НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

До захисту допущено:
Завідувач кафедри
_____ Сергій СТІРЕНКО
«___»_____2021 р.

Дипломний проект

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Комп'ютерні системи та мережі»

спеціальності 123 «Комп'ютерна інженерія»

на тему: «Система для комунікації в навчальному закладі на основі чат-боту»

Виконав: студент 4 курсу, групи ІВ-72
Тимко Андрій Олексійович _____

Керівник:
Доцент, к.т.н.
Ткаченко Валентина Василівна _____

Консультант з нормоконтролю:
Професор кафедри ОТ, д.т.н.
Сімоненко Валерій Павлович _____

Рецензент: _____

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2021 року

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 123 «Комп'ютерна інженерія»

Освітньо-професійна програма

«Комп'ютерні системи та мережі»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Сергій СТІРЕНКО
«___»_____20__ р.

ЗАВДАННЯ
на бакалаврський дипломний проект студента
Тимку Андрію Олексійовичу

1. Тема проєкту «Система для комунікації в навчальному закладі на основі чат-боту», керівник проєкту Ткаченко Валентина Василівна, доцент, к.т.н., затверджені наказом по університету від «07» травня 2021 р. № 1081-с

2. Термін подання студентом проєкту 01 червня 2021 р.

3. Вихідні дані до проєкту технічна документація. теоретичні та статистичні дані,

4. Зміст пояснювальної записки

5. Перелік графічного матеріалу

6. Консультанти розділів проєкту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Сімоненко В.П., професор, д.т.н.		

7. Дата видачі завдання 29 вересня 2020 р.

Якщо визначені консультанти. Консультантом не може бути зазначено керівника дипломного проєкту.

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Відмітки про виконання
1	*Затвердження теми роботи*	10.12.2021-15.12.2021	
2	*Вивчення та аналіз завдання*	15.12.2021-15.03.2021	
3	*Розробка архітектури та загальної структури програми*	15.03.2021-25.03.2021	
4	*Розробка структур окремих інтерфейсів програми*	25.03.2021-5.04.2021	
5	*Програмна реалізація*	05.04.2021-15.04.2021	
6	*Оформлення пояснювальної записки*	15.04.2021-20.05.2021	
7	*Захист програмного продукту*	25.04.2021	
8	*Передзахист*	16.05.2021	
9	*Захист*	17.06.2021	

Студент Андрій ТИМКО

Керівник Валентинка ТКАЧЕНКО

Анотація

У бакалаврській роботі було спроектовано і розроблено універсальну бот платформу та систему для комунікації у вищому навчальному закладі на основі чат-боту, використовуючи Telegram Bot API та Viber Bot API. Розроблено сервіс для розсилок повідомлень по базі користувачів ботуз можливістю фільтрування аудиторії. Розроблено сервіс для опитувань користувачів з відображенням статистики. Розроблено функціонал для викладачів, студентів та абітурєнтів для полегшення комунікації та доступу до інформації всередені університету.

Програмний продукт був реалізований на мові програмування JavaScript, за допомогою інструменту для візуального програмування Node-RED.

Annotation

In the bachelor's thesis, a universal bot platform and a system for communication in higher education based on a chatbot were designed and developed using the Telegram Bot API and Viber Bot API. Developed a service for sending messages to the user base of boots with the ability to filter the audience. Developed a service for user surveys with statistics. Functionality has been developed for teachers, students and entrants to facilitate communication and access to information within the university.

The software product was implemented in the JavaScript programming language, using the Node-RED visual programming tool.

1ВІДОМІСТЬ ДИПЛОМНОГО ПРОЄКТУ

№ з/п	Формат	Позначення	Найменування	Кількість листів	Примітка
1	А4		Завдання на дипломний проєкт	2	
2	А4	ІАЛЦ.466550.002 ТЗ	Система для комунікації в навчальному закладі на основі чат-боту.Технічне завдання	4	
3	А4	ІАЛЦ.466550.003 ПЗ	Система для комунікації в навчальному закладі на основі чат-боту.Пояснювальна записка	80	
4	А4	ІАЛЦ.466550.004 А1	Схема структурна	1	
5	А4	ІАЛЦ.466550.005 А2	Схема функціональна	1	
6	А4	ІАЛЦ.466550.006 А3	Схема принципова	1	

Технічне завдання
до дипломного проєкту

на тему: «Система для комунікації в навчальному закладі
на основі чат-боту»

Київ – 2021

ЗМІСТ

1. НАЙМЕНУВАННЯ ТА ОБЛАСТЬ ЗАСТОСУВАННЯ

Найменування: «Система для комунікації в навчальному закладі на основі чат-боту». Дане технічне завдання розповсюджується на розробку системи для комунікації для абітурієнтів, студентів та викладачів на основі чат-боту та універсальної бот платформи для Viber та Telegram .

Область застосування: розроблена система може бути застосована в будь-яких навчальних закладах.

2. ПІДСТАВИ ДЛЯ РОЗРОБКИ

Підставою для розробки є завдання на виконання роботи кваліфікаційно-освітнього рівня «бакалавр комп'ютерної інженерії», затверджене кафедрою обчислювальної техніки Національного технічного Університету України «Київський Політехнічний інститут ім. Ігоря Сікорського».

3. МЕТА ТА ПРИЗНАЧЕННЯ РОЗРОБКИ

Метою розробки є система для комунікації, щоб полегшити взаємодію між абітурієнтами ,студетами та викладачами з університетом. А також, щоб облегшити доступ до необхідної інформації.

4. ДЖЕРЕЛА РОЗРОБКИ

Джерелами для розробки служать науково-технічна література з комп’ютерних технологій, публікації в періодичних виданнях, публікації в Інтернеті за даним питанням.

5. ТЕХНІЧНІ ВИМОГИ

5.1. Вимоги до програмного продукту, що розробляється

- Можливість реєстрації в чат-боті.
- Можливість ідентифікації викладачів.
- Відображення різного функціоналу для кожної ролі.
- Можливість проведення розсилок по базі користувачів.
- Можливість проведення опитувань та показ статистики по ній.
- Можливість фільтрування аудиторії
- Можливість переглянути розклад

5.2. Вимоги до програмного забезпечення серверу

- Операційна система Windows, Linux;
- БД MongoDB;
- Мова програмування Java Script.

6. ЕТАПИ РОЗРОБКИ

	Дата
Вивчення літератури	30.03.2021
Складання і узгодження технічного завдання	01.04.2021
Розробка структур окремих інтерфейсів програми	01.04.2021-15.04.2021
Програмна реалізація	16.04.2021-12.05.2021
Тестування окремих модулів системи	13.05.2021
Допрацювання і виправлення помилок	14.05.2021
Оформлення документації дипломної роботи	20.05.2021

Пояснювальна записка
до дипломного проєкту

на тему: «Система для комунікації в навчальному закладі на основі чат-боту»

Київ – 2021

ЗМІСТ

ПЕРЕЛІК ТЕРМІНІВ ТА СКОРОЧЕНЬ

ОС	Операційна система
БД	база даних
ЄДРПОУ	Єдиний державний реєстр підприємств та організацій України
API	Прикладний програмний інтерфейс (англ. Application Programming Interface, API)
JSON	Текстовий формат обміну даними між комп'ютерами, який дозволяє описувати об'єкти та інші структури даних (англ. JavaScript Object Notation)
HTTP	(англ. HyperText Transfer Protocol) – протокол, який дозволяє отримувати різні ресурси, наприклад HTML-документи
URL	(англ. Uniform Resource Locator) – система унікальних адрес електронних ресурсів

ВСТУП

Поява мережі Інтернет вплинула на розвиток комунікації. Спочатку його використували для отримування та передачі різного виду інформації, але з часом мережа Інтернет все частіше брала на себе роль комунікатора.

В останні роки спілкування у формі віртуальної комунікації все частіше входить в наше життя і формує нову сферу інформаційної взаємодії, тим самим витісняючи живе спілкування. Люди перестали бути прив'язаним до місця розташування співрозмовника.

Технології для обміну миттєвими повідомленнями увійшли в життя кожної прогресивної людини. Глобальна мережа Інтернет стала невід'ємною частиною спілкування, роботи, навчання, відпочинку, шопінгу, розваг. Останнім часом завойовують популярність технології, засновані на месенджерах та чат-ботах.

IT-технології, маркетинг, менеджмент і прогрес не стоять на місці, тому їх розвиток сприяв появі чат-ботів в звичному нам розумінні. В сучасному світі дуже багато компаній активно створюють і впроваджують подібні технології в рамках своєї діяльності. Чат-боти можуть використовуватись не тільки для вирішення рутинних задач, таких як банальний збір данних та їх обробка, але і для аналітики та підтримки користувача.

Чат-боти в компаніях покликані спростити взаємодію співробітників з інформаційними системами, тим самим підвщующи їх ефективність роботи і функціонування компанії в цілому. Використовуючі подібні технологічні інновації, вдається підвищити залученість колективу в роботу, тим самим підвищити стабільність компанії, полегшуючи структуру інформування та управління. Чат-боти значно знижують зайве інформаційне навантаження, дозволяє заощадити час, а також підвищити ефективність роботи всередині компанії.

На теперішній час, використання чат-ботів приносить нові можливості для бізнесу і надає цілий ряд переваг: швидка відповідь на запити; можна знайомити з продуктом шляхом текстових і фото повідомлень; залучення нових клієнтів рекламуючи чат-бот; проводити опитування серед клієнтів і покупців; виконувати функції технічної підтримки; вивчати аудиторію і їхні вподобання.

Але потрібно розуміти що фунціонал чат-ботів обмежений і вони не зможуть зрівнятися з мобільними додатками чи веб-сайтами.

РОЗДІЛ 1
ЧАТ-БОТИ ЯК ОДИН ІЗ ШЛЯХІВ ВИРІШЕННЯ ПРОБЛЕМИ КОМУНІКАЦІЇ В ВИЩИХ НАВЧАЛЬНИХ ЗАКЛАДАХ

1.1. Проблеми комунікації в вищих навчальних закладах

Проблема взаємодії студентів та викладачів завжди була актуальна. Вирішити цю проблему дуже важко, тому що немає універсального рецепту. На мій погляд, ефективність навчального процесу багато в чому залежить від вміння викладача організувати вдале спілкування зі студентом, а також від бажання студента здобувати знання. Відсутність взаєморозуміння може привести до серйозних проблем в вивчанні предмету.Тому навіть якщо викладач має серйозну професійну підготовку, володіє методикою викладання предмету, ефект навчання може бути мінімальним через складні відносини між студентом та викладачем. Щоб успішно налагодити контакт зі студентом, потрібно розуміти його психологію.

Також проблема комунікації полягає в складності доступу до потрібної інформації. Щоб отримати необхідну методичку або книжку необхідно перерити безліч інтернет ресурсів. Наразі немає єдиного сховища с необхідною літературою, яка необхідна студенту для навчання.

Абутурієнти стикаються з проблемою комунікації найбільше. Для абітурієнта найважливішою інформацією є перелік необхідних документів для вступу, правила поселення в гуртожиток, розположення корпусів, адреси важливих місць. Наразі немає єдиного ресурсу для отримання цією інформації, тому потрібно згаяти досить часу, щоб отримати ті чи інші дані.

Для оптимізації всіх цих процесів, потрібно розробити єдину платформу, де можна буде отримати всю необхідну інформацію абітурієнтам, студентам та викладачам. Це може бути веб-сайт, мобільний додаток або чат-бот.

1.2. Чат-бот, як один з варіантів вирішення проблеми комунікації

На мою думку чат-бот є найбільш зручним варіантом, так як більшість людей проводять в месенджерах більшість свого часу, на відміну від мобільного додатку який потрібно окремо завантажувати або веб-сайту.

В наші дні чат-боти використовуються практично у всіх сферах діяльності: від електронної комерції до промисловості і сфери освіти. Використання чат ботів у сфері освіти стає популярною тенденцію і уже створено безліч чат-ботів, що сприяє отриманню і закріпленню знань, а також перевірці їх засвоєння.

Існує дослідження ефективності використання чат-ботів в освітньому процесі, в ході якого встановлено, що більшість учасників експерименту вчились, засвоювали інформацію і спілкувались з ботами так само, якби це були справжні люди.

Чат-боти мають безліч переваг перед використанням інших ресурсів і, зокрема, мобільних додатків: боти легко встановити, не використовуючи пам'ять пристрою, посилання на бот набагато легше розповсюдити, чат-бот легше створити і використовувати.

Однак одним із основних факторів, через які чат-боти стали настільки популярні , є поширення месенджерів. Адже спілкування сучасної людини відбувається практично завжди в месенджері. Ці чат-продукти встановлені в смартфонах практично у всіх.

Месенджер – це програма, яка дозволяє миттєво обмінюватися з друзями текстовими повідомленнями, телефонними дзвінками і навіть розмовляти по відеозв'язку. За данними 2020 року найбільш популярними месенджерами в Україні є Viber, Facebook Messenger та Telegram.

Таким чином, перед педагогами освітніх закладів відкриваються нові можливості встановлення швидкого контакту зі студентами і передачі їм інформації в зжатому вигляді. Звичайно не кожен освітній процес може бути переведений в формат спілкування з чат-ботом, але вже існує безліч ботів для отримання короткої структурованої інформації. Наявність такого чат-боту в смартфоні у студента значно полегшить отримання знань, покращить комунікацію між студентом та педагогом, а також полегшить отримання необхідної інформації.

1.3. Telegram та його можливості

Незваючи на те, що за статистикою найбільш популярним месенджером є Viber, проте серед молоді набуває популярності Telegram. Тому, на мою думку, для студентів буде зручніше і доцільніше, використовувати саме його.

Telegram — це веб додаток для обміну миттєвими повідомленнями з акцентом на швидкість і безпеку, розроблений на мові програмування C++. Це швидкий, простий, безпечний і безкоштовний сервіс. Він легко синхронізується на всіхпристроях, працює як на персональних комп'ютерах, так і на планшетах і мобільних телефонах. За допомогою цього месенджера можна відправляти необмежену кількість повідомлень, фотографій, відео та файлів будь-якого типу.

Програма розроблена Павлом Дуровим – російським програмістом та одним із засновників та авторів популярної соціальної мережі Вконтакте. В месенджері застосований спеціальний протокол шифрування MTProto – власна розробка Миколи Дурова. Згідно з даним протоколом, захист застосовується як для передачі повідомлень так і для авторизації користувача. Безпека месенджера підримується на такому високому рівні також за рахунок того, що сервери, де зберігаються користувацькі дані, розташовані в різних місцях планети: Лондон, Сінгапур, Сан-Франциско.

Сервіс надається користувачам безкоштовно, так як його розробники вклали в цей проект власні гроші, задля можливості вільного і ено коштів, а весь приплив клієнтів відбувається лише за рахунок рекомендбезпечного спілкування. Цікаво те, що на рекламу цього сервісу не потрачацій користувачів.

Що можна робити в Telegram? Даний месенджер не просто дозволяє миттєво обмінюватись повідомленнями з іншими користувачами сервісу. Його функціонал набагато більший. Додаток постійно оновлюється і поповнюється новими можливостями.

Telegram має ряд переваг:

- Висока швидкість роботи – незважаючи на те, що сервери знаходяться в різних куточках планети, доставка повідомлень займає долі секунди.
- Зрозумілий інтерфейс,який можна змінювати в налаштуваннях.
- Проста реєстрація. Необіхдний лише номер телефону.
- Всі функції і можливості безкоштовні.
- Високий рівень безпеки і захисту від взлому акаунту та листування.
- Сумісність зі всіма основними операційними системами.

Серед недоліків можна виділити обмеженість десктоп та веб версії Telegram. На жаль досить багато функцій які працюють в мобільному додатку не працюють в браузері або на комп'ютері. Наприклад видалити історію листування або створити секретний чат.

Також є декілька особливостей месенджера, які зустрічаються не у всіх його аналогах, серед яких:

- Можливість відправляти файли розміром до 1.5 Гб.
- Використання унікальних імен. Можна знайти людину в боті, не знаючи його номер телефону.
- Запис голосових та відео повідомлень
- Створення власних каналів для перегляду
- Пошук по чату. Можна знайти листування, знаючи лише його частину.
- Створення і використання власних стікерів.
- Поділ чатів на категорії
- Використання автоматизованих віртуальних помічників – чат-ботів.

Месенджер Telegram є в доступі на всіх популярних платформах. Приклад інтерфейсу месенджера для мобільного додатку IPhone та десктопної версії на Windows 10 зображені на рисунках 1.1 і 1.2.

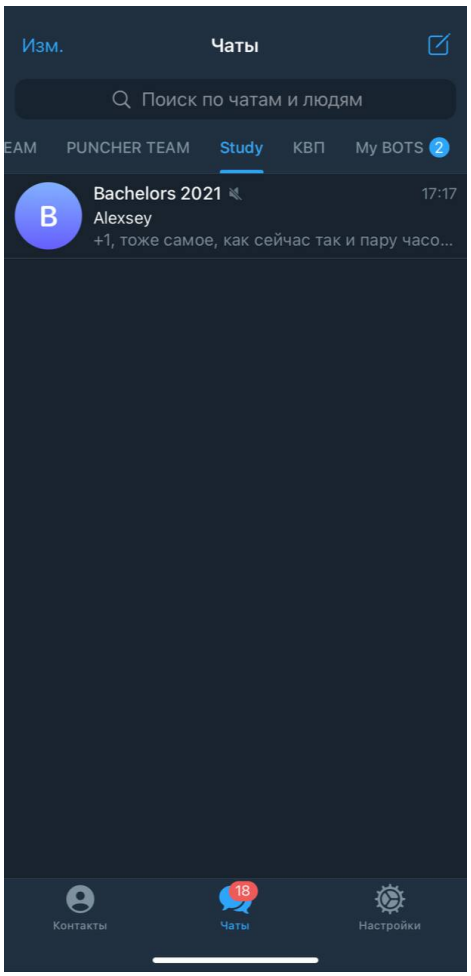


Рис. 1.2 Інтерфейс Telegram на Windows 10

Як бачимо інтерфейс мобільного додатку досить зрозумілий , зверху знаходиться поле для пошуку по чатам. Під ним знаходяться папки, які поділяють всі чати на категорії. Знизу знаходяться кнопки з контаками, чатами та налаштування.

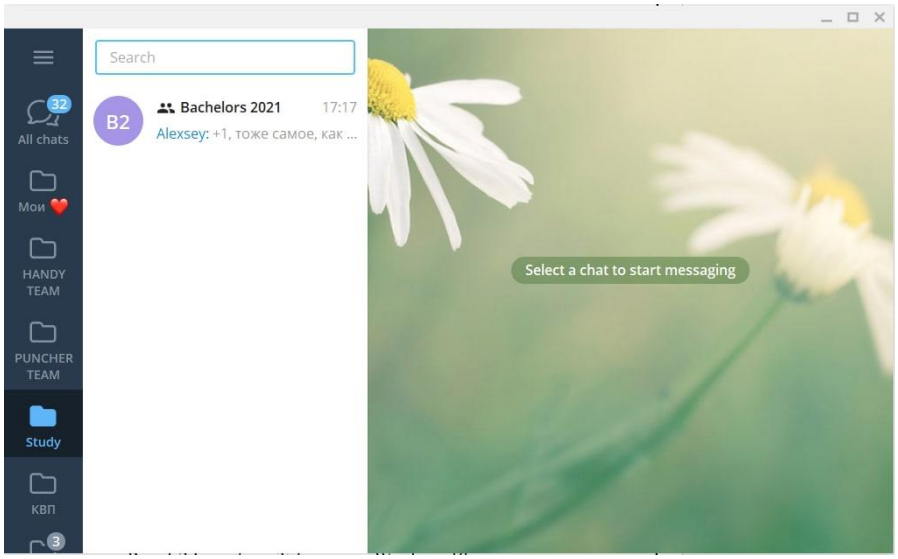


Рис. 1.2 Інтерфейс Telegram на Windows 10

1.4. Приклади чат-ботів Telegram

1.4.1. Сервіс пошуку та моніторингу квитків RailwayBot

RailwayBot – чат-бот для моніторингу та купівлі квитків на потяги в межах України. Сервіс допомагає користувачам швидко та зручно замовити квиток на потяг. За 2020 рік кількість користувачів ботов досягло відмітки у 200 000.

