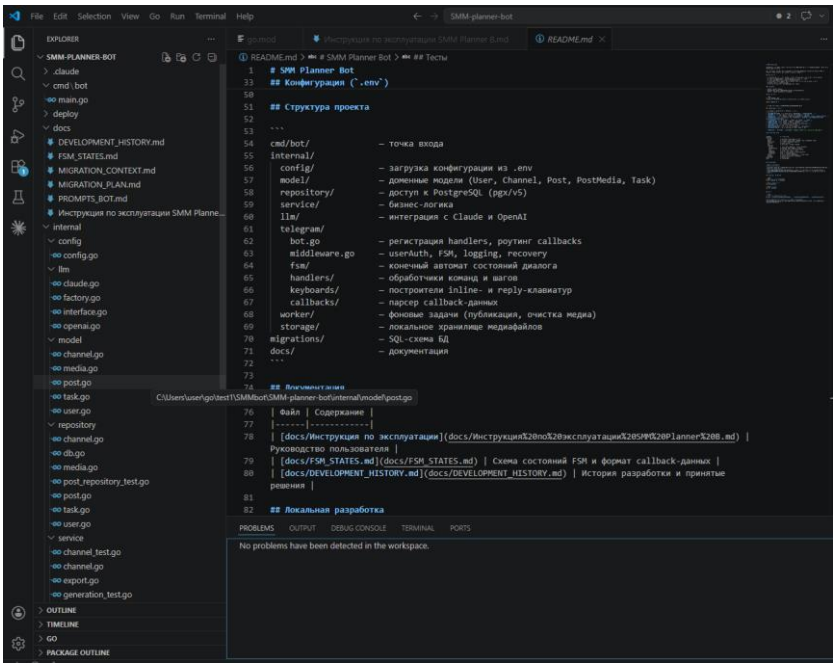

Рисунок 3.4 - Реалізація репозиторію для роботи з даними

Важливою особливістю архітектури є використання асинхронної моделі виконання. Це дозволяє системі одночасно обробляти декілька запитів

користувачів та виконувати фонові задачі, такі як надсилання нагадувань. Асинхронність забезпечує високу продуктивність та швидку реакцію системи [16] (рис. 3.5).

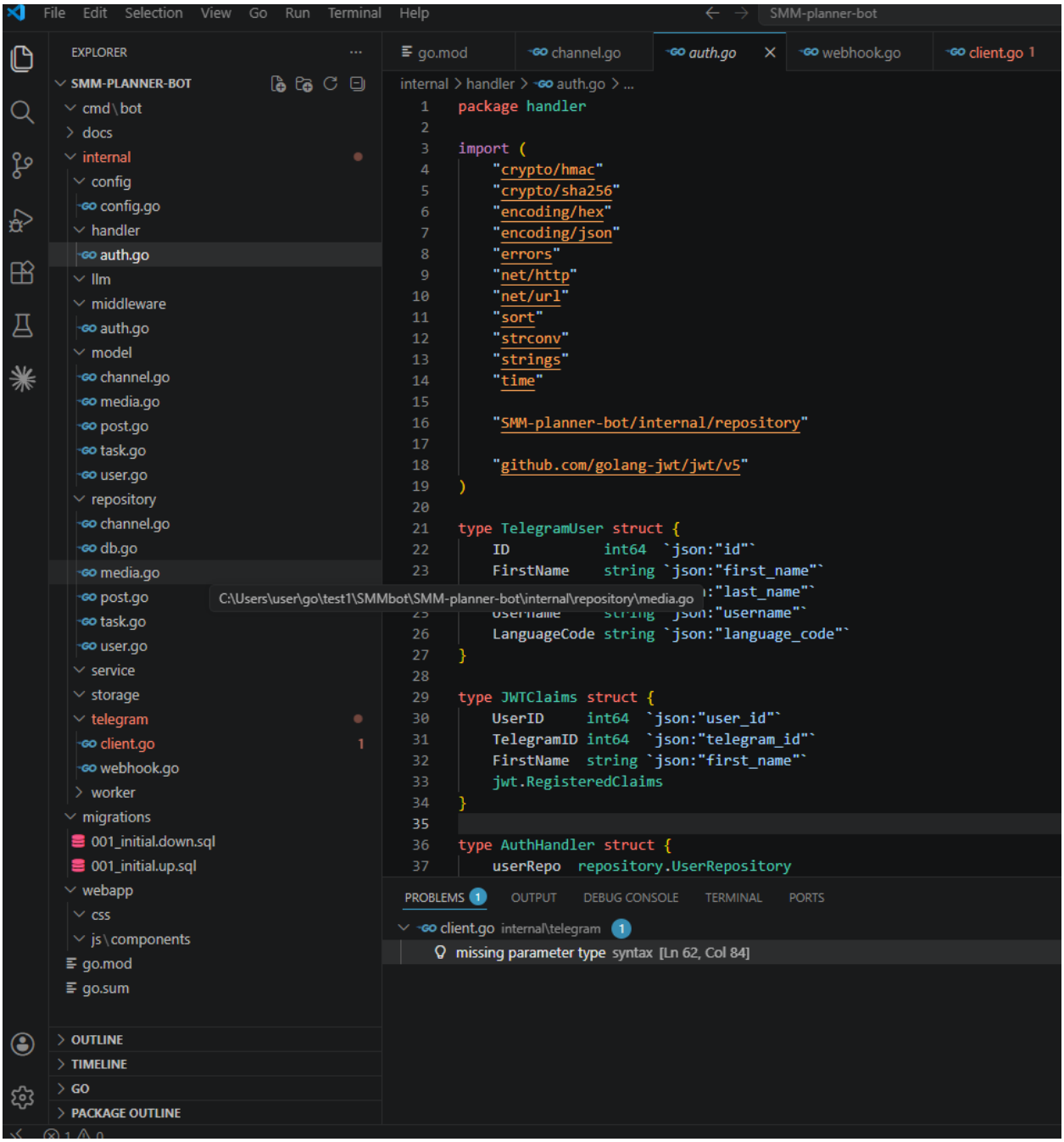

Рисунок 3.5 - Реалізація обробника автентифікації користувача

З точки зору реалізації, архітектура може бути представлена у вигляді багаторівневої системи. На верхньому рівні знаходиться інтерфейс взаємодії з користувачем, середній рівень представлений логікою обробки запитів, а

нижній рівень включає роботу з даними та планування подій. Такий підхід дозволяє розділити відповідальність між компонентами та підвищити ефективність розробки.