Окрім пошуку та купівлі квитків, за допомогою боту можна моніторити наявність квитків за багатьма критеріями, оплачувати купівлю банківською картою, Apple Pay або Privat24, отримати розклад руху потягів, повертати квитки та інформувати про зупинку під час маршруту. Процес вибору квитків на потяг забражені на рисунку 1.3.

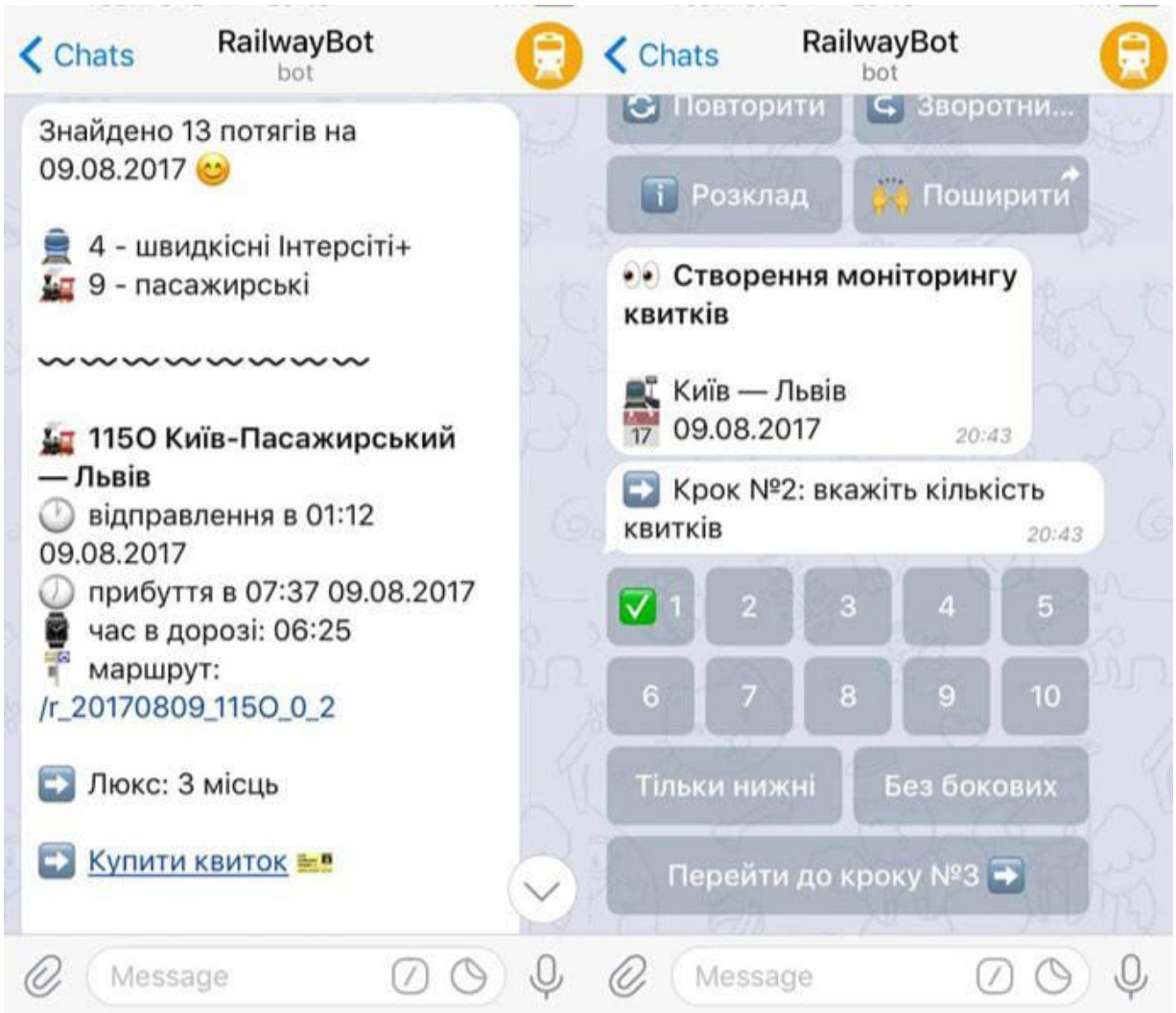


Рис. 1.3 Вибір потягу та квитка

Для того щоб придбати квиток необхідно:

1.Вказати свій номер телефону

2. Вказати звідки і куди потрібен квиток

3.Обрати дату виїзду

4.Серед списку доступних потягів обрати потрібний

5.Обрати клас вагону

6.Обрати місце

7.Вибрати тип квитка (звичайний, студентський, дитячий)

8.Написати прізвище та ім’я

9.Оплатити квиток

Всього 9 кроків і квиток вже куплений. Сервіс набирає популярності за рахунок своєї доступності, простоти використання та економії часу.

1.4.2. Чат-бот для клієнтів Приватбанку PrivatBankBot

PrivatBankBot – чат-бот, основними функціями якого є переказ та отримання коштів на карти фізичних осіб в один клік. Функціонал доступний лише для клієнтів Приватбанку.

Для того щоб операції проходили успішно необхідно щоб отримувач і відправник були зареєстровані в боті. Для реєстрації необхідно лише вказати номер телефону, номер карти та вказати код, який прийшов в смс.

Після реєстрації в боті користувач отримує головне меню (рис. 1.4).

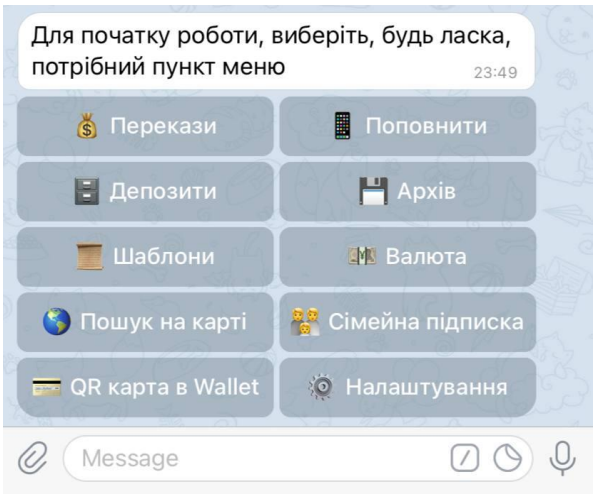


Рис. 1.4 Головне меню бота

Окрім переказу коштів, бот надає безліч інших корисних можливостей:

1. Поповнення мобільного телефону
2. Оформити депозит
3. Переглянути архів переказів за останній місяць
4. Створити шаблон переказу
5. Перегляд курсу валют та можливість купити чи продати валюту
6. Знайти найближчі відділення чи банкомати Приватбанку
7. Створити збір коштів

1.5 Viber та його можливості

Viber – месенджер, який дозволяє безкоштовно обмінюватися повідомленнями та використовувати прямі дзвінки без використання стандартних ліній зв'язку, а маючи лише доступ до інтернету. Це найпопулярніший месенджер на території України, який можна використовувати як на телефоні так і на комп'ютері. В додатку наявні безкоштовні функції, які можна налаштувалти в декілька кліків.

Для того щоб користуватися месенджером на телефоні, необхідно встановити безкоштовну програму із магазину додатків на телефоні. Додаток підходить як для гаджетів з ОС Android так і для IOS.

Наступним етапом після встановлення додатку є реєстрація. Для того щоб зареєструватися необхідно лише поділитися своїм номером та підтвердити його шляхом вводу коду, який прийде в повідомленні.

Після успішної реєстрації, бот готовий до використання, а його ярлик з'явиться на робочому столі телефону. Після відкриття програми, можна побачити стандартне вікно програми, яке містить декілька вкладок, розташованих внизу у вигляді кнопок.

Першою графою в головному вікні месенджера є закладка «Чати». В ній відображається все листування користувача з іншими людьми. Чати можна видалити або закріпити зверху, якщо це щось важливе.

Чат можна створювати для спілкування двух користувачів, а також запрошувати до нього групу користувачів для спілкування. В ньому кожен учасник може відслідковувати статус доставки своїх повідомлень, а також дивитися коли повідомлення буде прочитане співрозмовником. Окрім текстових повідомлень, користувачі можуть відправляти одне одному стікери, зображення, анімації, голосові повідомленяя та смайлики. Вигляд головного екрану можна переглянути на рис.1.5.

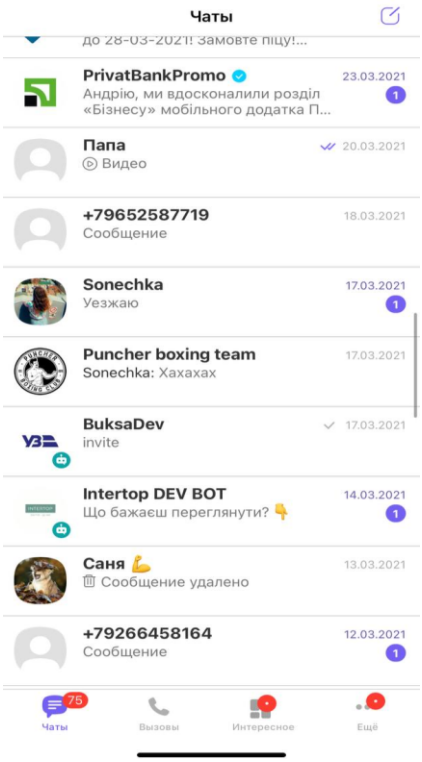


Рис. 1.5 Головний екран Viber

Список нещодавніх викликів можна переглянути за допомогою кнопки «Виклики». При переході на цю вкладку можна побачити останні відео та аудіодзвінки, які користувач здійснив або прийняв. Напроти кожного виклику вказано час дзвінка та тривалість розмови. По бажанню під час виклику, користувач може додавати нових співрозмовників.

Трохи нижче можна побачити список всіх контактів ,які занесені в телефонну книгу користувача, в алфавітному порядку. Програма може автоматично верифікувати номера, на пристроях яких встановлений месенджер.

Viber пропонує своїм користувачам велику кількість функціоналу, що дозволяє обмунюватися текстовими повідомленнями і файлами, вести групові чати, здійснювати аудіо та відеовиклики та робити грошові перекази. Всі ці опції отримали безліч позитивних відгуків та користуються популярністю серед користувачів. Також в месенджер додали можливість грати в ігри і змагатися зі своїми друзями.

Розробники месенджера попіклувалися про те, щоб всі особисті повідомлення користувачів були надійно захищені від взлому, зберігаючи приватність дзвінків та чатів. Основні харктеристики безпеки додатку вказані нижче:

- Всі способи зв'язку в месенджері захищені наскрізним шифруванням, тому кожне слово розмови чи листування відомо лише співрозмовникам;
- Користувачі програми захищені від розсилок персоналізованої реклами, так як ніхто не має доступу до особистих повідомлень і дзвінків абонента;
- Всі повідомлення видаляються з серверів відразу після відправки отримувачу;

Додаток Viber і популярні соціальні мережі мають відмінності, але даний месенджер має унікальні особливості, що роблять його експлуатацію більш вигідною.Серед основних переваг месенджера є:

- Відмінна якість зв'язку та звуку під час дзвінків;
- Відсутня плата за повідомлення та дзвінки при наявності інтернет підключення;
- Простота встановлення програми;
- Зручний інтерфейс месенджера;
- Наявність безлічі функцій;
- Відсутня необхідність в авторизації;
- Надійне шифрування дзвінків та листування;

Незважаючи на те, що месенджер позиціонує себе як конфіденційний та захищений месенджер, у нього є ряд серйозних недоліків, пов'язаних саме з беспекою. До них відноситься той факт, що дані користувачів можуть передавати провоохоронним органам та іншим особам. Однак ті, кого це не бентежить, найдуть для себе в цьому месенджері все необхідне, включаючи стікери та відеодзвінки.

1.6 Приклади чат-ботів Viber

1.6.1 Бот-рекрутер для пошуку роботи «Робота в Інтертоп»

Цей чат-бот був створений для того щоб прискорити та автоматизувати процес прийому на роботу, а також уникнути масових офлайн співбесід. Так як через пандемію проведення масових інтерв'ю неможливе. Таким чином, за допомогою цього чат-боту термін подачі та розгляду заявки значно скоротився.

Після переходу в чат-бот користувач бачить повідомлення з короткою інформацією про бот, та перший крок реєстрації (рис.1.6).

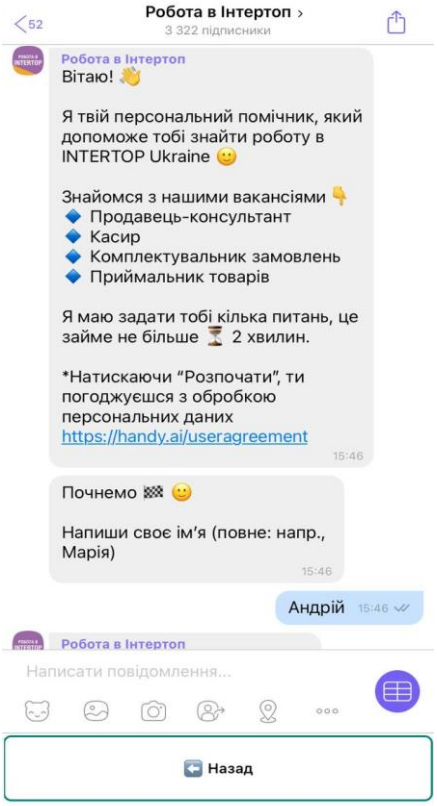


Рис. 1.6 Привітання чат-боту та перший крок реєстрації

Процес подачі анкети займає не більше 3 хвилин. Чат-бот збирає всю необхідну інформацію:

- Особиста інформація (ім’я, прізвище, вік, телефон);
- Інформація для рекрутера (досвід роботи, бажана зарплатня та графік);
- Вибір міста, вакансії та магазину;
- Вибір дати та часу співбесіди;
- Додаткова інформація (фото та коментар) ;

Після того як користувач пройшов етап реєстрації, йому відображається головне меню (рис.1.7), а анкета відправляється рекрутеру, де він повинен відповісти прийнята анкета чи ні. Якщо кандидату відмовили, йому прийде повідомлення з відмовою, а якщо анкету схвалено, прийде нагадування про співбесіду.

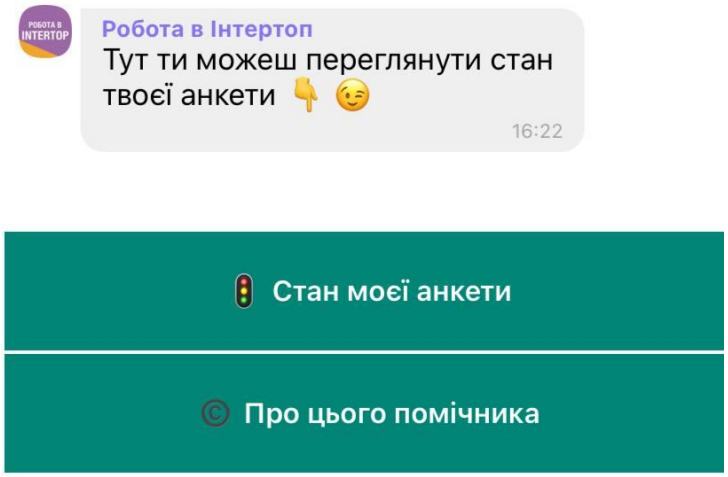


Рис. 1.7 Головне меню бота «Робота в Інтертоп»

1.6.2 Чат-бот для передачі показань електролічильника «Yasno»

Даний бот служить помічником людям та служить для таких задач:

- Передача показань електролічильника
- Сплата щомісячних рахунків
- Замовлення додаткових послуг
- Налаштування нагадувань про передачу показань
- Перегляд архіву передачі показань

Даний бот значно спрощує процес подачі показань та сплати разунків за електроенергію. Чат-ботом вже користується більш ніж 300 тисяч людей, що говорить про його популяність. В головному меню бота (рис.1.8) чотири головних кнопки:

- Кнопка «Передати показання» служить для подачі показань електролічильника. Для цього необхідно ввести номер особового рахунку, а потім ввести дані лічильника. Також в цьому пункті меню можна зробити нагадування, і чат-бот буде надсилати вам сповіщення тоді, коли треба подавати показання.
- Кнопка «Енергоофіси» допоможе вам знайти найближчий офіс або отримати повний список енргоофісів у потрібному вам місті всього за декілька кліків. Ця функція значно пришвидшує пошук адрес та контактних даних відділень при необіхдності.
- Кнопка «Мої особові рахунки» дозволяє додати, видалити або відредагувати номер особового рахунку. Можливість мати декілька особових номерів дозволяє сплачувати рахунки не тільки за себе, а й за батьків чи інших родичів.
- Кнопка «Газ» має декілька функцій, а саме можливість замовити консультацію і телефонному режимі або за допомогою чату, а також подати документи для подачі документів на газ.

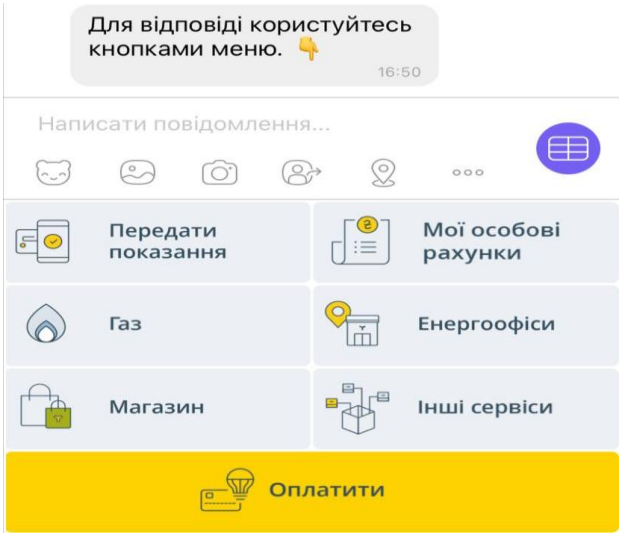


Рис. 1.8 Головне меню бота «Yasno»

1.7 Етапи розробки чат-бота

Кожного дня створюється безліч чат-ботів і кожен з них виконує свої функції: замовлення їжі, влаштування на роботу, технічна підтримка. Але перед тим як створити будь-який бот необхідно пройти декілька обов'язкових етапів.

Визначення кола обов'язків та цілей чат-боту. Не існує універсальних ботів які здатні відповісти на будь-яке питання. Якісні віртуальні співрозмовники виконують обмежене коло задач, з якими відмінно справляються. Адже чим більше задач, тим більше некоректних відповідей. Тому необхідно чітко встановити цілі, які чат-бот буде виконовути як для компанії, так і для користувача.

Вибір платформи. Чат-боти можуть спілкуватися с користувачами через різн канали: мобільного зв'язку, месенджерів, соціальних мереж чи інтегрованих в ресурси чати. Від правильно вибраної платформи залежить чи зможе чат-бот залучити нових клієнтів та підвищити лояльність постійних клієнтів.

Розробка комунікаційної архітектури. Необхідно продумати схему та стиль діалогу, детально проробити дизайн повідомлень, щоб спілкування не виглядало штучним. Незважаючи на те що користувачі повинні знати, що вони спілкуються з роботом, а не людиною, чат-бот повинен притримуватися максимально природнього діалогу. Необхідно уникати складної граматики, вести діалог коротко та ясно, та додати боту трішки характеру, щоб він мав конкретний образ.

Розробка інтеграцій. Якщо чат-бот включає в себе зв'язок з іншими сервісами, потрібно розробити схему інтеграцій.

Розробка боту та тестування. Після того як розробка боту закінчена, необхідно добре протестувати всі можливі сценарії і вивчити поведінку боту в різних випадках. Також необхідно протестувати як веде себе бот при вводі некоректних відповідей. Після цього, бот можна активувати та вводити в експлуатацію.

ВИСНОВОК ДО РОЗДІЛУ 1

У роботі проаналізована проблема комінікації в навчальних закладах, визначено причини та способи подолання цієї проблеми. Обгрунтовано, що чат-боти є досить перспективним напрямком на сьогоднішній день і є чудовим вирішенням проблеми комунікації.

Розглянуто месенджери Telegram та Viber , їх характеристики та особливості. Також наведені приклади застосування чат-ботів у різних сферах бізнесу. Визначено етапи розробки віртуальних помічників.