Перевагою обраної архітектури є її гнучкість. У майбутньому до системи можуть бути додані нові модулі, наприклад інтеграція з соціальними мережами або модуль аналітики. Це не потребуватиме суттєвих змін у вже існуючих компонентах (рис. 3.6).

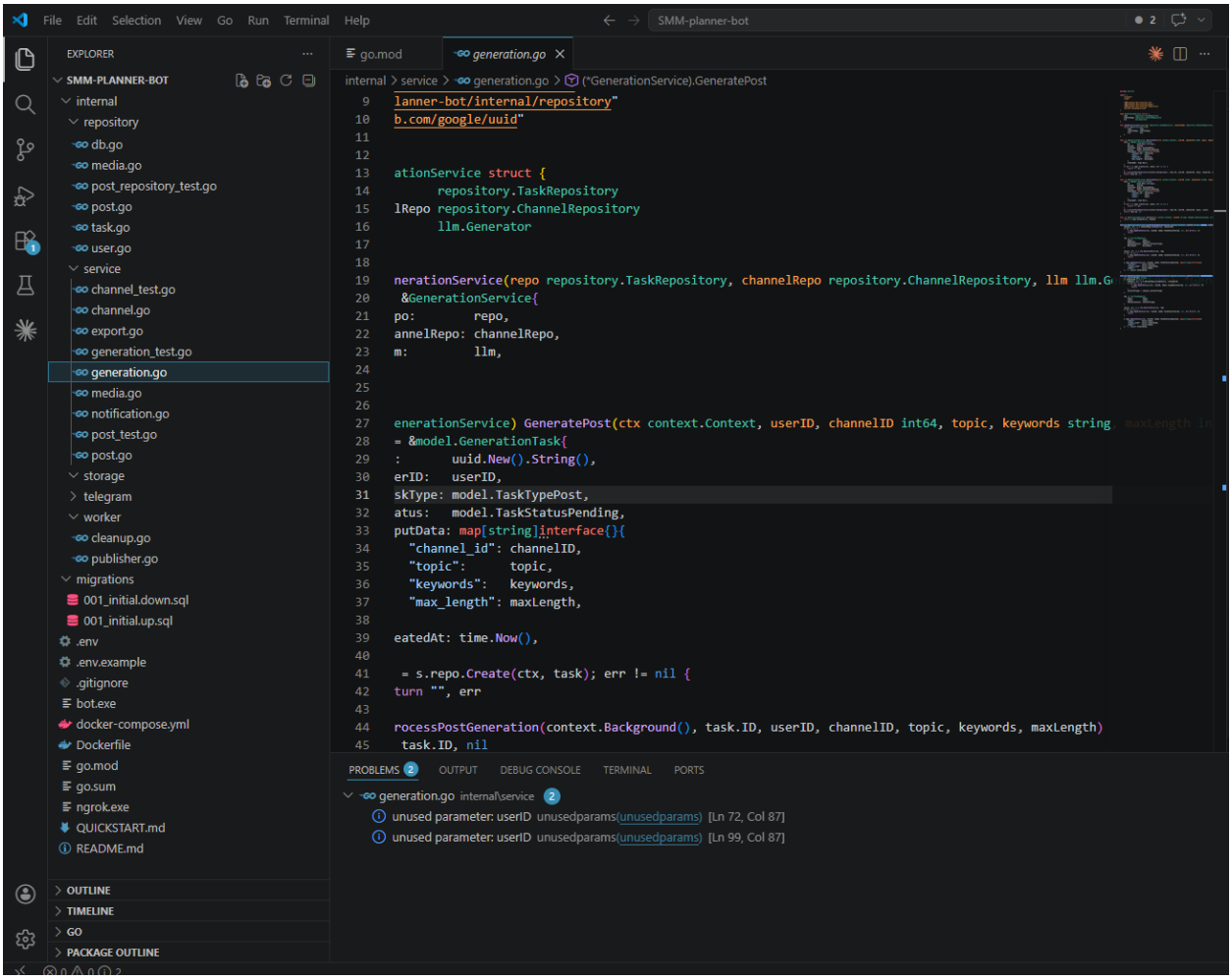

Рисунок 3.6 - Реалізація сервісу генерації контенту

Крім того, модульна архітектура сприяє повторному використанню коду та спрощує процес тестування. Кожен модуль може бути протестований окремо, що підвищує надійність системи в цілому.

Таким чином, обрана архітектура програмного продукту забезпечує ефективну організацію роботи Telegram-бота, дозволяє реалізувати необхідний функціонал та створює основу для подальшого розвитку системи.

3.3 Реалізація основних модулів Telegram-бота

Реалізація Telegram-бота для автоматизації процесів планування та публікації контенту в SMM передбачає створення кількох основних програмних модулів, які забезпечують взаємодію з користувачем, обробку команд, збереження даних та організацію нагадувань. У межах роботи було використано мову програмування Go та бібліотеку aiogram, яка дозволяє реалізувати асинхронну взаємодію з Telegram Bot API.

Основним модулем системи є файл запуску бота, у якому виконується ініціалізація об’єктів бота та диспетчера, підключення обробників команд і запуск процесу очікування повідомлень користувача. Приклад реалізації наведено у лістингу 3.1.

Лістинг 3.1 Ініціалізація та запуск Telegram-бота:

```
import asyncio
from aiogram import Bot, Dispatcher, types
from aiogram.filters import Command
from aiogram.types import ReplyKeyboardMarkup, KeyboardButton
from datetime import datetime, timedelta
import sqlite3

BOT_TOKEN = "YOUR_TELEGRAM_BOT_TOKEN"

bot = Bot(token=BOT_TOKEN)
dp = Dispatcher()

def create_database():
    connection = sqlite3.connect("content_plan.db")
    cursor = connection.cursor()

    cursor.execute("""
        CREATE TABLE IF NOT EXISTS posts (
```

```
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        user_id INTEGER NOT NULL,
        text TEXT NOT NULL,
        publish_time TEXT NOT NULL,
        is_notified INTEGER DEFAULT 0
    )
"""")

connection.commit()
connection.close()
```

У наведеному фрагменті коду створюється об’єкт Bot, який забезпечує зв’язок із Telegram, а також об’єкт Dispatcher, що відповідає за обробку вхідних повідомлень. Функція create_database() створює базу даних SQLite, у якій зберігаються записи контент-плану.

Створення бази даних виконується автоматично під час першого запуску програмного продукту. Такий підхід спрощує процес розгортання системи та не потребує додаткового налаштування з боку користувача. Усі необхідні таблиці формуються програмно, що забезпечує готовність системи до роботи відразу після запуску.

Обрана структура таблиці дозволяє зберігати основні параметри запланованих публікацій. Для кожного запису передбачено унікальний ідентифікатор, ідентифікатор користувача, текст повідомлення, дату та час публікації, а також службове поле для контролю надсилання нагадувань. Така структура є достатньою для реалізації основного функціоналу системи та може бути розширена у майбутньому за потреби.

Для забезпечення зручної взаємодії з користувачем було реалізовано головне меню бота. Воно містить основні команди, які відповідають функціональним можливостям системи: додавання публікації, перегляд контент-плану та видалення записів.

Лістинг 3.2 Реалізація головного меню бота:

```
main_menu = ReplyKeyboardMarkup(
    keyboard=[
        [KeyboardButton(text="Додати публікацію")],
```

```
        [KeyboardButton(text="Переглянути контент-план")],
        [KeyboardButton(text="Видалити публікацію")]
    ],
    resize_keyboard=True
)

@dp.message(Command("start"))
async def start_command(message: types.Message):
    await message.answer(
        "Вітаю! Я Telegram-бот для планування SMM-контенту. "
        "Оберіть потрібну дію в меню.",
        reply_markup=main_menu
    )
```

Реалізація головного меню є важливою складовою інтерфейсу програмного продукту. Використання кнопок дозволяє зменшити кількість помилок під час введення команд та забезпечує більш комфортну взаємодію користувача із системою. Завдяки цьому користувачеві не потрібно запам’ятовувати синтаксис команд або перелік доступних функцій.

Використання клавіатури Telegram також позитивно впливає на швидкість роботи з ботом. Найбільш поширені функції завжди знаходяться у швидкому доступі, що підвищує зручність використання програмного продукту та скорочує час виконання типових операцій.

Після надсилання команди /start користувач отримує вітальне повідомлення та меню з основними діями. Такий підхід спрощує роботу з ботом і не потребує запам’ятовування складних команд.

Для додавання публікації реалізовано обробку повідомлення користувача у визначеному форматі. Наприклад, користувач може ввести текст публікації та дату публікації у форматі: Текст публікації | 2026-05-10 14:30.

Одним із центральних елементів функціональності Telegram-бота є механізм додавання нових записів до контент-плану. Саме цей модуль забезпечує накопичення інформації про майбутні публікації та формує основу для роботи системи нагадувань. У процесі реалізації було

передбачено перевірку правильності введених даних та їх автоматичне збереження у базі даних.

Алгоритм роботи модуля складається з декількох послідовних етапів: отримання повідомлення від користувача, розділення введених даних на окремі складові, перевірка формату дати та часу, формування SQL-запиту та запис інформації до бази даних.

Лістинг 3.3 Додавання публікації до контент-плану:

```
@dp.message(lambda message: message.text == "Додати публікацію")
async def add_post_instruction(message: types.Message):
    await message.answer(
        "Введіть публікацію у форматі:\n"
        "Текст публікації | YYYY-MM-DD HH:MM\n\n"
        "Наприклад:\n"
        "Запустити нову акцію в Instagram | 2026-05-10 14:30"
    )


@dp.message(lambda message: "|" in message.text)
async def save_post(message: types.Message):
    try:
        post_text, publish_time = message.text.split("|", 1)
        post_text = post_text.strip()
        publish_time = publish_time.strip()

        datetime.strptime(publish_time, "%Y-%m-%d %H:%M")

        connection = sqlite3.connect("content_plan.db")
        cursor = connection.cursor()

        cursor.execute(
            "INSERT INTO posts (user_id, text, publish_time) VALUES (?, ?, ?)",
            (message.from_user.id, post_text, publish_time)
        )

        connection.commit()
        connection.close()

        await message.answer("Публікацію успішно додано до контент-плану.")
```

```
    except ValueError:
        await message.answer(
            "Помилка формату. Введіть дані у форматі:\n"
            "Текст публікації | YYYY-MM-DD HH:MM"
        )
```

Цей модуль дозволяє користувачу швидко отримати список усіх запланованих публікацій. Дані сортуються за датою та часом, що забезпечує зручність перегляду.

Для видалення публікацій використовується ідентифікатор запису. Користувач переглядає список публікацій, знаходить потрібний ID та надсилає команду у форматі delete ID.

Для підтримки актуальності контент-плану реалізовано механізм видалення записів. Наявність такої функції є необхідною, оскільки в процесі роботи користувач може змінювати свої плани або відмовлятися від публікації певних матеріалів. У результаті виникає потреба у швидкому видаленні неактуальних записів.

Під час реалізації було передбачено використання унікального ідентифікатора запису, що дозволяє однозначно визначати необхідну публікацію незалежно від її змісту або часу створення.