РОЗДІЛ 2
ВИБІР ТЕХНОЛОГІЙ І СЕРЕДОВИЩА РОЗРОБКИ

2.1 Технології для розробки проекту

2.1.1 Telegram Bot API

Bot API – це інтерфейс на основі HTTP, який створений для роботи з чат-ботами Telegram.

Кожному боту при створенні присвоюється унікальний токен, який складається з 42 символів. Токен потрібен для віправки запитів. Всі запити до Telegram Bot API повинні виконуватись через HTTPS.

Для передачі параметрів в Bot API досупно 4 способи: запит в URL, application/x-www-form-urlencoded, application/json, multipart/form-data. Можливі як GET так і POST запити. Відповідь приходить в форматі JSON-об'єкту.

Всі запити запити в Telegram створюються по шаблону: https://api.telegram.org/bot<token>/НАЗВА_МЕТОДУ. Метод API – дія яку повинна виконати система у вказаному боті.

Розглянемо найпопулярніші методи Telegram Bot API та дії які вони виконують.

- getMe – повертає основну інформацію про бот у вигляді об'єкта. Не вимагає додаткових параметрів;
- sendMessage – метод для відправки тексту та клавіатури. Потребує два обов'язкових параметра: chat_id та text;
- sendPhoto – метод для відправки фото. Обов'язкові параметри: chat_id та photo;
- sendAudio – метод для відправки аудіо;
- sendContact – метод для того щоб поділитися контактом;
- sendVideo - метод для відправки відео;

Це лише маленька частина всіх доступних методів, які використовуються в Telegram Bot API.

Для того щоб отримувати повідомлення в бот існує 2 абсолютно різних за своєю логікою способи:

1. Метод getUpdates – бот з певною регулярністю запитує у сервера чи є нові повідомлення.
2. Webhook – сервер Telegram сам сповіщує додаток як тільки з'являються оновлення.

Схема взаємодії користувача та чат-боту зображена на рисунку 2.1.

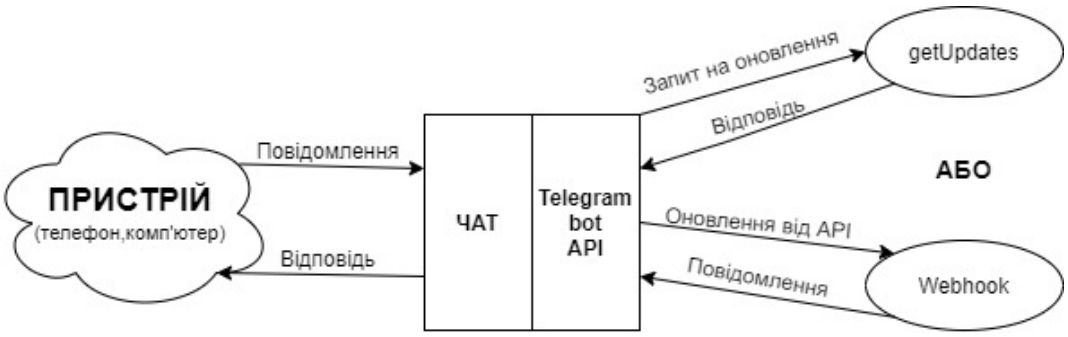


Рис. 2.1 Схема роботи чат-боту

2.1.2 Viber Bot API

Viber REST API - прикладний програмний інтерфейс для роботи з Viber і зокрема чат- ботами. Для роботи з Viber API необхідно мати токен авторизації, який надається після створення бота і його можуть преглянути адміністратори акаунта на панелі адміністратора Viber.

Токен – це унікальний таємний ідентифікатор облікового запису і необхідний для автентифікації запиту та запобігання несакціонованим особам надсилати запити від імені бота. Кожен запит API повинен містити заголовок під назвою X-Viber-Auth-Token, що містить маркер автентифікації облікового запису.

Отримавши токен, можна встановити веб-хук для облікового запису. Цей веб-хук буде використовуватися для отримання зворотних дзвінків та повідомлень користувачів від Viber.

Налаштування веб-хука відбувається шляхом виклику API за адресою https://chatapi.viber.com/pa/set_webhook. Ця дія визначає веб-хук облікового запису та тип подій, про які обліковий запис хоче отримувати повідомлення.

В запиті необхідно вказати типи подій на які відправляти зворотній виклик. Існує 3 обов’язкових події які неможливо відфільтрувати: відпарвка повідомлення, нова підписка та скасування підписки. Також є необов’язкові події, які можна підключити по бажанню:початок діалогу, доставка повідомлення, помилка відправки повідомлення та перегляд повідомлення користувачем.

Відправка повідомлень можливе лише користувачам які підписані на бота. Для відправки потрібно надіслати запит за адресою https://chatapi.viber.com/pa/send_message, з необхідними параметрами в залежності від типу повідомлення. API підтримує різні типи повідомлень: текст, зображення, відео, файл, місце розташування, наклейку, контакт, карусель та URL-адресу. Для того щоб було зрозуміло який тип повідомлення необхідно відправити, потрібно вказати в полі type потрібний тип.

Приклад запиту для відправки картинки забражений на рисунку 2.2.

```
{
   "receiver":"01234567890A=",
   "min_api_version":1,
   "sender":{
      "name":"John McClane",
      "avatar":"http://avatar.example.com"
   },
   "tracking_data":"tracking data",
   "type":"picture",
   "text":"Photo description",
   "media":"http://www.images.com/img.jpg",
   "thumbnail":"http://www.images.com/thumb.jpg"
}
```

Рис. 2.2 Запит для відправки картинки в Viber бот

API Viber дозволяє надсилати власну клавіатуру за допомогою API send_message, щоб забезпечити користувача набором заздалегідь визначених відповідей або дій. Клавіатуру можна приєднати до будь-якого типу повідомлення або надіслати, коли вона увімкнена. Отримавши, клавіатура з’явиться перед користувачем замість власної клавіатури пристрою. Клавіатури повністю налаштовуються і можуть бути створені та розроблені спеціально для потреб облікового запису. Клієнт завжди відображатиме останню клавіатуру, яка була йому надіслана.

2.1.3 Node-RED

Node-RED – інструмент для візуального програмування, розроблений на базі Node.js, і тому використовує його подійно-орієнтовану, неблокуючу модель. Даний інструмент дозволяє підключати різні пристрої, API та онлайн-сервіси.

Програмування здійснюється в браузерному редакторі (рис.2.3), де користувачі можуть створювати потоки, підключаючи один до одного ноди різного призначення. Також в наявності текстовий редактор, що дозволяє створювати JavaScript-функції прямо в редакторі Node-RED.

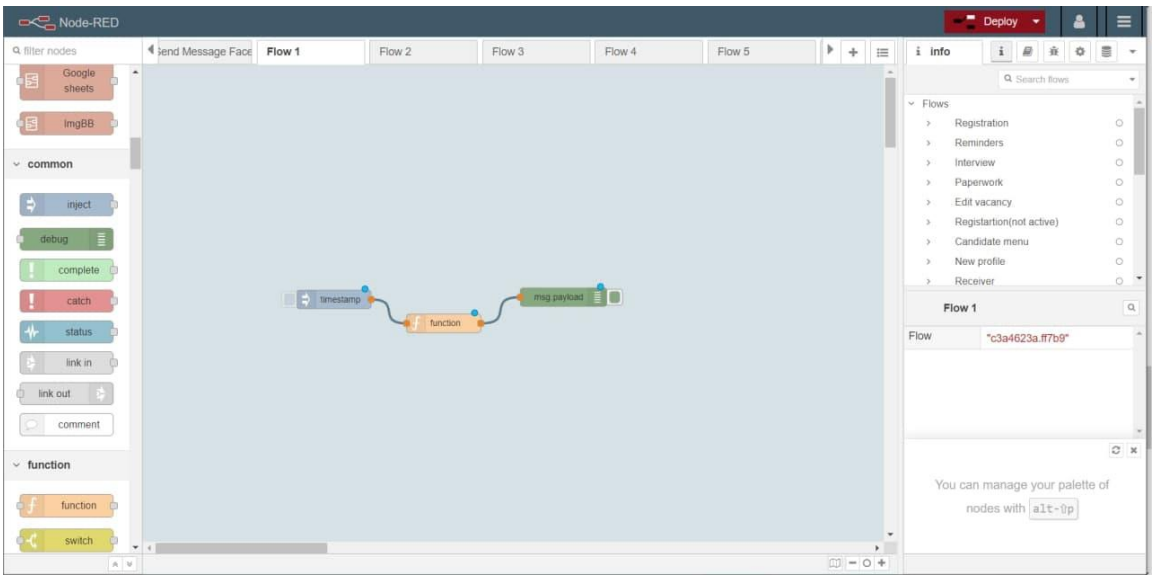


Рис. 2.3 Браузерний редактор Node-RED

Node-RED позиціонується, як система, що об’єднує розрізнені апаратні і програмні системи. В основі роботи системи лежить подія, яка генерується або зовнішнім джерелом, або процесом всередині Node-RED. Подія це не просто імпульс, що дає поштовх до якої-небудь дії. Подія ще несе певне смислове навантаження і може містити безліч корисної інформації, адже подія в Node-RED це звичайний об'єкт мови JavaScript. Цей об'єкт, містить стандартний набір полів, але може розширюватися і модифікуватися за допомогою обробки всередині Node-RED. У базовому налаштуванні є необхідний набір для старту: інтерфейси для підключення до MQTT, набори логіки, парсери форматів, виводи в консоль. Це робить Node-RED дуже гнучким інструментом, що може вирішувати задачі різної тяжкості

Завдяки підтримці багатьох добровольців які розвивають інструмент, можна організувати зв'язок не тільки з MQTT, але і з багатьма іншими системами. Так за допомогою Node-RED і плагінів можна підключитися до Telegram, Viber, Google Hangouts, WhatsApp, різних баз даних і зовнішніх веб-сайтів. За допомогою Node-RED можна написати своїх ботів, що зовсім не погано, заповнювати відомостями сховища даних, інформувати користувачів про події, що відбуваються, працювати як система алертінга. Але якщо і цього недостатньо, то можна написати свій плагін і використовувати зв'язок зі своєю системою по своєму власному протоколу.

Основні особливості Node-RED перелічені нижче:

- Підтримка редагування потоків в візуальному редакторі в браузері;
- Всі потоки створені в Node-RED зберігаються в форматі JSON, який можна легко імпортувати та експортувати для спільного використання з іншими;
- Можливість запуску локально;
- Він може працювати в хмарному середовищі, як Bluemix, AWS

Щоб почати використовувати Node-RED, спочатку його потрібно встановити. Це завдання залежить від платформи, і ми повинні вибрати правильну програму для встановлення відповідної платформи. Щоб Node-RED працював, нам потрібно встановити Node.js. Після цього потрібно встановити

Node-RED як глобальний модуль за допомогою команди(рис.2.4). Це додать команду node-red до системного шляху.

```
npm install -g --unsafe-perm node-red
```

Рис. 2.4 Команда для встановлення Node-RED

Після встановлення можна запускати за допомогою вводу команди node-red в командному рядку(рис.2.5).

```
C:>node-red
```

Рис. 2.5 Команда для встановлення Node-RED

Запуск Node-RED додає в корінь проекту папку з назвою .node-red. Цю папку можна вважати домашньою для конфігурації Node-RED для поточного користувача.Після запуску потрібно залишати термінал відкритим. Далі необхідно створити новий проект, для початку роботи з інструментом(рис.2.6). Для цього необхідно ввести ім’я проекту, опис, ключ шифрування та назва файлу де буде зберігатись проект, так як всі потоки в Node-RED зберігаються в форматі JSON. Цей файл можна імопртувати в інші проекти.

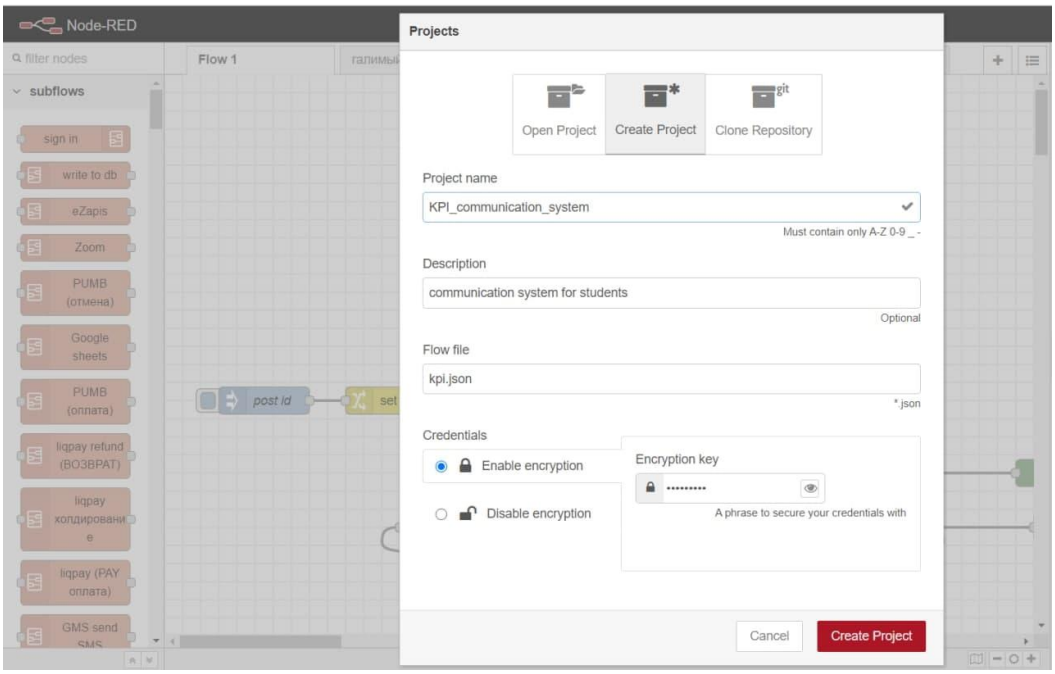


Рис. 2.6 Створення нового проекту Node-RED

Після створення проекту можна починати розробляти перший потік. Потоки для подій формуються візуальним способом на «полотні» потоку. Полотно виглядає як аркуш із зошита в клітинку. Таких листів в одній системі може бути багато. Досить тільки додати новий натиснувши плюсик в заголовку листа. Якщо натиснути два рази на ім'я полотна, то відкриються його властивості, де його можна перейменувати, або видалити. Щоб почати будувати потік, необхідно перетягувати вузли з палітри на полотно. Палітра Node-RED включає набір вузлів за замовчуванням, які є основними будівельними блоками для створення потоків:

- Debug – використовується для відображення повідомлень в структурованому вигляді на бічній панелі в редакторі.

- Inject - використовується для ручного запуску потоку, натискаючи кнопку вузла в редакторі. А також для автоматичного запуску потоків в зазначені інтервали часу.
- Function – запускає JavaScript код для об’єкту msg який приходить в повідомленні через нього.
- Change – використовується для модифікації властивостей повідомлення та встановлення властивостей контексту. Можно робити такі операції: встановленння, зміна, перейменування або видалення властивості.
- Switch – дозволяє перенаправляти повідомлення до різних гілок потоку.
- Template – використовується для створення тексту, використовуючи властивості повідомлення для заповнення шаблону.
- HTTP request - відправляє HTTP-запити і повертає відповіді. Може використовуватись для отримання веб сторінок чи запитів до API.

Якщо з якоїсь причини, не вистачає функцій вузлів, що вже присутні в наборі вузлів Node-RED, то їх кількість можна суттєво примножити. Нові набори вузлів додаються в систему через пункт системного меню Manage palette. Там можна побачити, які плагіни вузлів вже встановлені в системі, при необхідності їх можна відключити або видалити з системи.

Після того як в редакторі відбулися якісь зміни, з’являється можливість опублікувати проект. У Node-RED існує три способи публікації:

- Full - публікується і перезапускається одразу все полотно.
- Modified Flows - публікується відразу цілий потік, якщо хоча б один з його вузлів був модифікований.
- Modified Nodes - публікується тільки те, що було змінено.

Якщо обсяг проекту невеликий, то можна публікувати все разом. А ось в більш навантажених системах, де не слід переривати виконання вже працюючих потоків краще скористатися іншими опціями публікації.

2.1.4 MongoDB

MongoDB - це багатоплатформова, документоорієнтована база даних з відкритим кодом, призначена для зберігання великих масштабів даних, а також дозволяє дуже ефективно працювати з цими даними. Забезпечує високу продуктивність, високу доступність і простоту масштабування. Відноситься до не реляційних базам данних.

Дані, що зберігаються в MongoDB, мають формат BSON-документів. Тут BSON означає бінарне представлення документів JSON. Або іншими словами, у серверній системі сервер MongoDB перетворює дані JSON у двійкову форму, яка відома як BSON, і ця BSON зберігається та запитується більш ефективно.

У документах MongoDB можна зберігати вкладені дані. Це вкладання даних дозволяє створювати складні відносини між даними та зберігати їх в одному документі, що робить роботу та отримання даних надзвичайно ефективними порівняно з SQL

База данних містить в собі множину колекцій, яка в свою чергу містить в собі множину документів. Документи складаються з пар ключ-значення. Документ має динамічну схему, а це означає що документи в одній і тій самій колекції не повинні мати однакову множину пар ключ-значення, а також їх типи можуть бути різними. Отже, база даних, колекція та документи пов'язані між собою, як показано на рисунку 2.7.

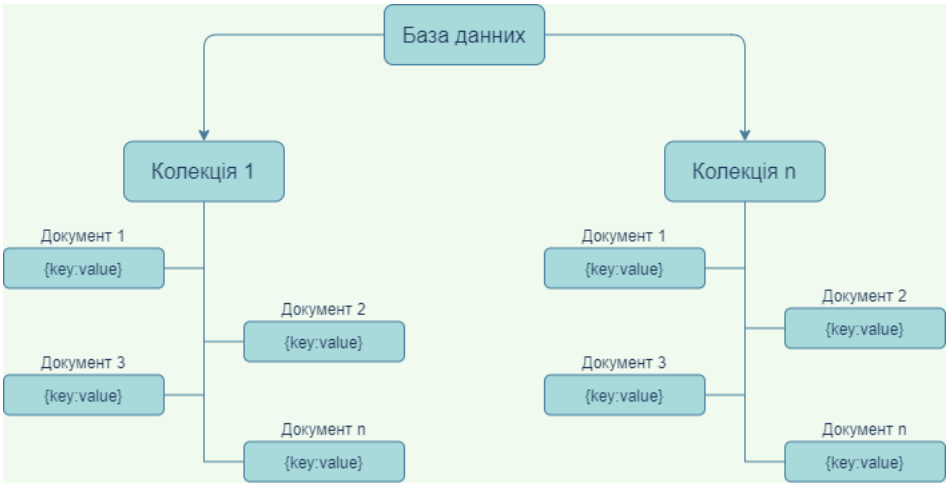


Рис. 2.7 Структура бази данних MongoDB

Розглянема особливості цієї бази данних:

- База даних без схем: чудова можливість, яку надає MongoDB. База даних без схем означає, що одна колекція може містити в собі різні типи документів. Іншими словами, в базі даних MongoDB в одній колекції може бути кілька документів, і ці документи можуть складатися з різної кількості полів, змісту та розміру. Не обов'язково, щоб один документ був подібним до іншого документа, як у реляційних базах даних. Завдяки цій цікавій функції MongoDB забезпечує велику гнучкість баз даних.
- Документоорієнтованість: у MongoDB всі дані, зберігаються в документах, а не в таблицях, як у RDBMS. У цих документах дані зберігаються в полях (пара ключ-значення) замість рядків і стовпців, що робить дані набагато гнучкішими порівняно із СУБД. І кожен документ містить свій унікальний ідентифікатор об’єкта.
- Індексування: у базі даних MongoDB кожне поле в документах індексується первинними та вторинними індексами, що полегшує та займає менше часу для отримання або пошуку даних із пулу даних. Якщо дані не індексуються, тоді пошук даних здійснюється в кожному документі із зазначеним запитом, що займає багато часу і не настільки ефективно.
- Масштабованість: MongoDB забезпечує горизонтальну масштабованість за допомогою шардінгу. Шардінг означає розподіл даних на декількох серверах.
- Реплікація: MongoDB забезпечує високу доступність і надмірність за допомогою реплікації, створює кілька копій даних і надсилає ці копії на інший сервер, так що, якщо один сервер виходить з ладу, дані отримуються з іншого сервера.
- Агрегація: дозволяє виконувати операції над згрупованими даними та отримувати одиничний результат або обчислений результат.