Лістинг 3.4 Видалення публікації з контент-плану:

```
@dp.message(lambda message: message.text == "Видалити публікацію")
async def delete_post_instruction(message: types.Message):
    await message.answer(
        "Введіть команду у форматі:\n"
        "delete ID\n\n"
        "Наприклад:\n"
        "delete 3"
    )


@dp.message(lambda message: message.text.startswith("delete"))
async def delete_post(message: types.Message):
    try:
        post_id = int(message.text.split()[1])
```

```
            connection = sqlite3.connect("content_plan.db")
            cursor = connection.cursor()

            cursor.execute(
                "DELETE FROM posts WHERE id = ? AND user_id = ?",
                (post_id, message.from_user.id)
            )

            connection.commit()
            deleted_count = cursor.rowcount
            connection.close()

            if deleted_count:
                await message.answer("Публікацію успішно видалено.")
            else:
                await message.answer("Публікацію з таким ID не знайдено.")

        except (IndexError, ValueError):
            await message.answer("Помилка. Введіть команду у форматі: delete ID")
```

Після виконання операції видалення система перевіряє результат SQL-запиту та визначає кількість змінених записів. Якщо запис із вказаним ідентифікатором існує, користувач отримує повідомлення про успішне завершення операції. У протилежному випадку формується повідомлення про відсутність відповідного запису.

Реалізація подібного механізму дозволяє забезпечити додатковий контроль за діями користувача та мінімізувати ймовірність випадкового видалення або виникнення неоднозначних ситуацій під час роботи із системою.

Окремим важливим модулем є модуль нагадувань. Він періодично перевіряє базу даних і надсилає користувачу повідомлення, якщо час нагадування настав.

Однією з ключових функцій розробленого Telegram-бота є автоматичне надсилання нагадувань про наближення часу публікації. Саме ця можливість забезпечує практичну цінність програмного продукту та дозволяє користувачу своєчасно виконувати заплановані дії.

Для реалізації даної функції використано окремий фоновий процес, який регулярно перевіряє наявність записів, для яких необхідно сформувати повідомлення-нагадування. Такий підхід дозволяє забезпечити автономність роботи системи та мінімізувати участь користувача в контролі графіка публікацій.

Лістинг 3.5 Реалізація модуля нагадувань:

```
async def check_reminders():
    while True:
        connection = sqlite3.connect("content_plan.db")
        cursor = connection.cursor()

        current_time = datetime.now()
        reminder_time = current_time + timedelta(minutes=30)

        cursor.execute("""
            SELECT id, user_id, text, publish_time
            FROM posts
            WHERE is_notified = 0
        """)

        posts = cursor.fetchall()

        for post in posts:
            post_id, user_id, text, publish_time = post
            post_datetime = datetime.strptime(publish_time, "%Y-%m-%d %H:%M")

            if current_time <= post_datetime <= reminder_time:
                await bot.send_message(
                    user_id,
                    f"Нагадування!\n\n"
                    f"Скоро потрібно опублікувати:\n{text}\n\n"
                    f"Час публікації: {publish_time}"
                )

                cursor.execute(
                    "UPDATE posts SET is_notified = 1 WHERE id = ?",
                    (post_id,)
                )

        connection.commit()
```

```
        connection.close()

        await asyncio.sleep(60)
```

Реалізований механізм нагадувань працює циклічно та виконує перевірку бази даних через встановлений проміжок часу. У результаті забезпечується постійний контроль за станом запланованих публікацій та своєчасне інформування користувача.

Додатковою перевагою запропонованого рішення є запобігання повторному надсиланню повідомлень. Після формування нагадування відповідний запис позначається як оброблений, що виключає дублювання сповіщень та підвищує зручність використання системи.

У наведеному коді перевірка нагадувань виконується кожні 60 секунд. Якщо до часу публікації залишилося не більше 30 хвилин, бот надсилає повідомлення користувачу та змінює статус запису, щоб уникнути повторного нагадування.

Завершальним етапом є запуск бота та паралельний запуск модуля нагадувань.

Після реалізації окремих модулів необхідно забезпечити їх спільну роботу в межах єдиного програмного продукту. Для цього створюється головна функція запуску, яка виконує ініціалізацію бази даних, запуск механізму нагадувань та активацію процесу обробки повідомлень користувачів.

Саме на цьому етапі окремі компоненти системи об’єднуються у цілісне програмне рішення та забезпечують реалізацію всіх передбачених функціональних можливостей.

Лістинг 3.6 Запуск системи:

```
async def main():
    create_database()

    asyncio.create_task(check_reminders())

    await dp.start_polling(bot)
```

```
if __name__ == "__main__":
    asyncio.run(main())
```

Таким чином, реалізовані основні модулі Telegram-бота забезпечують виконання ключових функцій програмного продукту: додавання публікацій до контент-плану, перегляд запланованих записів, видалення публікацій та організацію автоматичних нагадувань. Використання модульного підходу дозволяє у подальшому розширювати функціональність системи, зокрема додавати редагування записів, категоризацію контенту, інтеграцію з соціальними мережами або аналітичний модуль.

3.4 Реалізація функцій планування та публікації контенту

Реалізація функцій планування та публікації контенту є основним етапом створення програмного продукту, оскільки саме ці функції забезпечують виконання поставленої мети автоматизації SMM-процесів. У межах розробленого Telegram-бота реалізовано комплекс програмних механізмів, що дозволяють створювати контент-план, зберігати інформацію про майбутні публікації, формувати нагадування та забезпечувати зручну взаємодію користувача із системою [2, 9].

Основною задачею розробленого програмного продукту є спрощення процесу роботи з контентом у соціальних мережах. У традиційному підході користувач змушений самостійно вести облік запланованих публікацій, контролювати строки їх розміщення та пам’ятати про необхідність підготовки нових матеріалів. Такий підхід потребує значних витрат часу та створює ризик пропуску важливих публікацій. Автоматизація зазначених процесів дозволяє зменшити навантаження на користувача та забезпечити більш ефективну організацію роботи з контентом [1].

Робота бота розпочинається з отримання команди від користувача через інтерфейс Telegram. Після отримання відповідного запиту система

запускає необхідний сценарій обробки, який визначається типом обраної функції. Для забезпечення коректної роботи реалізовано механізм маршрутизації команд, що дозволяє розподіляти запити між окремими функціональними компонентами програмного продукту [12].

Механізм маршрутизації є важливим елементом програмної реалізації, оскільки саме він визначає подальший алгоритм роботи системи. Кожне повідомлення, отримане від користувача, аналізується програмою, після чого визначається відповідний обробник, який відповідає за виконання конкретної функції. Такий підхід дозволяє забезпечити структурованість програмного коду та уникнути дублювання логіки обробки запитів.

Використання окремих обробників для кожної функції також спрощує процес модернізації програмного продукту. У разі необхідності додавання нових можливостей достатньо створити новий модуль або функцію та підключити її до загальної системи маршрутизації. Завдяки цьому забезпечується масштабованість системи та можливість її подальшого розвитку.

Однією з основних функцій системи є створення записів контент-плану. У процесі додавання нового запису користувач послідовно вводить текст майбутньої публікації, дату її розміщення та час публікації. Отримані дані проходять перевірку на коректність, після чого зберігаються у базі даних. Завдяки цьому забезпечується цілісність інформації та мінімізується ймовірність виникнення помилок під час подальшого використання системи [18].

На етапі введення даних особливу увагу приділено контролю правильності введеної інформації. Система перевіряє наявність усіх необхідних полів, правильність формату дати та часу, а також логічну коректність введених параметрів. Наприклад, дата публікації не може бути меншою за поточну дату, а час повинен відповідати встановленому формату. У випадку виявлення помилки користувач отримує відповідне повідомлення та можливість повторного введення інформації [18].

Після проходження перевірки дані перетворюються у внутрішній формат представлення та передаються до модуля збереження. У системі кожен запис має власний ідентифікатор, що дозволяє виконувати подальші операції редагування та видалення без ризику втрати інформації.

Для роботи з контент-планом реалізовано функції перегляду, редагування та видалення записів. Користувач може отримати доступ до актуального списку запланованих публікацій, переглянути необхідну інформацію та внести зміни у разі потреби. Усі зміни автоматично зберігаються в системі та стають доступними для подальшої обробки [12, 18].

Функція перегляду контент-плану забезпечує швидкий доступ до всієї інформації про майбутні публікації. Після відповідного запиту користувач отримує впорядкований список записів із зазначенням дати, часу та змісту публікації. Такий підхід дозволяє швидко оцінити поточний стан контент-плану та визначити необхідність внесення змін.

Функція редагування дозволяє змінювати параметри вже створених записів. Користувач може оновити текст повідомлення, дату або час публікації, а також інші параметри, що використовуються системою. Після внесення змін програма автоматично оновлює інформацію в базі даних та перераховує параметри нагадувань.

Не менш важливою є функція видалення записів. Вона забезпечує очищення контент-плану від неактуальних або помилково створених публікацій. Після підтвердження операції відповідний запис видаляється з бази даних та більше не бере участі в роботі системи.

Важливим компонентом програмного продукту є механізм організації нагадувань. Після створення нового запису система автоматично розраховує час надсилання повідомлення та додає відповідну подію до планувальника задач. У визначений момент бот надсилає користувачу нагадування про наближення часу публікації. Такий підхід дозволяє забезпечити своєчасне виконання запланованих дій та підвищує ефективність роботи з контентом [12].

Механізм нагадувань функціонує автономно та не потребує постійної участі користувача. Після створення запису система самостійно контролює наближення запланованого часу та виконує всі необхідні дії для формування повідомлення. Це дозволяє користувачу зосередитися на підготовці контенту, не витрачаючи додатковий час на контроль графіка публікацій.

Застосування автоматичних нагадувань є особливо корисним для користувачів, які працюють із великою кількістю контенту або ведуть декілька інформаційних ресурсів одночасно. У таких умовах ручний контроль за строками розміщення публікацій є складним та може призводити до помилок. Автоматизація цього процесу значно підвищує ефективність роботи.

Для забезпечення стабільної роботи Telegram-бота використано асинхронну модель обробки подій. Це дозволяє одночасно обслуговувати запити користувачів та виконувати фонові операції без блокування основного процесу роботи. Завдяки цьому система демонструє високу швидкодію навіть під час одночасного виконання декількох задач [12].

Використання асинхронного підходу є важливим з точки зору продуктивності програмного продукту. Під час роботи бота можуть одночасно виконуватися різні операції: отримання повідомлень від користувачів, звернення до бази даних, формування нагадувань, генерація контенту та експорт інформації. Асинхронна модель дозволяє виконувати всі ці операції незалежно одна від одної, що позитивно впливає на швидкодію системи.

Додатковою функцією програмного продукту є генерація ідей для майбутніх публікацій. Користувач може обрати напрям або тематику контенту, після чого система формує відповідні пропозиції. Реалізація цієї можливості дозволяє спростити процес підготовки матеріалів та прискорити формування контент-плану [4].

Однією з найпоширеніших проблем у сфері SMM є пошук нових тем для публікацій. Навіть за наявності чіткої маркетингової стратегії

користувачі часто стикаються з труднощами під час підготовки контенту. Саме тому функція генерації ідей має практичну цінність та дозволяє зменшити витрати часу на пошук нових тем.

Після вибору категорії або напряму система формує перелік можливих ідей, які можуть використовуватися як основа для створення нових матеріалів. Отримані результати можуть бути безпосередньо використані під час формування контент-плану або доопрацьовані користувачем відповідно до його потреб.

Також у системі реалізовано можливість експорту сформованих публікацій. Завдяки цьому користувач може зберігати підготовлений контент для подальшого використання або перенесення до інших програмних засобів. Наявність такої функції підвищує практичну цінність програмного продукту та розширює можливості його застосування [12].

Механізм експорту дозволяє використовувати результати роботи Telegram-бота незалежно від конкретного програмного середовища. Підготовлений контент може бути перенесений до інших систем управління соціальними мережами, текстових редакторів або корпоративних платформ. Це створює додаткові можливості інтеграції програмного продукту у вже існуючі бізнес-процеси.

Крім того, функція експорту сприяє збереженню результатів роботи користувача та створює умови для резервного копіювання інформації. У разі необхідності користувач може швидко відновити раніше підготовлений контент або використати його в іншому середовищі без необхідності повторного введення даних.