2.1.5 Текстовий редактор Visual Studio Code

Visual Studio Code - це легкий, але потужний редактор вихідного коду, який доступний для Windows, macOS та Linux. Він постачається із вбудованою підтримкою JavaScript, TypeScript та Node.js і має багату екосистему розширень для інших мов.

При першому запуску перед нами відкриється вікно, через яке можна отримати швидкий доступ до раніше запущених проектів або створити нові.

Інтерфейс Visual Studio Code (рис.2.8) розділений на п'ять основних областей, які можна легко налаштувати.

- Activity Bar: дозволяє перемикатися між меню: провідник, пошук, контроль версій, налагодження та розширення;
- Status: відображає інформацію про поточний відкритому проекті і файлі. Також містить кнопки для виконання дій з управління версіями, а також для включення функцій розширення.
- Panel: відображає різні панелі: вбудований термінал, панелі виводу для налагоджувальної інформації, помилок і попереджень.
- Editor: основна область для редагування ваших файлів. Можна розташувати кілька відкритих файлів поруч.

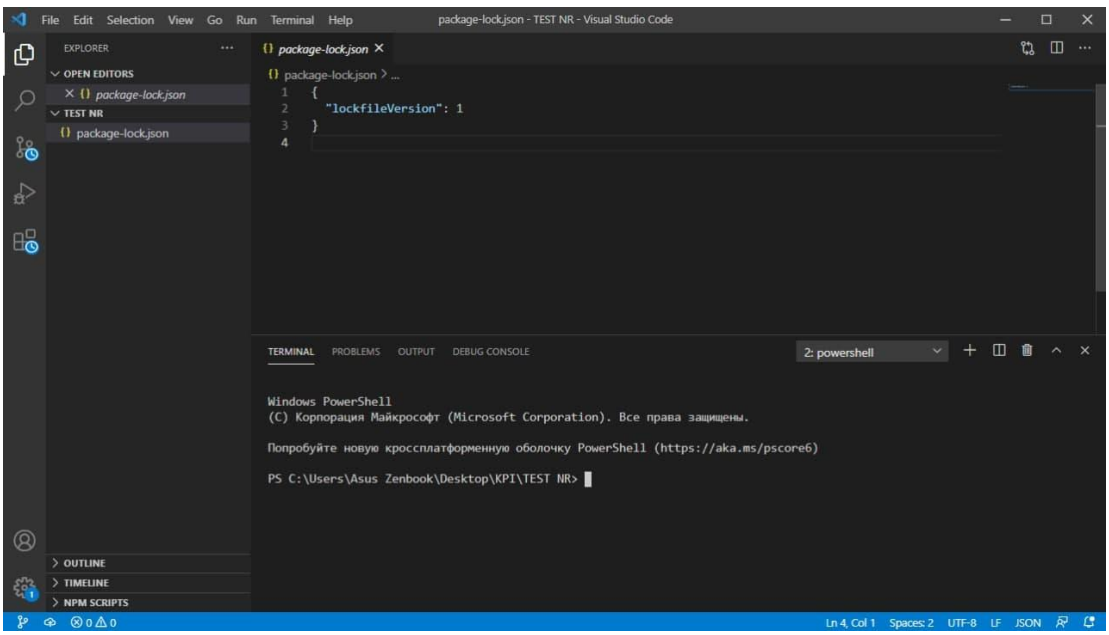


Рис. 2.8 Інтерфейс Visual Studio Code

Visual Studio Code незважаючи на свою простоту приховує в собі багато корисних можливотей:

- Сніпети - це серйозний спосіб підвищення продуктивності програміста. Вони допомагають автоматизувати ручну працю. Програмування - це, по суті, знання про те, де і як користуватися певними патернами. Сніпети прискорюють роботу через те, що полегшують доступ до заздалегідь заготовленим фрагментам коду. В результаті програміст може витрачати більше часу на роздуми про те, як певні конструкції поєднуються і як вони працюють разом, а не на те, щоб писати код цих конструкцій. Використання фрагментів, крім того, може допомогти програмісту швидше освоїти якийсь фреймворк або якусь бібліотеку. Справа в тому, що людині доводиться витрачати менше часу на те, щоб зрозуміти причину якоїсь помилки.
- IntelliSense - це система автозаповнення коду. Вона дуже інтелектуальна, що вигідно відрізняє її від інших подібних систем. Якщо навести курсор на якусь ділянку коду, IntelliSense покаже додаткові відомості про типи і дасть можливість дістатися до відомостей про джерело їх опису. Коли програміст, наприклад, вводить ім'я об'єкта і ставить після нього точку, з'являється список, що випадає з переліком методів цього об'єкта. У міру того, як вводяться все більше і більше букв імені методу, система фільтрує список. На певному етапі введення імені методу можна, клавішами-стрілками, вибрати потрібне ім'я з досить короткого списку, і натиснути Enter для його вставки в код. Якщо відразу не зрозуміло - який саме метод потрібен в даний момент - тут же можна поглянути і на документацію.

- Інтегрований термінал може допомогти економити час за рахунок того, що програмісту не доводиться постійно перемикатися між вікнами терміналу і редактора. Це ще означає і те, що в ході автоматичного запуску проекту дуже легко помічати помилки, не перериваючи при цьому свого звичайного робочого процесу. Для того щоб відкрити термінал, можна скористатися поєднанням клавіш CTRL + `.Термінал відкривається для кореневої директорії поточного робочого простору.

- Клавіатурні скорочення. Кожен раз, коли програмісту доводиться прибирати руки з клавіатури, він втрачає кілька секунд. Це подовжує терміни роботи над проектами. Багато розробників прагнуть до того, щоб робити все, що можна, за допомогою клавіатури. Клавіатурні скорочення дозволяють вирішувати більшість завдань тільки за допомогою клавіатури. Це означає, що менше часу витрачається на пошук потрібних команд в меню, що допомагає програмісту швидше досягати своїх цілей. Панель клавіатурних скорочень можна подивитися, скориставшись комбінацією клавіш CTRL + K + S.
- Робота з GIT. Прямо з редактора можна робити коміти в Git-репозиторії. Як виявилося, Visual Studio Code дуже добре підтримує Git. А саме, тут можна готувати до комітів змінені файли, робити коміти, повертати зміни, робити коментарі. Все це можна зробити не покидаючи редактора. Хоча все це можна зробити і в консолі, і скориставшись додатком GitHub, можливість працювати з Git, не залишаючи редактора, може допомогти в підвищенні продуктивності праці програміста.

ВИСНОВОК ДО РОЗДІЛУ 2

У цьому розділі були описані вибрані інструменти та технології для розробки системи комунікації у навчальних закладах. Були описані схема роботи та методи Telegram Bot API та Viber Bot API.

Наведена коротка характеристика інструмента візуального програмування Node-RED. Розглянуто створення простого проекту.

Описана структура бази данних MongoDB та її особливості. А також можливості текстового редактора Visual Studio Code.

РОЗДІЛ 3
РЕАЛІЗАЦІЯ ЧАТ-БОТУ TELEGRAM ТА VIBER

3.1 Отримання токену

3.1.1 Отримання токену Telegram

Для того, щоб почати роботу з чат-ботом необхідно створити бот та отримати його токен. Сторюється бот в Telegram за допомогою іншого бота під назвою @Botfather (рис.3.1).

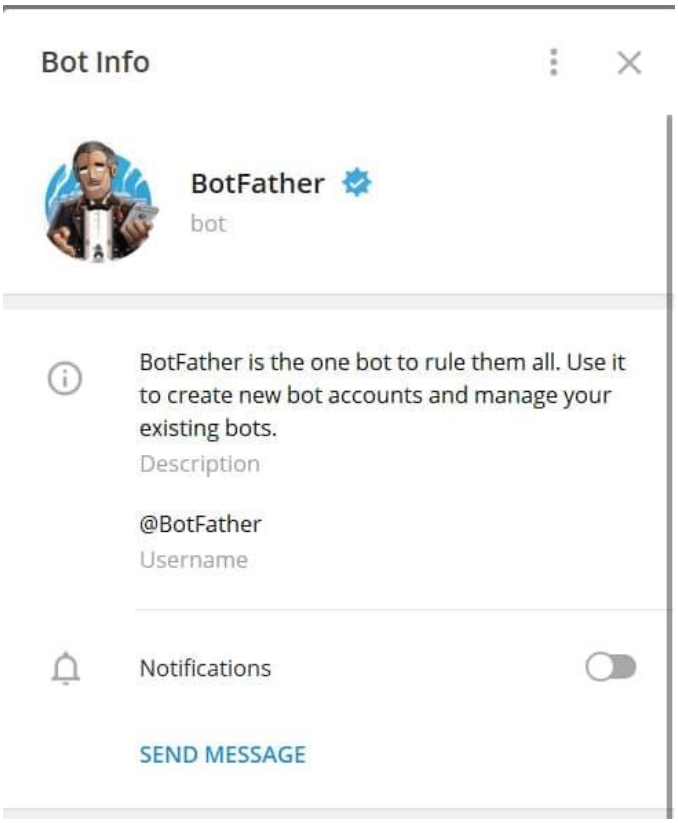


Рис. 3.1 Бот для створення чат-ботів

Для того щоб створити бот, необхідно пройти декілька кроків (рис.3.2).Спочатку необхідно ввести команду /start, після чого бот відразу запропонує вказати назву чат-бота. Якщо ім'я введено коректно, наступним кроком є ввід юзернему. Потрібно щоб ця назва обов'язково закінчувалась на bot, інакше виведеться повідомлення про помилку. Якщо ці два кроки пройшли успішно, ви отримаєте посилання на бота та його токен. Після цього, бот можна налаштувати, а саме: відредагувати ім'я, опис, фото бота або додати команди. Після отримання токену та налаштування боту можна приступати до його розробки.

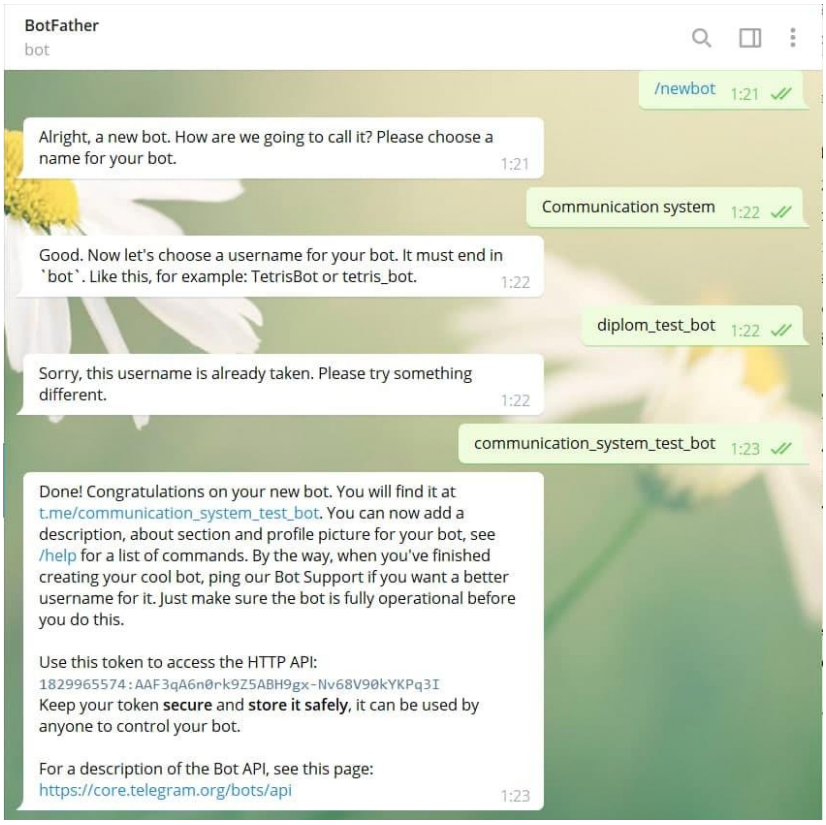


Рис. 3.2 Етапи створення чат-боту Telegram

3.1.2 Отримання токену Viber

Для створення боту в Viber потрібно зайти в панель адміністратора (рис.3.3). Для цього потрібно перейти за посиланням https://partners.viber.com/login та ввести свій номер телефону. Після успішної верифікації номеру ви попадете в панель адміністратора та зможете створити бот.

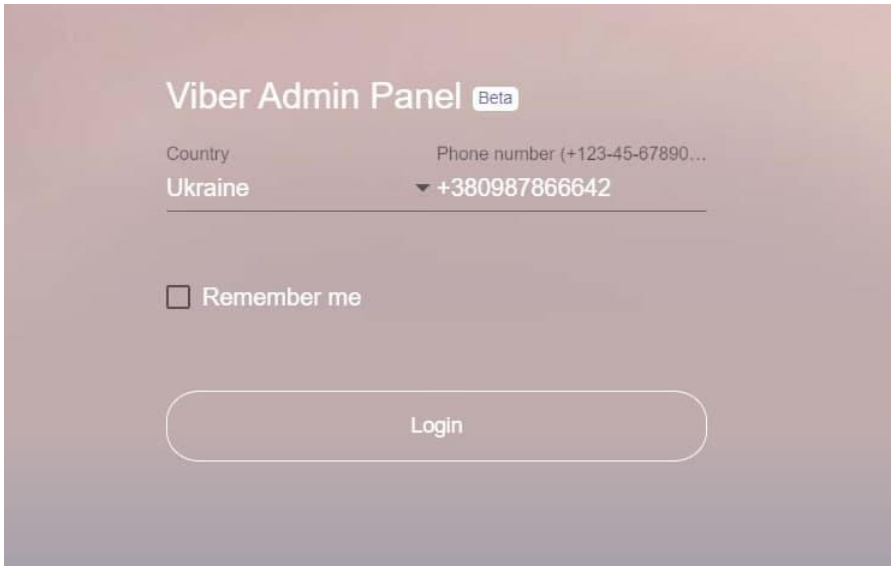


Рис. 3.3 Сторінка входу в панель адміністратора Viber

Для створення боту потрібно заповнити поля реєстрації боту як вказано на рисунку 3.4. А також завантажити фото та погодится з умовами використання. Вя Після цього ви отримаєте токени бота та можна приступати до його розробки.

Рис. 3.4 Реєстрація боту Viber

3.2 Розробка структури БД

В якості бази данних для чат-бота було використано MongoDB по ряду причин: зручність використання за рахунок зберігання данних в JSON форматі, простота інтеграції з іструментом Node-RED, відсутність схеми. Для встановлення MongoDB було завантажено один із пакетів з офіційного сайту. Після встановлення та налаштування бази данних було вирішено створити кластер за допомогою MongoDB Atlas. Після успішного налаштування кластеру, було отримано рядок для підключення за допомогою якого ми можемо робити запити до бази данних.

Схема взаємодії користувача з базою данних забражена на рисунку 3.5. Користувач надсилає повідомлення, воно звертається до сервера месенджера, далі сервер боту обробляє інформацію і звертається до бази данних, яка відправляє у відповідь потрібні дані для відповіді користувачу.

Рис. 3.5 Схема взаємодії БД з користувачем чат-боту

Окрім цього було розроблено структуру бази данних, яка складається з 3 колекцій для роботи бот платформи, та інших колекцій для роботи функціоналу бота.

- Колекція attachments в якій зберігаються шаблони клавіатур для ботів Telegram та Viber (рис.3.6). Складається з назви функціоналу, назви поточного кроку, та шаблонів клавіатур для месенджерів.

Рис. 3.6 Приклад документу з колекції attachments

- Колекція localization необхідна для зберігання тексу повідомлень та клавіатури. Структура документу складається з назви функціоналу та об'єкту з локалізацією (рис.3.7). Локалізація окрема для кожної функції.

Рис. 3.7 Приклад документу з колекції localization

- Колекція userStates необхідна для функціонування бот платформи, адже в ній ми зберігаємо поточний стан користувача, що не дає йому заблукати в процесах. В документі ми зберігаємо channel та chatId користувача, поточний та попередній крок користувача, а також поточне та попереднє повідомлення, яке було надіслано користувачу. Вигляд документу можна побачити на рисунку 3.8.

Рис. 3.8 Приклад документу з колекції userState

- Колекція users необхідна для зберігання данних про користувача: назва месенджеру та ідентифікатор користувача в ньому, роль, ім'я, вік, телефон, факультет, група та ознака реєстрації, яка може приймати значення true або false. Приклад документу із даної колекції зображений на рисунку 3.9.

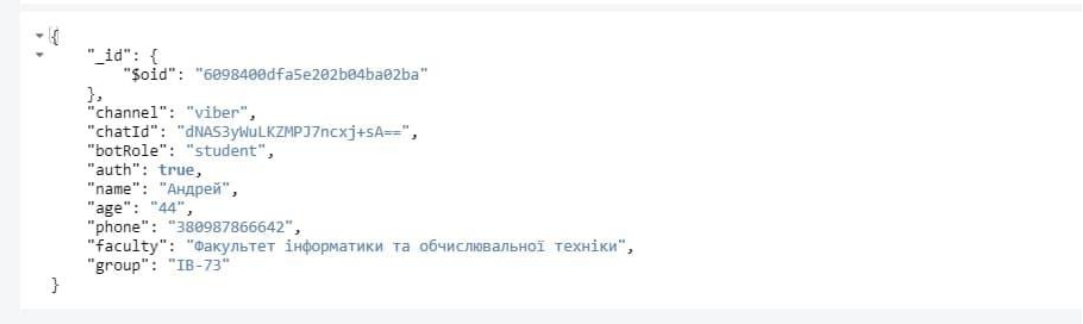


Рис. 3.9 Приклад документу з колекції users

Всі інші колекції використовуються для роботи кожної окремої функції, яка є в боті.

Колекції mailingHistory, pollHistory та userPoll необхідні для коректної роботи інформаційних розсилок та опитувань, а також запису відповідей на питання в опитуванні.

Колекція banks зберігає всю необхідну інформацію по відділенням банків, що ми відображаємо в боті: назва банку, довжина та широта для відображення на карті, а також назва вулиці де знаходиться відділення.

Для коректного відображення місць для харчування, ми зберігаємо дані по столовим в колекції cafe. В ній ми зберігаємо назву столової, її адресу, розклад, ціни, опис, фото, а також посилання на меню із цінами на кожну окрему страву.

Колекція deanRequest необхідна для зберігання запитів які надсилаються в деканат студентами. Для кожного факультета окремий документ. Зберігаємо channel та chatId деканату, назву факультета,питання яке було задано, інформацію про студента, час коли було задане питання, а також статус заявки, для розуміння на якому етапі знаходиться запит.

Колекція dormitories зберігає всю необхідну інформацію по гуртожитках для подальшого відображення в боті. Документ зберігає в собі: назву гуртожитку, опис, ім’я коменданта та завідуючого, телефон, ціни та фото гуртожитку.

Колекція faculties включає в себе інформацію по факультетам, а саме: повна та коротка назва, адреса, ім’я декана, телефон та посилання на веб-сайт.

Колекція popularQuestion зберігає список популярних питань та відповідей у зручному для відображення в боті вигляді.

Дані колекції зберігаються в базі данних під назвою main, яка створена для даного проекту. Зберігання данних в цій колекції пришвидшує час отримання інформації, ніж якби ми її отримували за допомогою парсингу.

3.3 Запуск Node-RED

Для запуску інструменту Node-RED необхідно його завантажити його за допомогою команди «npm install -g --unsafe-perm node-red». Дана команда встановить Node-RED як глобальний модуль разом з усіма залежностями. Після цього можна запустити інструмент за допомогою команди node-red і з'явиться пусте полотно Node-RED.

3.4 Розробка бот платформи

Бот платформа повинна складатися з декількох компонентів (процесів):

1. Receiver – відповідає за підключення вебхуків та їх обробку
2. Router – обробляє повідомлення яке прийшло з процесу Receiver та в залежності від параметрів skill та message.text направляє в потрібний процес
3. Collect message – відповідає за збірку повідомлення перед відправкою
4. Send message – відправляє повідомлення в бот за допомогою API месенджера.
5. Main menu – процес для відправки головного меню користувачу.

Дані компоненти разом створюють універсальну бот платформу, яка дає можливість створювати діалогові чат-боти як мінімум для двох месенджерів.