У процесі реалізації особливу увагу було приділено забезпеченню простоти використання програмного продукту. Усі функції доступні через звичний інтерфейс месенджера Telegram і не потребують встановлення додаткового програмного забезпечення або спеціальної підготовки користувача [12].

Інтерфейс системи побудований таким чином, щоб навіть користувачі без технічних знань могли швидко освоїти роботу з ботом. Використання кнопок, меню та зрозумілих повідомлень дозволяє зробити взаємодію максимально простою та інтуїтивною.

Таким чином, реалізовані функції планування та публікації контенту забезпечують автоматизацію основних SMM-процесів, дозволяють ефективно організовувати роботу з контентом, своєчасно отримувати нагадування, формувати нові ідеї для публікацій та спрощують процес підготовки матеріалів для розміщення у соціальних мережах [2, 26].

3.5 Приклади роботи розробленого Telegram-бота

Для перевірки працездатності розробленого програмного продукту було проведено тестування основних функцій Telegram-бота. У процесі тестування перевірялася коректність обробки команд користувача, створення записів контент-плану, відображення запланованих публікацій, надсилання нагадувань та робота механізмів керування контентом.

Після запуску бота користувач отримує вітальне повідомлення та доступ до основних функцій системи. Інтерфейс реалізовано у форматі діалогу, що забезпечує простоту взаємодії та не потребує спеціальної підготовки користувача (рис. 3.7).

Однією з основних функцій системи є створення нового запису контент-плану. Для цього користувач послідовно вводить текст майбутньої публікації, дату та час її розміщення. Отримані дані проходять первинну перевірку на коректність та відповідність встановленому формату. Такий підхід дозволяє зменшити кількість помилок під час роботи з системою та забезпечити достовірність інформації, що зберігається у контент-плані.

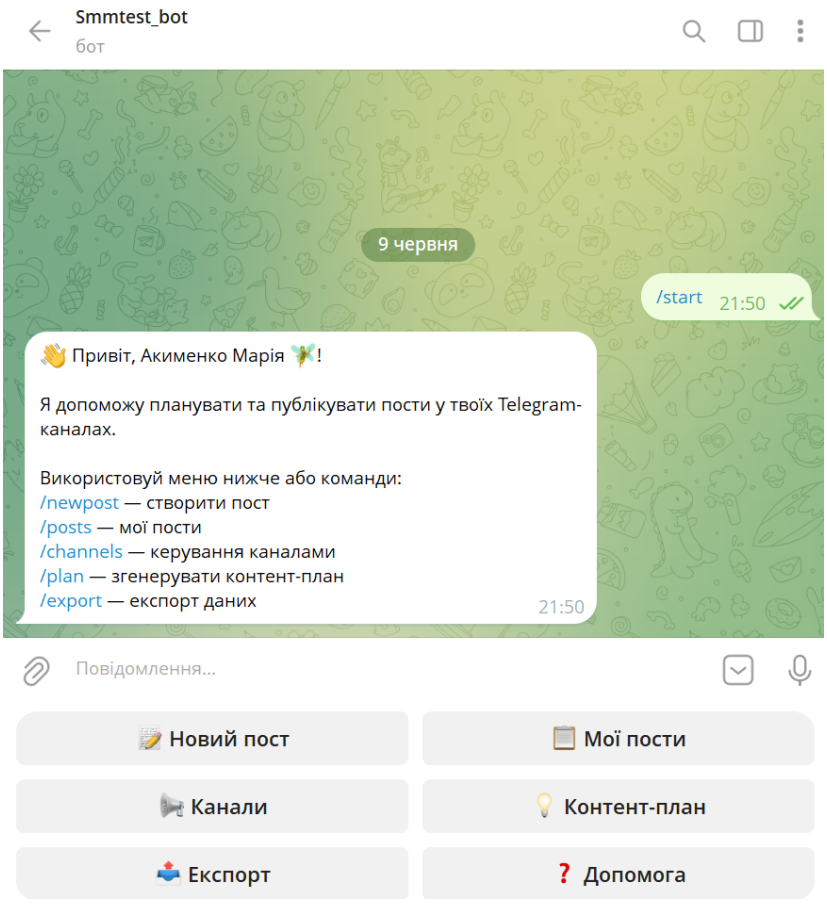

Рисунок 3.7 - Початковий інтерфейс Telegram-бота

Після отримання всіх необхідних даних бот формує запис та зберігає його у системі (рис. 3.8).

Рисунок 3.8 - Процес створення запису контент-плану

Після успішного додавання інформації система формує повідомлення про успішне збереження запису.

Це дозволяє користувачу переконатися у правильності виконання операції та уникнути втрати даних (рис. 3.9).

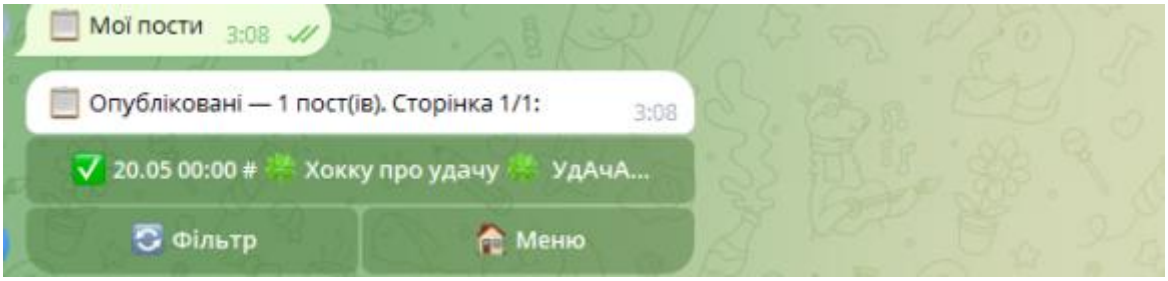

Рисунок 3.9 - Підтвердження створення запису

Однією з додаткових можливостей розробленого Telegram-бота є автоматизоване формування ідей для майбутніх публікацій. Користувач має можливість обрати напрям або тематику контенту, після чого система генерує варіанти ідей, які можуть бути використані під час наповнення контент-плану. Такий підхід дозволяє спростити процес підготовки матеріалів, пришвидшити пошук нових тем для публікацій та підвищити ефективність роботи SMM-фахівця (рис. 3.10).

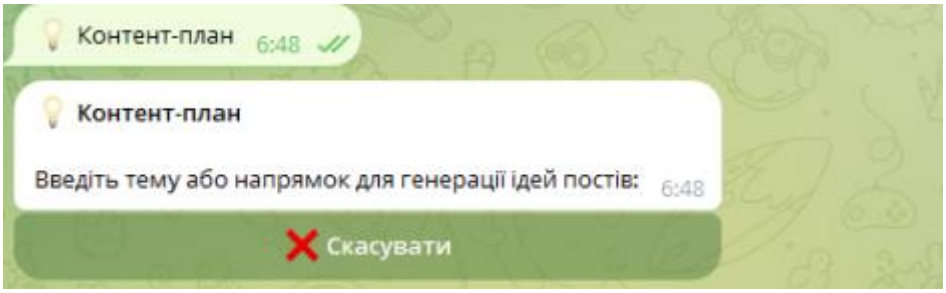

Рисунок 3.10 - Формування ідеї для публікації за заданим напрямом

Важливою функцією розробленого Telegram-бота є можливість експорту сформованих публікацій для їх подальшого використання. Після створення або редагування контенту користувач може виконати експорт записів у зручному форматі для збереження, подальшого опрацювання або розміщення на інших платформах. Така можливість спрощує роботу з контентом, забезпечує його перенесення між різними сервісами та підвищує зручність використання програмного продукту (рис. 3.11).

Рисунок 3.11 - Експорт сформованих публікацій

Результати проведеного тестування підтвердили коректність роботи основних функцій розробленого програмного продукту. Під час перевірки було протестовано всі ключові сценарії взаємодії користувача з Telegram-ботом, включаючи створення нових записів контент-плану, збереження інформації до бази даних, перегляд раніше сформованих публікацій, редагування та видалення записів. У процесі тестування система продемонструвала стабільну роботу та коректне виконання всіх передбачених операцій.

Особливу увагу було приділено перевірці механізму обробки користувацьких даних. Результати показали, що бот успішно виконує перевірку правильності введених параметрів, контролює формат дати та часу публікації, а також забезпечує збереження інформації без втрати даних. У випадках введення некоректних значень система формує відповідні повідомлення про помилки та пропонує користувачу повторне введення інформації, що позитивно впливає на надійність програмного продукту.

Окремо було перевірено роботу модуля нагадувань, який є одним із найважливіших компонентів системи. У ході тестування підтверджено правильність розрахунку часу надсилання повідомлень та коректність роботи механізму планування подій. Telegram-бот своєчасно надсилає користувачу повідомлення про наближення запланованої публікації та автоматично оновлює статус відповідного запису після виконання нагадування. Це дозволяє мінімізувати ризик пропуску важливих публікацій та підвищує ефективність роботи з контентом.

Позитивні результати також були отримані під час тестування додаткових функціональних можливостей системи. Зокрема, підтверджено коректну роботу механізму генерації ідей для майбутніх публікацій, а також функції експорту підготовленого контенту. Реалізовані можливості дозволяють розширити сферу практичного використання програмного продукту та зробити його більш зручним для користувачів, які займаються веденням сторінок у соціальних мережах.

Під час проведення тестування не було виявлено критичних помилок, які могли б вплинути на працездатність системи або призвести до втрати даних користувача. Усі основні функціональні модулі працюють узгоджено та забезпечують виконання поставлених завдань відповідно до визначених вимог. Це свідчить про правильність обраних архітектурних рішень, технологій реалізації та програмних засобів, використаних у процесі розробки.

Таким чином, результати тестування підтвердили працездатність і надійність розробленого Telegram-бота. Програмний продукт успішно реалізує функції планування контенту, організації нагадувань, керування записами та підтримки користувача під час роботи з контент-планом. Отримані результати свідчать про можливість практичного використання системи для автоматизації процесів планування та публікації контенту в соціальних мережах, а також створюють передумови для подальшого розширення функціональних можливостей програмного продукту.

ВИСНОВКИ

У процесі виконання кваліфікаційної роботи розглянуто теоретичні та практичні аспекти автоматизації процесів планування та публікації контенту в соціальних мережах у межах SMM-діяльності. Встановлено, що SMM є важливим інструментом цифрового маркетингу, який забезпечує ефективну взаємодію між брендом і цільовою аудиторією. Визначено, що процеси планування та публікації контенту характеризуються складністю, потребують значних часових витрат та містять велику кількість рутинних операцій, що обумовлює необхідність їх автоматизації.

Проаналізовано існуючі програмні засоби автоматизації SMM-процесів. Встановлено, що сучасні сервіси забезпечують широкий спектр функціональних можливостей, проте часто характеризуються складністю використання, високою вартістю або наявністю надлишкового функціоналу. Це підтвердило доцільність створення спеціалізованого програмного продукту, орієнтованого на автоматизацію основних процесів планування контенту та організації нагадувань.

Сформульовано вимоги до програмного продукту та визначено основні функціональні можливості системи. Обґрунтовано доцільність використання Telegram-бота як інструменту автоматизації завдяки його доступності, простоті використання та можливостям інтеграції з користувачем через звичне середовище месенджера.

Розроблено логіку функціонування системи, алгоритм формування контент-плану, алгоритм організації нагадувань і керування публікаціями, алгоритми взаємодії користувача із системою та сценарії обробки даних. Запропоновані алгоритмічні рішення забезпечують впорядкованість процесів, коректність роботи системи та можливість подальшого розширення її функціональних можливостей.

Для реалізації програмного продукту обрано мову програмування Python, бібліотеку aiogram для взаємодії з Telegram Bot API та базу даних

SQLite для збереження інформації про заплановані публікації. Запропоновано модульну архітектуру системи, яка забезпечує гнучкість, масштабованість та зручність супроводу програмного забезпечення.