3.4.1 Receiver

Даний процес виконує декілька функцій – підключення вебхуків (рис.3.10) та їх обробка і преведення до нормального вигляду (рис.3.11).

Підключення вебхуків для Telegram та Viber відбувається автоматично при запуску Node-RED. За замовчуванням створюється тунель ngrok +/botToken, який і є посиланням для підключення вебхуків. Якщо в файлі env вказаний домен хмарного сховища на якому хоститься Node-RED, то посилання береться як serverUrl +/botToken.

Обробка вебхуків для Viber відбувається за таким сценарієм :

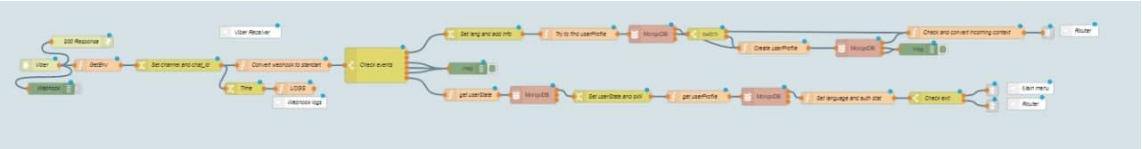


Рис. 3.11 Процес обробки вебхуків для Viber

1. Отримуєм вебхук від месенджера та відправляєм statusCode = 200 у відповідь.
2. Дістаємо всі змінні із файлу env.
3. Записуємо в логи вхідне повідомлення (що прийшло і від кого).
4. Приводимо всі вебхуки для зручності до єдиного вигляду (рис.3.12).

```
if(msg.payload.message){
switch (msg.payload.message.type) {
  case "text":
    msg.message = {"type":"text","text":msg.payload.message.text};
    break;
  case "picture":
    msg.message = {"type":"image","text":msg.payload.message.media};
    break;
  case "video":
    msg.message = {"type":"video","text":msg.payload.message.media};
    break;
  case "url":
    msg.message = {"type":"url","text":msg.payload.message.media};
    break;
  case "file":
    msg.message = {"type":"file","text":msg.payload.message.media};
    break;
  case "contact":
    msg.message = {"type":"contact",
    "contact":{
        "name":msg.payload.message.contact.name,
        "phone":msg.payload.message.contact.phone_number,
        "text":msg.payload.message.contact.text
        }
    };
    break;
  case "location":
    msg.message = msg.payload.message;
    break;
}
} else {
    msg.message = {"type":"conversation_started","text":msg.payload.event};
}
return msg;
```

Рис. 3.12 Приведення вебхуку до єдиного стандарту

5. Перевіряємо тип вебхуку
6. Якщо тип - start створюємо профіль користувача і додаємо основні параметри: channel, chatId, language (рис.3.13). Якщо інший тип – отримуємо із колекції userStates поточний крок користувача
7. Якщо команда /exit переходимо до головного меню, якщо ні - в процес Router.

3.4.2 Router

Даний компонен бот платформи призначений для того щоб направляти користувача в потрібний процес, забражений на рисунку 3.12. Спочатку ми парсимо вебхук який прийшов (рис.3.13), і збираємо необхідні дані, а потім в залежності від параметрів skill та userState направляємо в потрібний процес

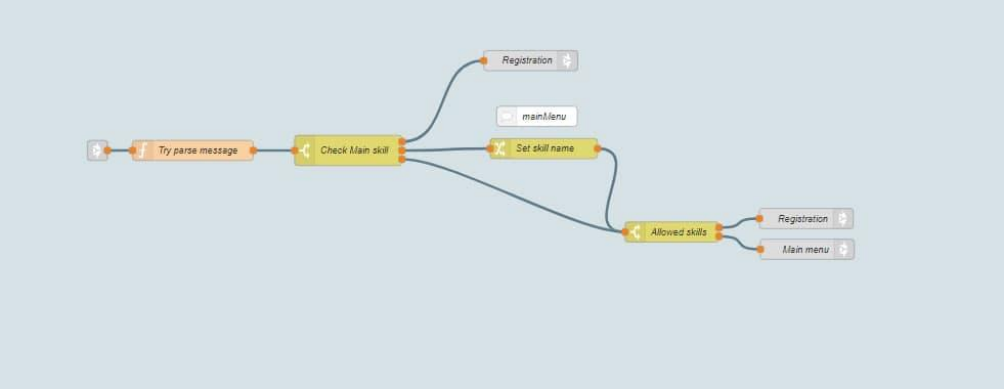


Рис. 3.12 Процес Router

Наявність параметрів skill та userState – ознака, яка дозволяє направити користувача в потрібний процес. Перевіряємо параметр skill на значення, яке дозволяє направити користувача в потрібний процес.

```
1  //skill_registration_userState_start_userMessage_example
2
3  if (msg.message.text){
4      if (msg.message.text.includes('userState')){
5        var arr = msg.message.text.split('_')
6        msg.skill = arr[1]
7        msg.userState = arr[3]
8          if (arr[5]){
9             msg.message.text = arr[5]
10         } else {
11            msg.message = msg.prevUserMessage
12         }
13     }
14
15 }
16 return msg;
```

Рис. 3.13 Приклад парсингу вхідного повідомлення

3.4.3 Collect message

Даний процес потрібен для того, щоб зробити готовий шаблон з необхідними параметрами і готовим до відправки запиту до API месенджера. Процес забражений на рисунку 3.14.

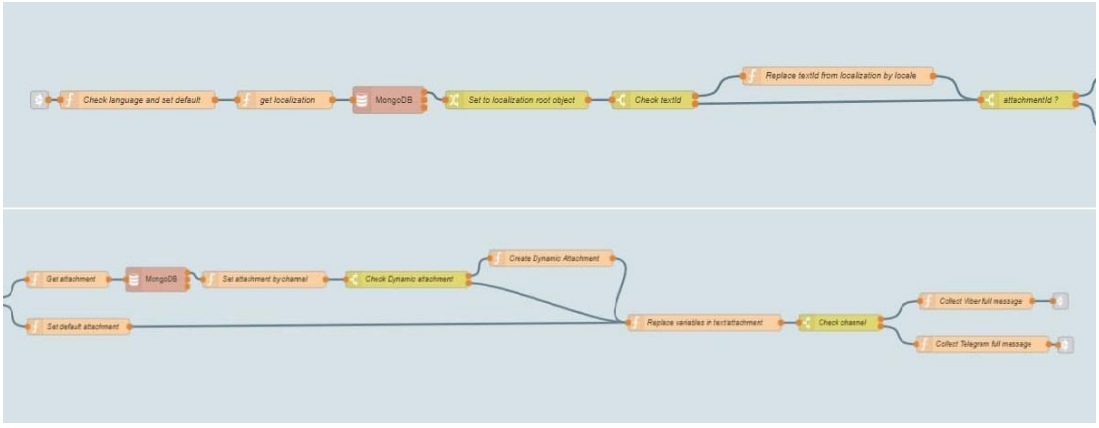


Рис. 3.14 Процес Collect message

Алгоритм дій в процесі Collect message:

1. Отримуємо із бази об’єкт з локалізацією, по параметру skill.
2. Знаходимо потрібний ключ в локалізації по значенню параметра textId.
3. Дістаємо із бази шаблон клавіатури, по значенню параметрів skill та attachmentId.
4. Перевіряємо шаблон на динамічність (рис.3.15).

```
// PAGINATION
var totalPages = 0;
var maxItemsCountToShow = 0;
var currentPage = parseInt(msg.currentPage) || 0;
var currentOffset = 0;

if (!msg.attachment.buttons) {
    msg.attachment.buttons = [];
}

// Get max. items count to render
if (!msg.attachment.maxItemsCountToShow && !msg.maxItemsCountToShow) {
    switch (channel) {
        case "viber":
            if (attachment.type === "carousel" || attachment.type === "carousel_with_keyboard") {
                if (msg.itemsCount){
                    maxItemsCountToShow = msg.itemsCount;
                } else {
                    maxItemsCountToShow = 6;
                }
            }
            if (attachment.type === "keyboard") {
                maxItemsCountToShow = 0;
            }
            break;
        case "telegram":
            if (attachment.type === "inline_keyboard") {
                maxItemsCountToShow = 5;
            } else if (attachment.type === "keyboard") {
                maxItemsCountToShow = 6;
            } else {
                maxItemsCountToShow = 10;
            }
            break;
```

```
// Create dynamic carousel
if (attachment.items && msg.items) {
    if (attachment.type == "image_list_picker") {
        var images = [];
        var imagesPattern = attachment.images;
        createDynamicAttachment(images, itemOptions, imagesPattern, re);
        msg.attachment.images = images;
    }
    createDynamicAttachment(items, itemOptions, itemsPattern, re);
    msg.attachment.items = items;
}

if (attachment.type === "keyboard" || attachment.type === "quick_replies" || attachment.type === "inline_keyboard") {
    msg.buttons = itemOptions;
}
// Create dynamic keyboard
if (attachment.buttons && msg.buttons) {
    createDynamicAttachment(buttons, msg.buttons, buttonsPattern, re);
    msg.attachment.buttons = buttons;
}

msg.totalPages = totalPages;

// Create pagination buttons
if (totalPages > 1) {
    var paginationButtons = createPaginationButtons(currentPage, totalPages, channel);
    paginationButtons.forEach(function(button) {
        if (attachment.type === "keyboard" || attachment.type === "quick_replies" || attachment.type === "inline_keyboard"
            msg.attachment.buttons.push(button);
        } else {
            msg.attachment.buttons.unshift(button);
        }
    });
}
```

Рис. 3.15 Створення динамічного повідомлення

5. Підстановка тексту і клавіатури в шаблон повідомлення (рис. 3.16).

```
function matchToReplace(stringToReplace, dataSource, regExp) {
    if (typeof stringToReplace === 'string') {
        var d = stringToReplace.match(regExp);
        if (d !== null && d !== undefined && d[0] !== "") {
            for (var i = 0; i < d.length; i++) {
                // Localization
                if (mode == "localize") {
                    v = d[i].replace(/{{t'/, "").replace(/}}/, "");
                    if (v !== undefined) {
                        if (dataSource[v] !== undefined) {
                            stringToReplace = stringToReplace.replace(d[i], dataSource[v][msg.language]);
                        }
                    }

                // Variables
                } else {
                    v = d[i].replace(/{{/, "").replace(/}}/, "");
                    if (v !== undefined) {
                        var arrayOfKeys = v.split(".");
                        if (arrayOfKeys.length > 1) {
                            var o = dataSource;
                            for (var k in arrayOfKeys) {
                                var s = arrayOfKeys[k];
                                o = o[s];
                                if (typeof o === "string") {
                                    stringToReplace = stringToReplace.replace(d[i], o);
                                }

                            }
                        } else {
                            if (dataSource[v] !== undefined) {
                                stringToReplace = stringToReplace.replace(d[i], dataSource[v]);
                            }
                        }
                    }
                }

            }
        }
    }
    return stringToReplace;
```

Рис. 3.15 Функція для підстановки динамічних значень

6. В залежності від типу повідомлення додати потрібні параметри для успішного запиту до API месенджера.

3.4.4 Send message

Всього існує 2 процеси Send message для кожного месенджера (рис.3.16). Вони аналогічні між собою по логіці, але мають різну структуру запитів. Даний процес перевіряє тип повідомлення та додає необхідні параметри до запиту і власне виконує запить до API месенджера.

Якщо запит виконався успішно то за допомогою процесу Change userState змінюємо поточний userState користувача в базі. Якщо запит виконався з помилкою, то записуємо в логи помилку.

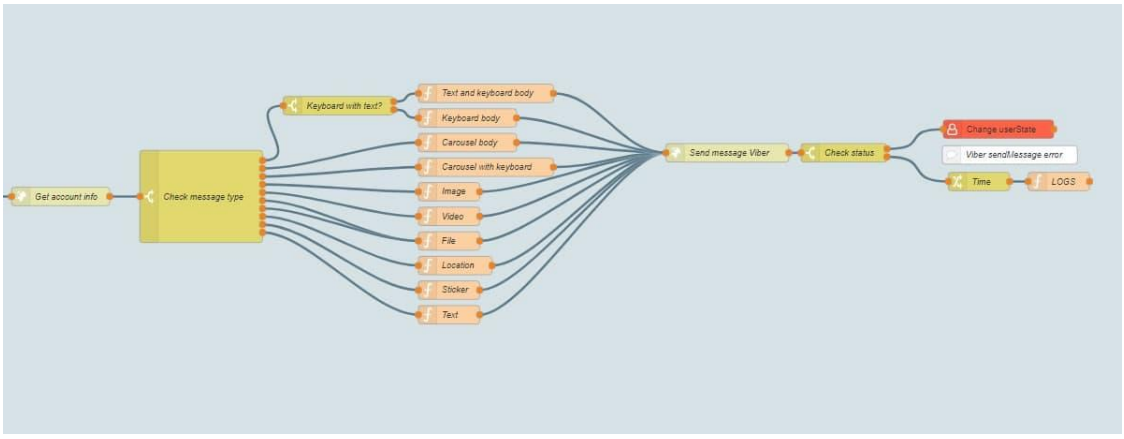


Рис. 3.16 Процес відправки повідомлення

Для кожного типу повідомлень формується окремий запит, так як необхідні різні парметри. Тип повідомлення перевіряється за допомогою значення параметру type в шаблоні клавіатури. Можливі типи повідомлень для месенджера Viber: text, keyboard, carousel, carousel_with_keyboard, image, video, file, location, sticker. Приклад підготовки запиту для каруселі забражений на рисунку 3.17.

```
msg.method = "POST"
msg.url = msg.sendMessageUrl
msg.payload = {
  "auth_token": msg.token,
  "receiver": msg.chatId,
  "type": "rich_media",
  "sender":{"name":msg.payload.name,"avatar":msg.payload.icon},
  "rich_media": msg.rich_media,
  "min_api_version": 8

}

msg.requestBody = msg.payload
return msg;
```

Рис. 3.17 Приклад запиту для відправки повідомлення

3.4.5 Інші процеси бот платформи

Окрім основних процесів для функціонування бот платформи необхідні додаткові процеси та інтеграції. Серед них процеси Change userState та підпроцеси для обробки вебхуків в телеграм. А також інтеграція з MongoDB та Google Sheets.

Розглянемо дані процеси:

- Change userState – процес для зміни поточного userState (кроку) та skill (функціонал), зображений на рисунку 3.18.

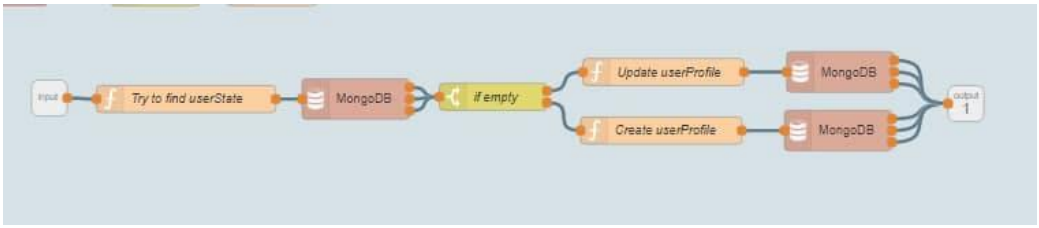


Рис. 3.18 Процес Change userState

Це потрібно для того щоб користувач міг рухатися по процесу, так як проект складається з великої кількості процесів, і системі потрібно якось розуміти де знаходиться кожен із користувачів. Дані беруться та зписуються в колекцію під назвою userStates. Документи в цій колекції зберігають skill, userState, prevUserState, channel та chatId.

- keyboardConverter - потрібен для того щоб змінювати повідомлення яке приходить у вебхуку на значення ключа цього повідомлення в колекції localization. Спочатку дістаємо із колекції localization весь об’єкт поточного функціоналу. А потім знаходимо значення яке співпадає із вебухуком, та змінюємо його на ключ цього значення.
- removeReplyMarkup – процес потрібен для того щоб прибрати клавіатуру, якщо вона не потрібна. Для цього нам потрібно на вхід подати поточний userState та повідомлення яке буде надсилатись.
- hideInlineKeyboard – процес для того щоб сховати клавіатуру якщо вона не потрібна. Її можна легко показати натиснувши на знак біля поля вводу в месенджері.

- MongoDB – процес в якому відбувається відправка запитів до бази данних та обробка відповідей (рис. 3.19). Для кожної колекції створюється окремий endpoint на який відправляється запит. Для успішного виконання процесу в нього потрібно подати декілька параметрів: collection та body. В collection задається колекція до якої потрібно звернутися, а в об'єкт body подаються параметри: operation – метод який потрібно виконати(insert, findOne, findAndUpdate),request – записуємо сам запит до бази данних.

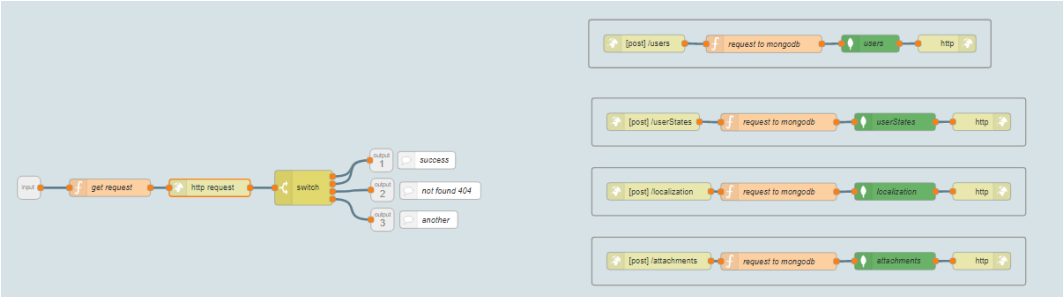


Рис. 3.19 Процес для запитів до бази данних

Зв'язок з базою відбувається за допомогою плагіну Node-RED під назвою mongodb3. Для коректної роботи його потрібно підключити до бази та вказати колекцію до якої потрібно звертатись.

- Google sheets – інтеграція з google sheets потрібна для запису та читання з таблиць всієї необхідної інформації. Підтримується 4 дії: запис по рядкам, запис в комірку та читання таблиці. Для успішної інтеграції з таблицями потрібно спочатку створити таблицю, написати програму для запису чи читання,опублікувати як веб-додаток та отримати посилання для запиту. Приклад програми для запису в рядок зображений на рисунку 3.20.

```
function doGet(request) {

   var row = [];
   row.push(request.parameter.date);
   row.push(request.parameter.name);
   row.push(request.parameter.phone);
   row.push(request.parameter.role);
   row.push(request.parameter.question);


   var ss = SpreadsheetApp.openById("1lw_Wp-IIOSs75YVYpAVPgut2Kpv9qACS-uyV3121Cfg")
   var sheet = ss.getSheetByName("questions");
   sheet.appendRow(row);

   var result = {"result":"ok"};

   return ContentService.createTextOutput(JSON.stringify(result))
     .setMimeType(ContentService.MimeType.JSON);
}
```

Рис. 3.20 Приклад програми для запису в таблицю

Після того як отримали посилання для запиту, потрібно в тіло запиту додати всі необхідні параметри для успішного запису в таблицю.Вигляд процесу для запису в таблицю зображений на рисунку 3.22.

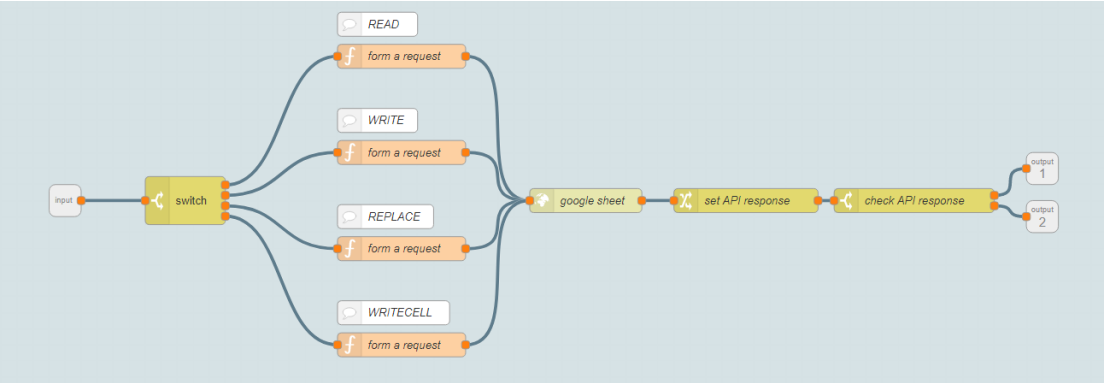


Рис. 3.21 Процес інтеграції з Google sheets

3.5 Розробка функціоналу

3.5.1 Реєстрація в чат-боті

Після того як бот платформа разом з усіма компонентам розроблена та зв'язана, можна починати реалізацію функціоналу.