Створено Telegram-бот для автоматизації процесів планування та публікації контенту. Реалізовано функції додавання, перегляду, редагування та видалення записів контент-плану, а також механізм автоматичного надсилання нагадувань про заплановані публікації. Розроблений програмний продукт забезпечує зручну взаємодію з користувачем та дозволяє підвищити ефективність організації роботи з контентом.

Практичне значення отриманих результатів полягає у можливості використання створеного програмного продукту для підтримки діяльності SMM-фахівців, контент-менеджерів, підприємців та інших користувачів, які здійснюють планування контенту в соціальних мережах. Використання Telegram-бота дозволяє скоротити витрати часу на виконання рутинних операцій, підвищити організованість роботи та зменшити ймовірність пропуску важливих публікацій.

Перспективи подальшого розвитку полягають у реалізації автоматичної публікації контенту, інтеграції з популярними соціальними мережами, розширенні аналітичних можливостей системи та вдосконаленні механізмів взаємодії з користувачем.

Таким чином, мету кваліфікаційної роботи досягнуто, а поставлені завдання виконано в повному обсязі.

Результати кваліфікаційної роботи апробовано у вигляді 2 тез доповідей під час 30-го Міжнародного молодіжного форуму «Радіоелектроніка і молодь у XXI столітті» [12] та XVIII Міжнародної науково-практичної конференції «Digital Transformation: Science in Search of Answers to Global Questions» [13].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Chaffey D., Ellis-Chadwick F. (2019), Digital Marketing: Strategy, Implementation and Practice. 7th ed. Harlow : Pearson,. 816 p.

2. Tuten T. L., Solomon M. R. (2020), Social Media Marketing. 4th ed. London : Sage Publications,. 432 p.

3. Kaplan A. M., Haenlein M. (2010), Users of the world, unite! The challenges and opportunities of social media. Business Horizons.. Vol. 53, No. 1. P. 59-68.

4. Kotler P., Keller K. L. (2019), Marketing Management. 15th ed. Pearson Education,. 832 p.

5. Зозульов О. В. (ред.). (2021), Маркетинг у соціальних мережах. Київ : КПІ ім. Ігоря Сікорського,. 312 с.

6. Окландер М. А. (2020), Цифровий маркетинг. Київ : Центр учбової літератури,. 292 с.

7. Довгий С. О. (ред.), (2022), Хмарні технології та цифрова трансформація. Київ : НЦ «Мала академія наук України»,. 248 с.

8. Пономаренко В. С., Золотарьова О. В. (2019), Інформаційні системи та технології в управлінні підприємством. Харків : ХНЕУ ім. С. Кузнеця,. 320 с.

9. Kumar V., (2012), Mirchandani R. Increasing the ROI of social media marketing. MIT Sloan Management Review.. Vol. 54, No. 1. P. 55-61.

10. Ryan D. (2016), Understanding Digital Marketing: Marketing Strategies for Engaging the Digital Generation. London : Kogan Page,. 296 p.

11. Gunelius S, (2011), 30-Minute Social Media Marketing. New York : McGraw-Hill,. 272 p.

12. Дам-Васильєва Ч. А. (2026), Автоматизація процесів планування та публікації контенту в SMM. Радіоелектроніка і молодь у XXI столітті : матеріали Міжнародного молодіжного форуму. Харків : ХНУРЕ, 2026. С. 43-44.

13. Дам-Васильєва Ч. А. (2026), Автоматизація планування контенту в соціальних мережах за допомогою Telegram-ботів. Digital Transformation: Science in Search of Answers to Global Questions : Proceedings of the XVIII International Scientific and Practical Conference. Warsaw, 2026. P. 68-69.

14. Творошенко І. С. (2021), Технології прийняття рішень в інформаційних системах : навч. посіб. Харків : ХНУРЕ,. 180 с.

15. Гороховатський В. О., Творошенко І. С. (2021), Методи інтелектуального аналізу та оброблення даних : навч. посіб. Харків : ХНУРЕ, , с. 45-49.

16. Telegram Bot API. URL: https://core.telegram.org/bots/api (дата звернення: 01.06.2026).

17. Go Documentation. URL: https://go.dev/doc (дата звернення: 10.06.2026).

18. Go Packages Reference. URL: https://pkg.go.dev (дата звернення: 11.06.2026).

19. PostgreSQL Documentation. URL: https://www.postgresql.org/docs (дата звернення: 22.05.2026).

20. Docker Documentation. URL: https://docs.docker.com (дата звернення: 14.06.2026).

21. Visual Studio Code Documentation. URL: https://code.visualstudio.com/docs (дата звернення: 03.05.2026).

22. GitHub. URL: https://github.com (дата звернення: 07.05.2026).

23. Stack Overflow. URL: https://stackoverflow.com (дата звернення: 12.06.2026).

24. Hootsuite Blog. URL: https://blog.hootsuite.com (дата звернення: 03.06.2026).

25. Buffer Resources. URL: https://buffer.com/resources (дата звернення: 09.06.2026).

26. Sprout Social Insights. URL: https://sproutsocial.com/insights (дата звернення: 10.05.2026).

27. Later Blog. URL: https://later.com/blog (дата звернення: 13.06.2026).

28. Safko L. (2012), *The Social Media Bible: Tactics, Tools, and Strategies for Business Success*. 3rd ed. Hoboken : John Wiley & Sons,. 624 p.

29. Kietzmann J. H., Hermkens K., McCarthy I. P., (2011), Silvestre B. S. Social media? Get serious! Understanding the functional building blocks of social media. *Business Horizons*.. Vol. 54, No. 3. P. 241-251.

30. Tiago M. T. P. M. B., Veríssimo J. M. C. (2014), Digital marketing and social media: Why bother? *Business Horizons*.. Vol. 57, No. 6. P. 703-708.

31. Project Management Institute. (2021), *A Guide to the Project Management Body of Knowledge (PMBOK® Guide)*. 7th ed. Newtown Square : PMI,. 370 p.

32. Ілляшенко С. М. (2020) *Інноваційний маркетинг*. Суми : Університетська книга,. 334 с.

33. Примак Т. О. (2019), *Маркетингові комунікації в системі управління підприємством*. Київ : КНЕУ, 328 с.