Для того щоб створити процес, потрібно:

- Створити новий потік за допомогою панелі Node-RED
- Перейти в процес Router та додати нову функцію в ноді switch(рис.3.22)

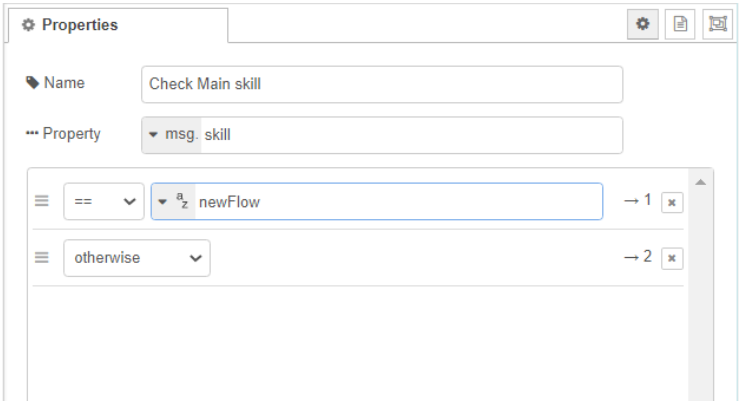


Рис. 3.22 Додавання нової функції в Router

- За допомогою нод link in та link out з'єднати процеси
- Створити в базі данних новий документ в колекції localization.

Після цього можна починати розробляти перший процес в Node-RED. Даний алгоритм дій актуальний для кожного процесу який створюється

Для того щоб відправити повідомлення потрібно в ноді function задати декілька параметрів: attachmentId, textId, skill, userState, prevUserState. Після цього потрібно з'єднати ноду function з нодою link in, яка веде в процес Collect message. Приклад відправки повідомлення зображений на рисунку 3.23.

```
msg.attachmentId = "startMessage";
msg.textId = "startMessage";
msg.skill = "registration";
msg.userState = "startMessage";
msg.prevUserState = "start";
return msg;
```

Рис. 3.23 Приклад коду для відправки повідомлення

Реалізація даного проекта починається з реєстрації. Вона представляє собою список питань для ідентифікації та збору інформації про користувача. Пройти реєстрацію можна пройти лише один раз, а якщо користувач захоче пройти її ще раз, чат-бот буде відображати головне меню, в залежності від ролі людини в боті. Це зроблено для того, щоб користувачу який видалив бот та зайшов до нього знову, не відображати заново реєстрацію, а відразу переходити до головного меню.

Під час реєстрації чат-бот запитує ім'я, вік та роль користувача та записує їх в таблицю в потрібний лист, в залежності від ролі. Після кожного питання відбувається запис в базу данних в колекцію users.

Якщо роль – абітурієнт, то реєстрація закінчена, а дані користувача записуємо в таблицю

Якщо роль – студент , запитуємо факультет на якому він навчається. Список факультетів ми отримуємо з бази данних, тому некоректний ввід неможливий. Наступним кроком іде ввід групи, в якій навчається студент. Групу користувач вводить самостійно, але вона перевіряється на валідність за допомогою API для отримання розкладу. Якщо розклад знайдений, то така група існує (рис. 3.24).

	A	B	C	D	E	F
1	Дата	Ім'я	Телефон	Факультет	Група	
2	21.05.2021	Андрій	380987866642	Факультет інформатики та обчислювальної техніки	ІВ-72	
3	22.05.2021	Олена	380978345673	Інженерно-хімічний факультет	ТХ-83	
4	22.05.2021	Ваня	380976568712	Факультет електроніки	ЕЛ-72	
5	25.05.2021	Владислав	380957655834	Факультет лінгвістики	ІК-93	
6						

Рис. 3.24 Список зареєстрованих в боту студентів

Якщо роль користувача – деканат, потрібно звірити введений користувачем номер телефону та телефон який вписаний в таблиці (рис.3.25). Якщо знайшли номер в таблиці, виводимо інформацію користувачу в бот та просимо підтвердити що це він. Якщо підтверджено,вітаємо з реєстрацією та виводимо користувачу головне меню адміністратора. Якщо номер не знайдено або не підтверджено, то записуємо користувача в таблицю та відправляємо головне меню гостя. Так як номером можна поділитися лише через кнопку, то ймовірність того, що адміністратором стане хтось небажаний – мінімальна.

	A	B	C	D
1	ПІБ	Телефон	Факультет	
2	Іванов Іван Іванович	380987866643	ФІОТ	
3	Тимко Андрій Олексійович	380987866644	ІПСА	
4	Петренко Петро Петрович	380987866641	РТФ	
5	Олексенко Олексій Олексійович	380987866646	ФЛ	
6				
7				
8				

Рис. 3.25 Таблиця з даними адміністраторів (деканат)

3.5.2 Функціонал абітурієнта

Головне меню абітурієнта складається з дев’яти кнопок.

- Приймальна комісія – дана кнопка меню потрібна для того щоб показати всю необхідну інформацію для вступу абітурієнту. Процес складається з двох кроків і зображений на рисунку 3.26.

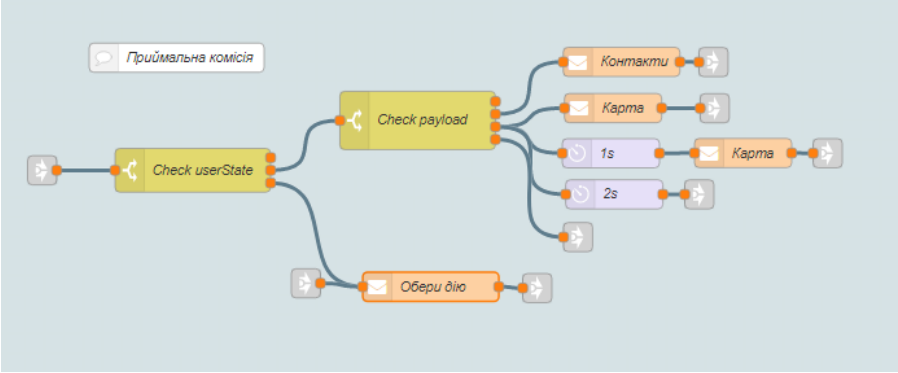


Рис. 3.26 Функціонал «Приймальна комісія»

Для відображення кнопок використовуємо InlineKeyboardMarkup. Шаблон кнопок для Telegram зображений на рисунку 3.27.

Рис. 3.27 Шаблон клавіатури для Telegram

Вона включає в себе кнопки з посиланням на сайт університету та звичайні кнопки. Можна переглянути графік роботи, контактний номер телефону приймальної комісії та їх розміщення по території університету для різних факультетів. Окрім цього можна підрахувати свій конкурсний бал

- Кнопка «Необхідні документи» це процес для відображення документів необхідних для вступу. Виділений в окрему кнопку, так як це найбільш потрібна інформація для абітурієнта, яка потрібна бути доступна
- Кнопка «Вартість навчання» необхідна для показу актуальних цін на навчання на різних спеціальностях. Інформація показана у вигляді документу, тому його можна легко завантажини собі на носій. Посилання на файл зберігається в базі данних тому для оновлення інформаціїї необхідно лише змінити посилання на файл.
- Кнопка «Гуртожитки» відображає всі гуртожитки які є в університеті. Потрібно користуватися кнопками пагінації для перегляду всього списку. Та при нажатті на конкретний гуртожиток відображає його фото, опис, телефон для зв'язку, ціни та посилання на телеграм канал цього гуртожитку. Список гуртожитків зберігається в колекції dormitories (рис. 3.28).

```
_id: ObjectId("60a8d50848358c4978b056c3")
name: "Гуртожиток №1"
description: "Гуртожиток коридорного типу, санвузли та кухні розташовані на кожному ..."
telegramChannel: "https://t.me/kpi_hostel1"
manager: "Філіпішина Любов Петрівна "
commandant: "Лисак Олександра Констянтинівна"
phone: "+380442049496"
photo: "https://i.ibb.co/6FxcxXz/dorm1.jpg"
price: "https://telegra.ph/cost-of-living-06-16#%D0%93%D1%83%D1%80%D1%82%D0%BE..."

_id: ObjectId("60a8d5ab48358c4978b056c5")
name: "Гуртожиток №3"
description: "В гуртожитку розміщена дирекція студмістечка КПІ ім. Ігоря Сікорського..."
telegramChannel: "https://t.me/kpi_hostel3"
manager: "Лозинська Лідія Іванівна"
commandant: "Нагорная Світлана Іванівна"
phone: "+380442049266 (прохідна)
+380442048043 (зав. гуртожитком)"
photo: "https://i.ibb.co/Sm8P5gN/dorm3.jpg"
price: "https://telegra.ph/cost-of-living-06-16#%D0%93%D1%83%D1%80%D1%82%D0%BE..."
```


Рис. 3.28 Колекція dormitories

Після того як користувач обрав гуртожиток, ми шукаємо в базі по назві гуртожитка документ, і виводимо інформацію з цього документу.

- Кнопка «Факультети» потрібна для ознайомлення абітурієнта зі всіма існуючими факультетами в університеті. Процес в Node-RED зображений на рисунку 3.29.

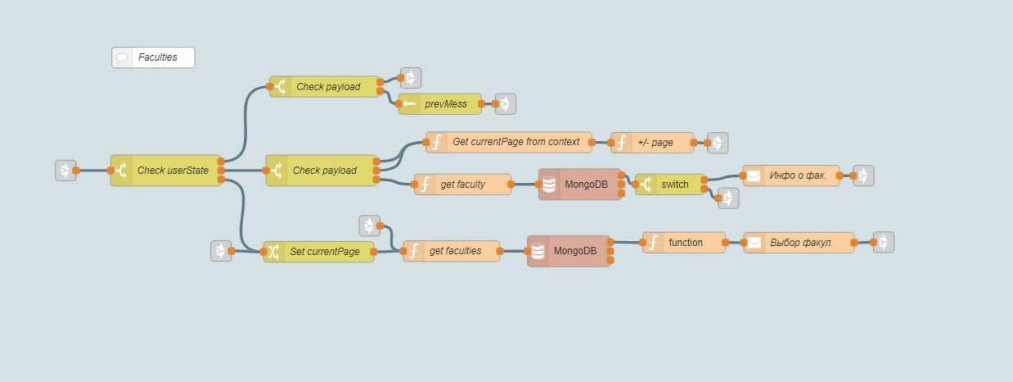


Рис. 3.29 Процес функціоналу «Факультети» в Node-RED

Спочатку шукаємо в базі список всіх факультетів та відображаємо їх в клавіатурі. Після відповіді користувача перевіряємо ввід на коректність, і якщо все правильно, відображаємо інформацію по факультету. Показуємо користувачу адресу, телефони для довідок та посилання на сайт.

- Кнопка «Як дібратись» показує варіанти, як можна дібратися до університету. А також є можливість подивитися на карті зупинки громадського транспорту в районі університету. Використовується лише відправка тексу то фото.

- Кнопка «Популярні питання» відображає найбільш популярні питання, з якими стикається абітурієнт при вступі в університет. Процес зображений на рисунку 3.30. Для перегляду всіх питань необхідно користуватися кнпоками пагінації. При нажатті на питання, користувачу відображається відповідь у текстовому вигляді. Питання та відповіді зберігаютья в базі данних в колекції popularQuestions

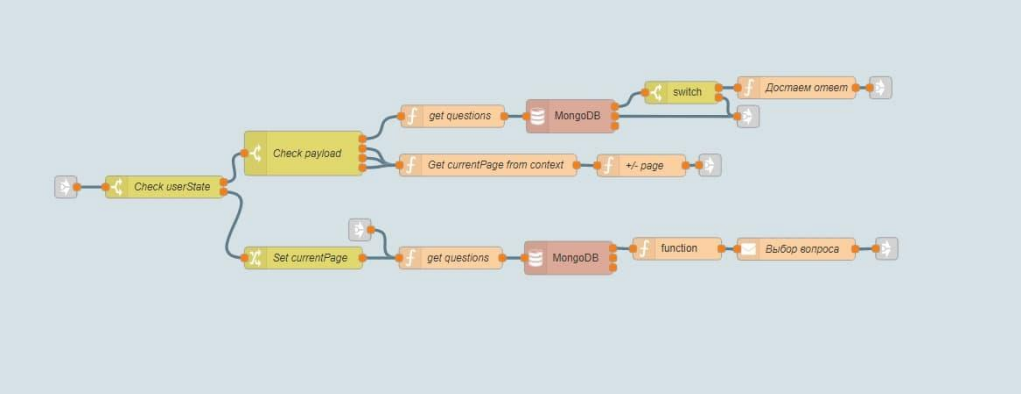


Рис. 3.30 Процес функціоналу «Популярні питання» в Node-RED

В базі данних зберігається всього 3 параметра: question, answer, id (рис 3.31). В кнопку з питаннями вшивається ідентифікатор питання, по якому потім ми шукаємо в базі відповідь на обране питання.


```
_id: ObjectId("604f2c67be53c2c77ef9302a")
question: "Як подати документи електронно?"
answer: "Подання документів до ЗВО України для вступу на денну та заочну форми ..."
id: "question1"

_id: ObjectId("604f2c75be53c2c77ef9302c")
question: "Які документи потрібні для вступу?"
answer: "паспорт або свідоцтво про народження для тих, у кого немає паспорту, к..."
id: "question2"

_id: ObjectId("60993f5a967b4f4f08cf70ee")
question: "Чи потрібно подавати нотаріально завірені копії?"
answer: "Ні, із собою ти береш оригінали та копії документів, їх на місці завір..."
id: "question3"
```


Рис. 3.31 Документи в колекції popularQuestions

- Кнопка «Залишити відгук» надає можливість користувачу залишити скаргу чи побажання до університету. Після чого ця скарга буде записана в таблицю та взята в роботу співробітниками університету. Запис в таблицю відбувається за допомогою інтеграції з Google sheets, таким чином після вводу запитання програма відправляє запит до Google sheets та записує в таблицю всі необхідні поля.
- Кнопка «Налаштування» потрібна для зміни своєї ролі у випадку якщо абітурієнт вже став студентом або якщо при реєстрації по помилці неправильно ввів свою роль. Процес нічим не відрізняється від процесу реєстрації. Процес зображений на рисунку 3.32

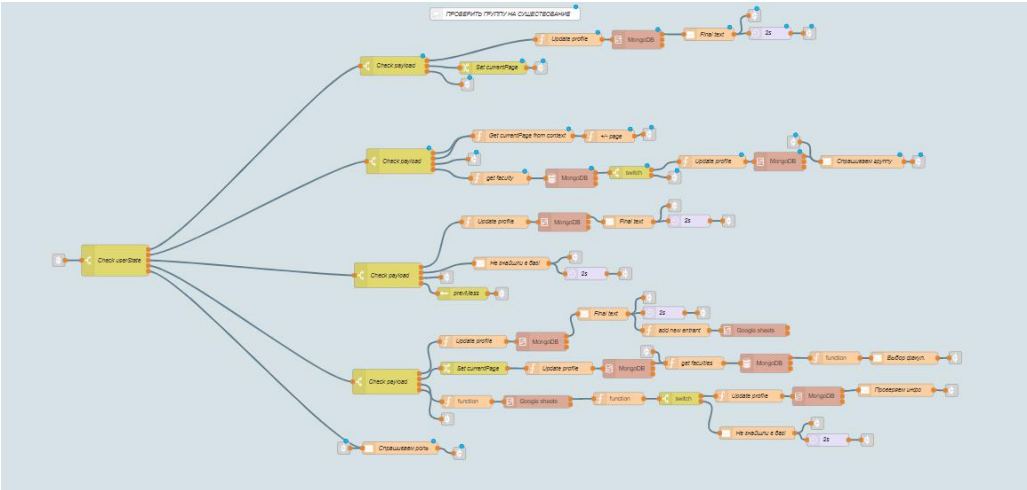


Рис. 3.32 Процес налаштування профілю

3.5.3 Функціонал студента

Функціонал студента більш насичений цікавими та корисними можливостями. Головне меню налічує десять кнопок. Деякі кнопки співпадають з тими, що є в функціоналі абітурієнта, але вони перероблені під інтереси студента. За допомогою цього функціоналу можна переглянути розклад, відправити запит в деканат, знайти заклади харчування, бази відпочинку та відділення банку, а також залишити відгук та переглянути популярні питання.

- Кнопка «Розклад» потрібна для того щоб відобразити розклад та графік пар студента. Показує розклад групи, яку користувач обрав під час реєстрації. Є можливість переглянути розклад на сьогодні та на завтра.
- Кнопка «Де поїсти» потрібна для вибору місця де можна поїсти на території університету. При нажатті на кнопку меню, ви отримаєте сповіщення про заклади які є на території університету та кнопки з вибором типу закладу в якому планується перекусити А також можна переглянути схему розташування закладів, для того щоб зрозуміти куди йти ближче. Документи з закладами зберігаються в колекції cafe (рис.3.33). Всі заклади поділяються на буфети та їдальні. Після вибору типу закладу, шукаємо в колекції cafe документи з закладами обраного типу. Відображаємо всі можливі заклади які є на території університету. Після нажаття на назву закладу, шукаємо в базі інформацію по ньому і показуємо користувачу фото, опис закладу, розташування, час роботи, вартість комплексного обіду та посилання на меню


```
_id: ObjectId("60a27a039c77314402e3a15d")
name: "1 корпус"
adress: "внутрішній двір 5-го корпусу"
schedule: "09.00-17.00"
price: "25-30 грн"
description: "Їдальня центрального корпусу  із залою на 48 місць, обслуговує співроб..."
image: "https://i.ibb.co/Cz8q05N/buffet1.jpg"
menu: "https://relax.kpi.ua/csh/menyu/"
type: "cafe"
payload: "cafe_1"
```

```
_id: ObjectId("60a27ba99894c75094057695")
name: "5 корпус"
adress: "внутрішній двір 5-го корпусу"
schedule: "09.00-17.00"
price: "25-30 грн"
description: "У їдальні харчуються  студенти та викладачі університету. Зал вміщує 3..."
image: "https://i.ibb.co/3CKjh0H/buffet5.jpg"
menu: "https://relax.kpi.ua/csh/menyu/"
type: "cafe"
payload: "cafe_5"
```


Рис. 3.33 Документи в колекції cafe

- Кнопка «Відділення банку» потрібна, щоб пришвидшити студенту пошук відділення в районі університету. Всі відділеня зберігаються в колекції banks. Після нажаття на кнопку меню, відправляється запит в базу і дістаються назви всіх банків. Після вибору банку, дістаємо адреси всіх відділень цього банку. Після вибору відділення відправляємо користувачу карту з точкою, до якої можна прокласти маршрут. Відправлення карту відбувається за допомогою методу location. Процес даного функціоналу зображений на рисунку 3.35.

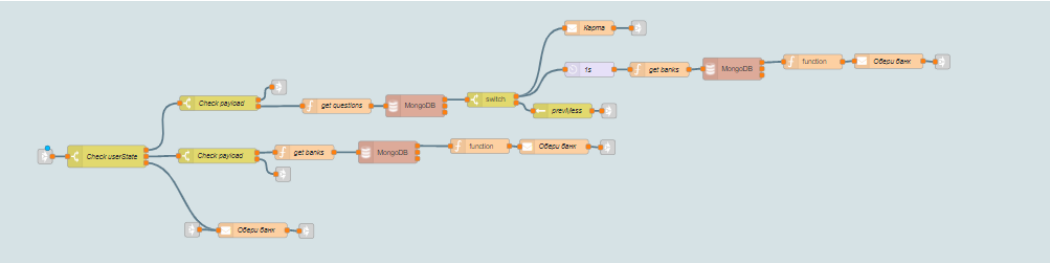


Рис. 3.34 Процес функціоналу «Банки» в Node-RED

- Кнопка «Поліклініка» відображає контактні дані та графік роботи поліклініки. Досить часто студентам потрібно дізнатися щось в поліклініці і вони витрачають багато часу на пошук контактних даних.
- Кнопка «Бази відпочинку» відображає можливі бази відпочинку від университету з посиланням на сайт де можна почитати інформацію та замовити путівку.
- Кнопка «Запит в деканат» потрібна для того щоб звернутися в деканат з питанням. Для того щоб задати питання потрібно натиснути на кнопку меню та ввести свій запит. Запит відправляється відповідальному від кожного факультету. Для цього дістаємо з колекції deanRequests (рис. 3.35) , channel та chatId відповідальної людини з цього факульету та відправляємо йому запит.

```
{
    "_id": {
        "$oid": "60ac2aef2fb73b5ce4c68df1"
    },
    "channel": "viber",
    "chatId": "dNAS3yWuLKZMPJ7ncxj+sA==",
    "faculty": "Факультет інформатики та обчислювальної техніки",
    "requests": [{
        "channel": "telegram",
        "chatId": 414279467,
        "name": "Іванов Іван Іванович",
        "phone": "+380987866643",
        "group": "IB-72",
        "status": "done",
        "dayRequest": "28.05",
        "timeRequest": "21:10",
        "unixTime": 1622225444,
        "question": "Коли буде стипендія?",
        "requestId": 918666,
        "answer": "Стипендія буде у двадцятих числах місяця"
    }]
}
```


Рис. 3.35 Колекція deanRequest

3.5.4 Функціонал адміністратора

Роль адміністратора займає одна людина від кожного факультету в університеті. Його задача обробляти запити студентів та відповідати на них. Меню адміністратора складається з трьох кнопок : «Очікують на відповідь», «Історія звернень» та «Налаштування».

- «Очікують на відповідь» - дана кнопка меню необхідна для перегляду нерозглянутих раніше запитів та відповіді на них. При нажатті на кнопку, програма шукає всі запити в базі та фільтрує їх по статусу та факультету (рис.3.36). Відображається інформація у вигляді каруселі з такими данними: дата та час подачі заявки, ім’я, телефон, група та саме питання

```
msg.requests = msg.payload.requests;

msg.requestItem = [];
for(var i=0;i<msg.requests.length;i++){
    if(msg.requests[i].status =="waitForAnswer"){

        msg.requestItem.push({
            "channel":msg.requests[i].channel,
            "chatId":msg.requests[i].chatId,
            "name":msg.requests[i].name,
            "phone":msg.requests[i].phone,
            "group":msg.requests[i].group,
            "status":msg.status,
            "dayRequest":msg.requests[i].dayRequest,
            "timeRequest":msg.requests[i].timeRequest,
            "question":msg.requests[i].question,
            "statusText":" Очікує ",
            "statusColor":"#FFD2D2",
            "requestId":msg.requests[i].requestId
        })
    }
}

msg.len = msg.requestItem.length;
msg.itemsCount = msg.requestItem.length;
return msg;
```

Рис. 3.36 Формування нерозглянутих запитів

Після розробки функціоналу для трьох ролей залишилося реалізувати сервіс розсилок та опитувань.

3.5.5 Інформаційна розсилка

Інформаційна розсилка це сервіс для розсилки повідолмень в месенджері по базі користувачів чат-боту. Розсилка може бути як на всю кількість користувачів, так і таргетована, тобто з використанням фільтрів, які відбирають лише необхідних користувачів. Фільтрувати можна лише по тим параметрам, які є в колекції users, а саме: вік, роль, факультет, група.

На даний момент існує два основних типи повідомлень, які можна комбінувати між собою, щоб створювати потрібну розсилку.

Табл 3.1 Типи повідомлень

Тип повідомлення	**Опис та необхідний формат**
Текст	Звичайний текст з необхідним форматуванням
Картинка	Дозволяє відправляти картинки та файли gif: Картинка: пряме посилання на картинку в форматі .jpg, .jpeg GIF файл: пряме посилання на файл в форматі .mp4

Формування розсилки відбувається за допомогою поєднання назв типів повідомлень, розділених нижнім підкресленням. Є декілька правил формування розсилок які потрібно дотримуватись:

- Тип розсилки передається як рядок в поле type в запиті.
- Порядок відправлення повідомлень залежить від порядку розміщення блоків з типами в полі type.
- Кожен із блоків в розсилці може використовуватись один раз.
- Якщо останній блок розсилки – текст, то разом з ним відправляється головне меню, якщо картинка, то головне меню відправляється окремо

Тобто якщо потрібно, щоб в розсилці була спочатку картинка а потім текст, потрібно в полі type написати image_text.

Запуск розсилки відбувається за допомогою HTTP запиту. Метод запиту – POST. URL запиту формується таким чином : {{serverUrl}}/mailing, де serverUrl – це сервер на якому знаходиться Node-RED.

Параметри необхідні для відправки запиту наведені в таблиці 3.2.

Табл 3.2 Параметри для запиту

Параметр	**Значення**	**Опис**
password	please	**Required**
mailingId	Наприклад: kpi_2805	**Required**
type	Наприклад: image_text	**Required**
image	*Пряме посилання*	**Optional**
text	*Будь-який текст*	**Optional**
action	mailing	**Required**
filter	Фільтр для бази данних	**Optional**

Повноцінний запит можна подивитися на рисунку 3.37.

```
{
    "password": "please",
    "type": "image_text",
    "image": "https://i.ibb.co/26mjD6N/kpi.jpg",
    "text": "Привіт, це перша розсилка",
    "mailingId": "image_text_test",
    "action" : "mailing"
}
```

Рис. 3.37 Приклад запиту для відправки розсилки

Можливі відповіді від API забражені в таблиці 3.3.

Табл 3.3 Можливі відповіді від API

Код	**Повідомлення**
200	Success!
401	Unauthorized
415	For this mailing type expected params: image,text
415	Warning: mailingId is required
415	Warning: wrong image link type
500	Internal server error

Процес розсилок складається з 4 частин :

1. Отримання та обробка запиту
2. Робота з лічильником аудиторії розсилки
3. Робота з базою та формування аудиторії розсилки
4. Відправка повідомлень користувачу

Розглянемо кожну частину окремо:

Отримання та обробка запиту (рис. 3.38).

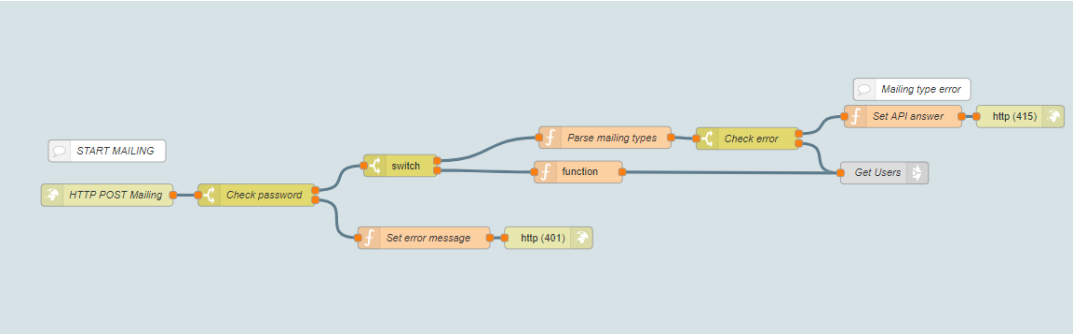


Рис. 3.38 Процес прийому та обробки запиту

Запит, для відправлення розсилки приходить в даний процес. Всі параметри які передаються зберігаються в об’єкті msg.payload. Спочатку перевіряємо правильність формування запиту. Нода Parse mailing types формує з отриманих даних список об'єктів, а також валідує значення які передаються. Код з коментарями зображений на рисунку 3.39. Якщо все правильно - переходимо до наступного етапу розсилки, інакше, відправляємо повідомлення про помилку і зупиняємо розсилку.

```
//Формуємо тип розсилки
msg.mailingTypes = msg.payload.type.split("_")
msg.mailingId = msg.payload.mailingId;
msg.usersCount = msg.payload.usersCount;
msg.action = msg.payload.action;

// Назначаем доступні формати розсилок
var imageTypes = ["jpg", "jpeg", "mp4", "gif"]
msg.mailingData = []; //список обьектов сообщений
var data;

// Проходимся за списком повідомлень для розсилки. Формуємо з отриманих даних список об'єктів повідомлень
// Один об'єкт повідомлення зберігає в собі тип повідомлення і контент
// Якщо тип посилання не валідна, сетім msg.errorString з описом помилки
msg.mailingTypes.map((element) => {
    if(element.includes("image") && msg.payload.image){
            if (!imageTypes.includes(msg.payload.image.split(".").slice(-1)[0])){
                msg.errorString = `Warning: wrong image link type. Supported link type is .jpg .jpeg (or .mp4 for gif)`;
            }
            data = {
                "type" : "image",
                "link" : msg.payload.image
            }
            msg.mailingData.push(data)
    }
    if(element.includes("text") && msg.payload.text){
            data = {
                "type" : "text",
                "text" : msg.payload.text
            }
            msg.mailingData.push(data)
    }

});
```

Рис. 3.39 Приклад формування розсилки

Робота з лічильником аудиторії розсилки (рис. 3.39).

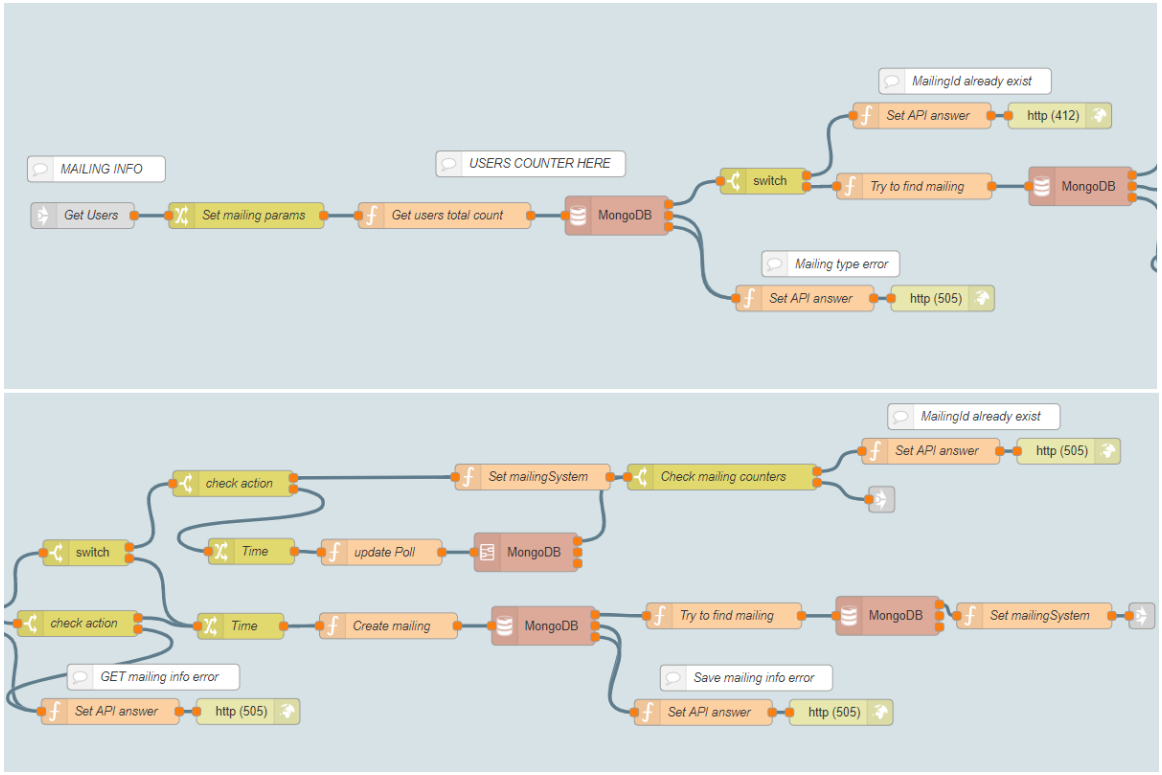


Рис. 3.39 Процес роботи з лічильником аудиторії

У блоці Set mailing params вказані параметри для розсилки. Одним з параметрів є msg.itemsCount, де вказано кількість користувачів за один запит. Стандартно вказано 50 користувачів.

Нода Get user total count робить запит до бази і повертає кількість користувачів. Якщо запит пройшов успішно переходимо до наступного етапу, де шукаємо розсилку в базі для того щоб зрозуміти, розсилка нова чи вже запускалась. Якщо такої розсилки ще немає, то створюємо. Якщо така розсилка вже існує і вона ще не закінчилася, то продовжуємо її з того моменту де вона зупинилась.

Процес формування аудиторії розсилки. Ця частина процесу знаходиться всередині циклу. В ноді get part of users відбувається набір по 50 людей з всієї аудиторії. Потім для кожного користувача з набору, проводимо приведення його данних до стандартного вигляду і відправляємо в наступну частину процесу, і в цей час відбувається набір наступної порції користувачів.

Відправка повідомлень користувачу (рис. 3.40).

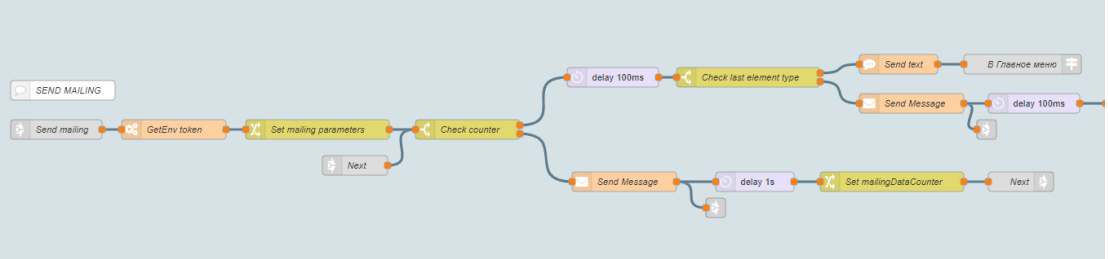


Рис. 3.40 Процес відправки розсилки

Спочатку відбувається перевірка скільки елементів в розсилці. Беремо по черзі елементи розсилки і перевіряємо останній він чи ні. Якщо елемент розсилки останній, то після повідомлення відправляємо головне меню.

Після того як всім користувачам була відправлена розсилка, вона стає неактивною і більше не можна буде відправити розсилку з таким самим ідентифікатором.

3.5.5 Масове опитування

Процес опитування відбувається в тому ж самоу процесі що й іформаційна розсилка, але змінюється формат запиту та підготовки контенту.

Перед тим як відправити запит, потрібно в базі підготувати контент.В колекції pollHistory необхідно вказати такі значення як mailingId, active та questions, як зображено на рисунку 3.41.


```
{
    "_id": {
        "$oid": "60af8818e981fb5aec816d7d"
    },
    "mailingId": "poll3",
    "active": true,
    "questions": [{
        "id": "poll3_1",
        "text": "💬 Запитання 1",
        "type": "open"
    }, {
        "id": "poll3_2",
        "text": "💬 Запитання 2",
        "type": "open"
    }, {
        "id": "poll3_3",
        "text": "💬 Запитання 3",
        "type": "open"
    }, {
        "id": "poll3_4",
        "text": "💬 Запитання 4",
        "type": "open"
    }, {…}]
}
```


Рис. 3.41 Шаблон для відправки опитування

Після того як успішно доданий шаблон в базу. Можна готувати запит. Параметри необхідні для відправки запиту зображені в таблиці 3.4.

Табл 3.2 Параметри для відправки опитування

Параметр	**Значення**	**Опис**
password	please	**Required**
mailingId	Наприклад: poll_2805	**Required**
action	poll	**Required**
filter	Фільтр для бази данних	**Optional**

Приклад запиту зображений на рисунку 3.42.

```
1  {
2      "password": "please",
3      "mailingId": "poll3",
4      "filter":{},
5      "action": "poll"
6  }
```

Рис. 3.41 Приклад запиту для відправки опитування

В іншому відмінностей між відправкою інформаційних розсилок та опитувань немає, так як використовується одна і та сама логіка розсилки та один і той самий процес.

Розсилки є гарним сервісом для сповіщення студентів про новини в університеті чи факульеті, а опитування один із способів зібрати інформацію від студентів на питання які турбують.

ВИСНОВОК ДО РОЗДІЛУ 3

В рамках цього розділу було розглянуто реалізацію системи для комунікацій у вищому навчальному закладі на основі чат-боту. Проект реалізований за допомогою іструменту Node-RED та мови програмування JavaScript.

Було описано хід розробки чат-боту для Telegram та Viber, реалізацію універсальної бот платформи, інтеграції з Google Sheets, розробка структури бази данних, функціоналу абітурієнта, студента та панель адміністратора. Окрім цього було описано реалізацію сервісу для розсилки інформаційний повідомлень та опитувань.

В результаті виконання було створено зручний та корисний бот для вищого навчального закладу. Розроблено універсальну бот платформу, яка легко масштабується для інших месенджерів, та зручна у використанні. А також створено сервіс розсилки в чат-боті по базі користувачів

РОЗДІЛ 4
ОГЛЯД РОБОТИ СТВОРЕНОЇ СИСТЕМИ ДЛЯ КОМУНІКАЦІЇ НА ОСНОВІ ЧАТ-БОТУ

4.1 Реєстрація

Після переходу за посиланням в чат-бот, користувачу відразу відправляється текст з привітанням, та кнопка для початку реєстрації(рис. 4.1).

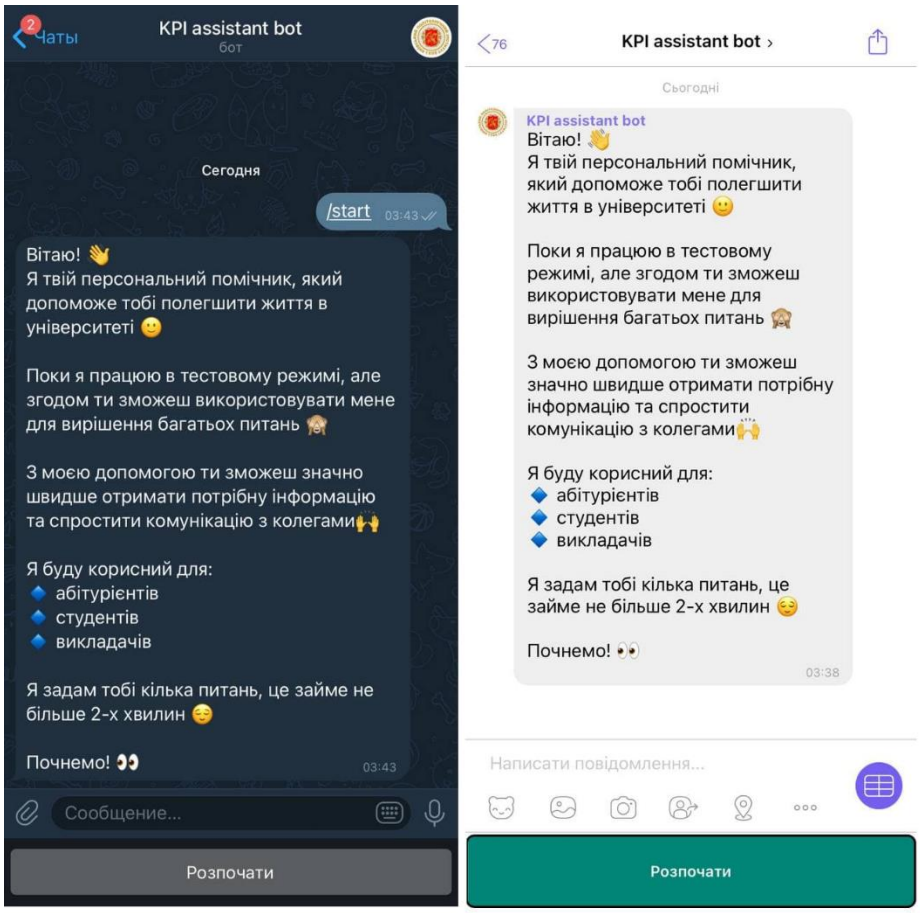


Рис. 4.1 Текст привітання для Telegram та Viber

Для того щоб почати реєстрацію, потрібно натиснути кнопку «Розпочати» та чат-бот почне запитувати анкетні дані: ім’я, вік, телефон (рис.4.2).

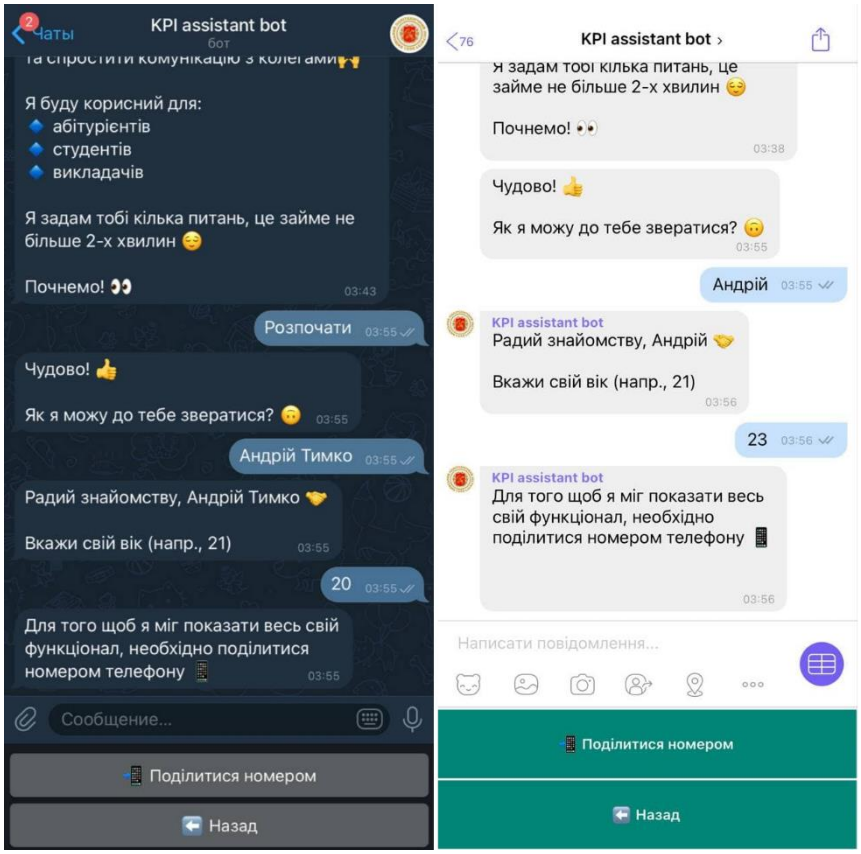


Рис. 4.2 Збір анкетних даних в чат-боті

Після того як користувач поділився номером телефону за допомогою кнопки, йому пропонується обрати роль до якої він належить: абітурієнт, студент, деканат (рис. 4.3).

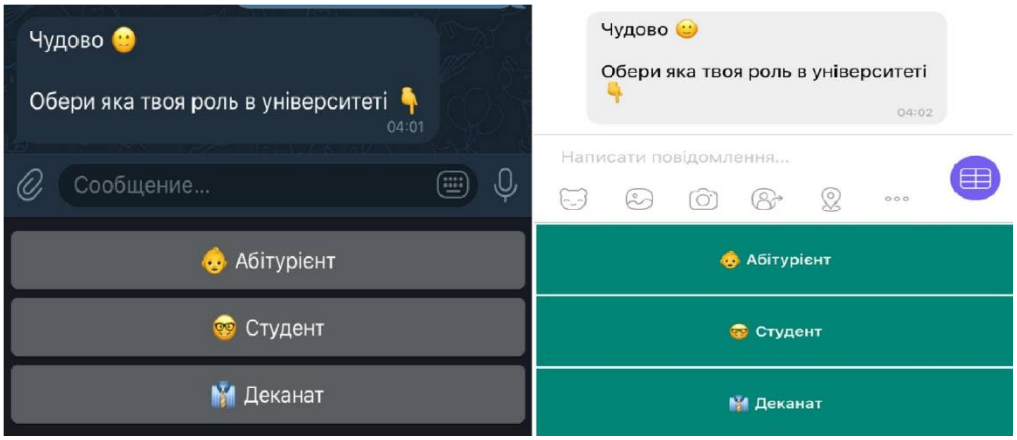


Рис. 4.3 Вибір ролі користувача

Далі процес реєстрації розділяється на три гілки, в залежності від того яку роль обрав користувач.

Якщо користувач обрав роль абітурієнт, реєстрація на цьому закінчена і користувачу відправляється головне меню (рис.4.4).

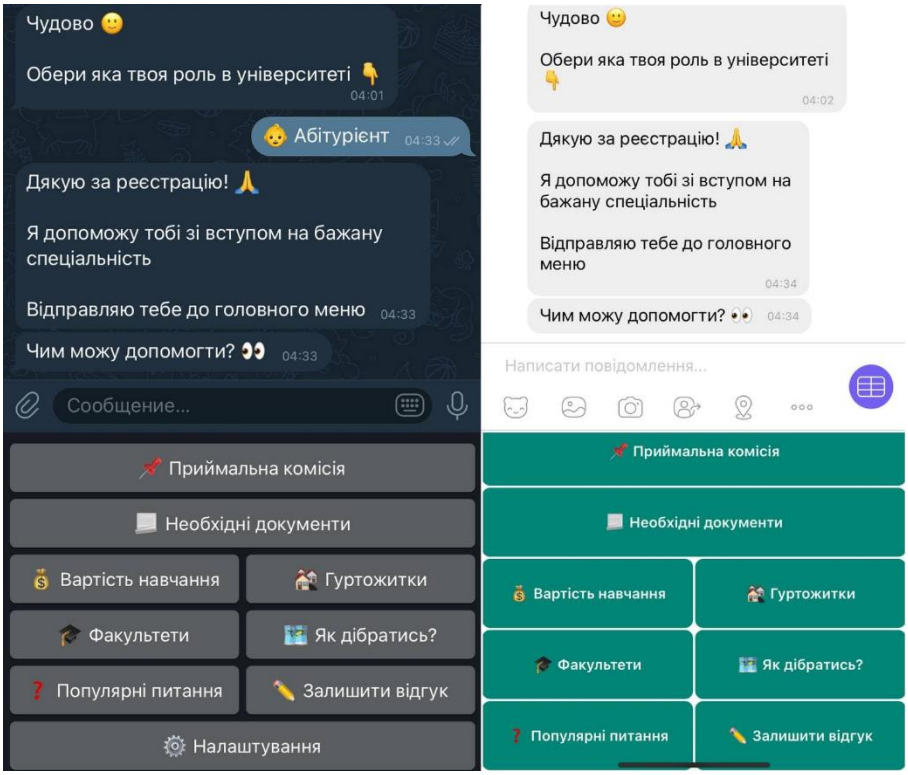


Рис. 4.4 Головне меню абітурієнта

Якщо користувач обрав роль студент, то потрібно пройти ще два додаткових кроки для визначення факультету і групи в якій навчається користувач(рис.4.5). Якщо все обрано коректно – користувачно відправиться головне меню студента.

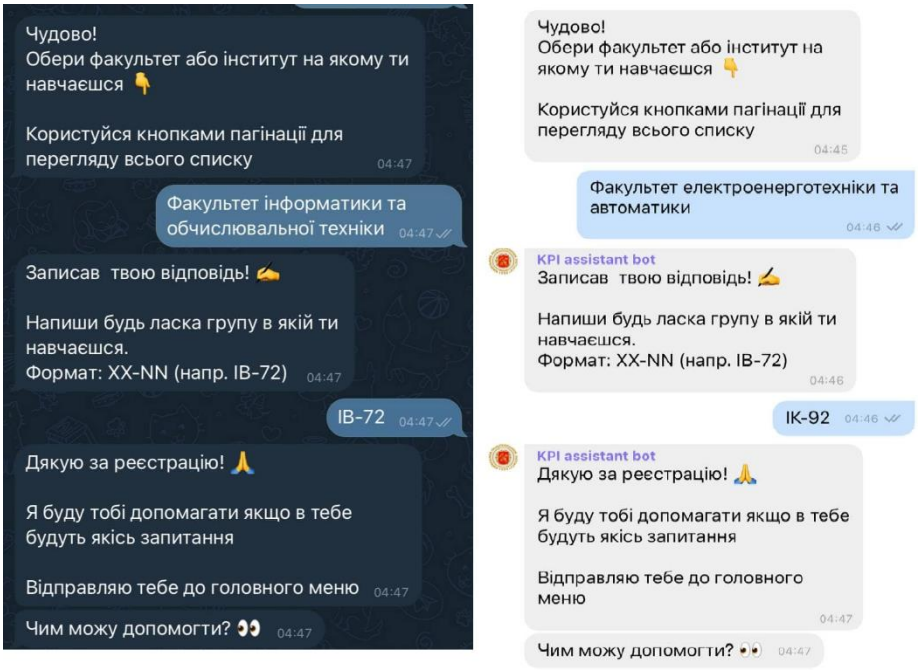


Рис. 4.5 Реєстрація студента

Якщо користувач обрав роль деканат, то відбувається пошук по таблиці з адміністраторами по номеру телефона. Якщо номер телефона який ввів користувач в бот співпав з одним із таблиці, користувачу відправляється повідомлення з його данними і прохання підтвердити що це він. Якщо користувач підтверджує, йому відправляється привітання з успішно пройденою верифікацією (рис.4.6).

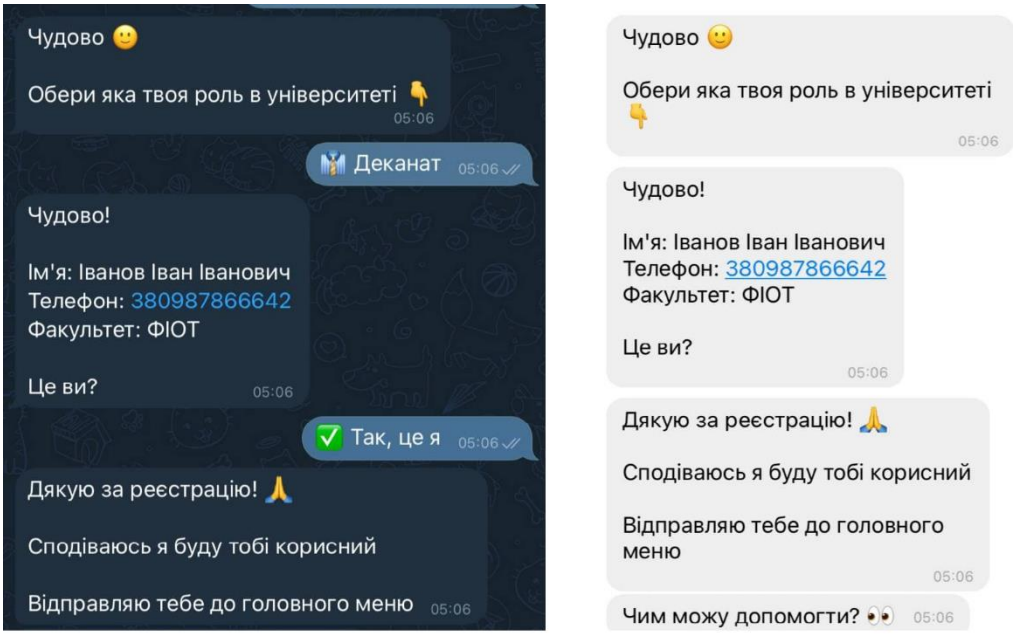


Рис. 4.5 Успішна реєстрація адміністратора

Таким чином відбувається реєстрація для всіх ролей в боті. Розглянемо функціонал кожної ролі окремо.

4.2 Функціонал абітурієнта

Головне меню абітурієнта див. рис. 4.4 складається з 9 кнопок. Кожна кнопка виконує окремий функціонал.

Кнопка «Приймальна комісія» приховує в собі ще 5 кнопок (рис. 4.6), які ведуть на інший веб-сай або виводять коротку інформацію у вигляді тексу.

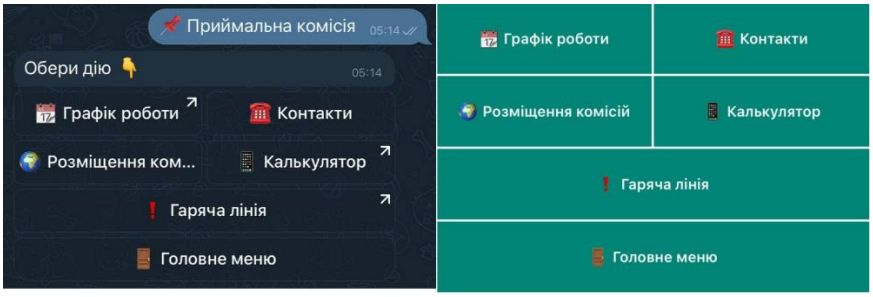


Рис. 4.6 Кнопки в меню приймальна комісія

Кнопка меню «Гуртожитки» пропонує абітурієнту обрати гуртожиток та переглянути по ньому всю необхідну інформацію. Приклад відображення для Telegram зображений на рис. 4.7.

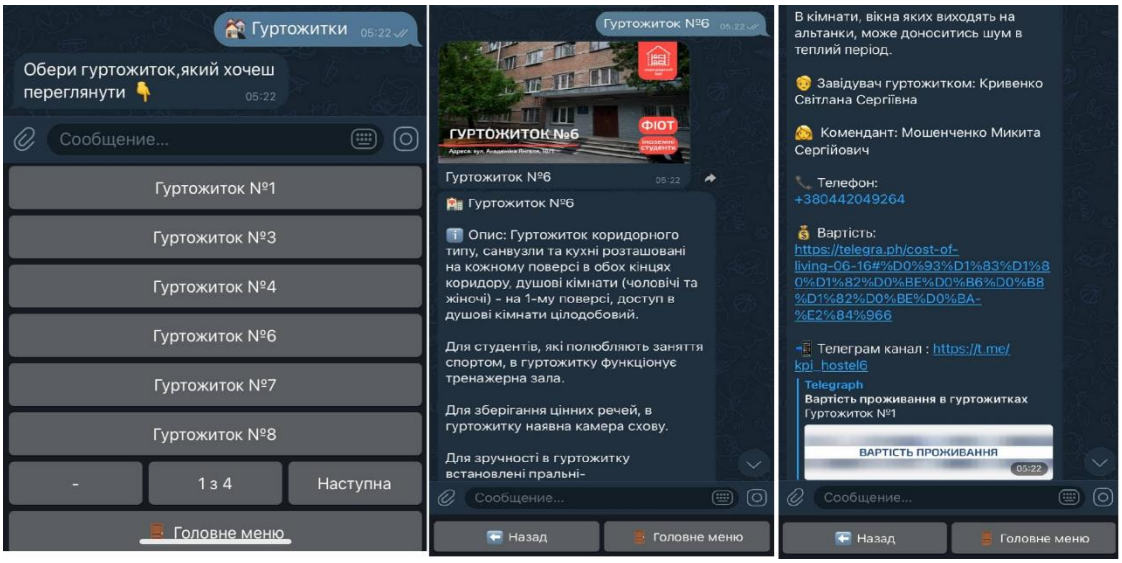


Рис. 4.7 Приклад відображення інформації про гуртожитки

Відображення цього функціоналу для месенджера Viber аналогічний, тому

Кнопка меню «Гуртожитки» спочатку виводить список всіх факультетів в університеті(рис. 4.8).

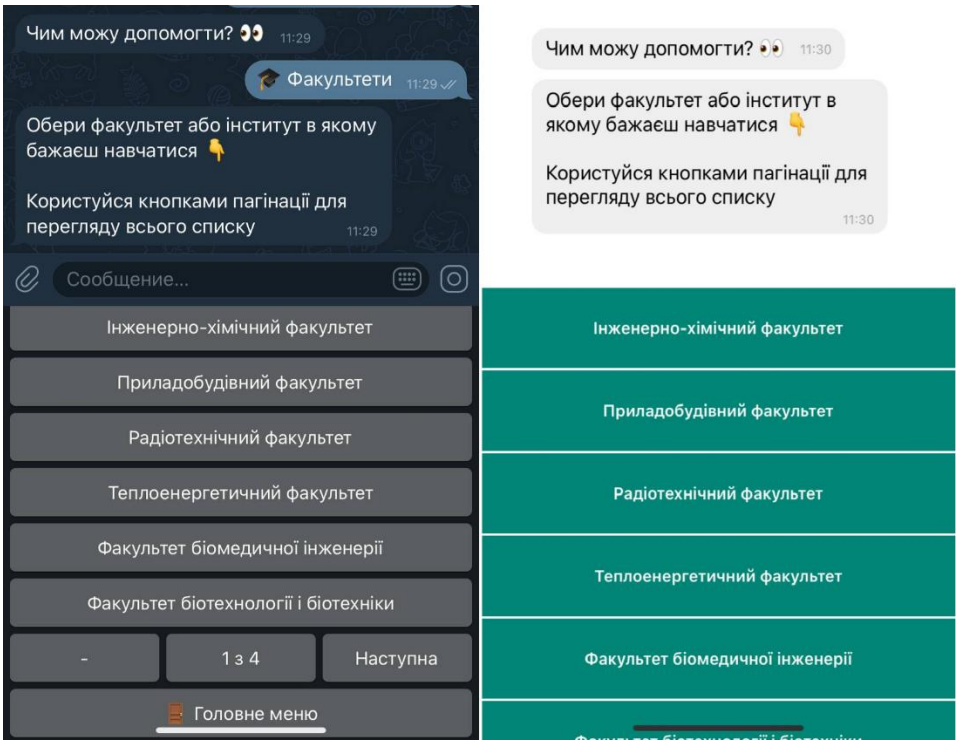


Рис. 4.8 Етап вибору факультету

Після того як студент обрав факультет, відображаємо користувачу інформацію про факультет (рис. 4.9): повна назва, ім'я декану, адреса, телефони для довідок, веб-сайт факультету.

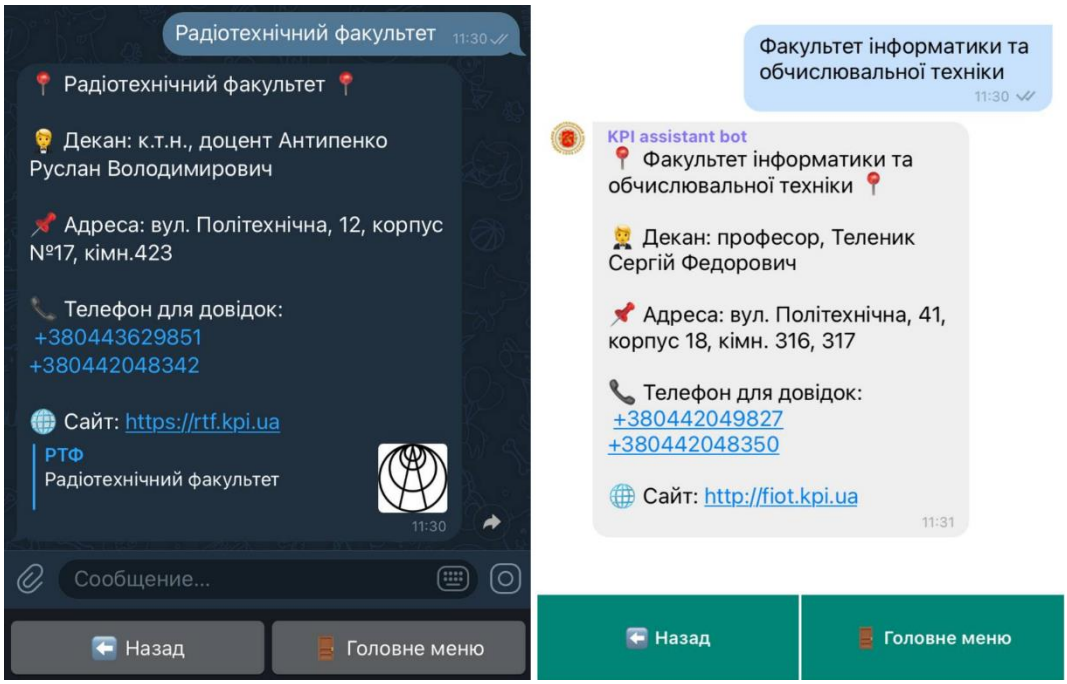


Рис. 4.9 Приклад відображення інформації про факультет

Кнопка «Як дібратись» пропонує абітурієнту переглянути список можливих маршрутів до університету (рис. 4.10).

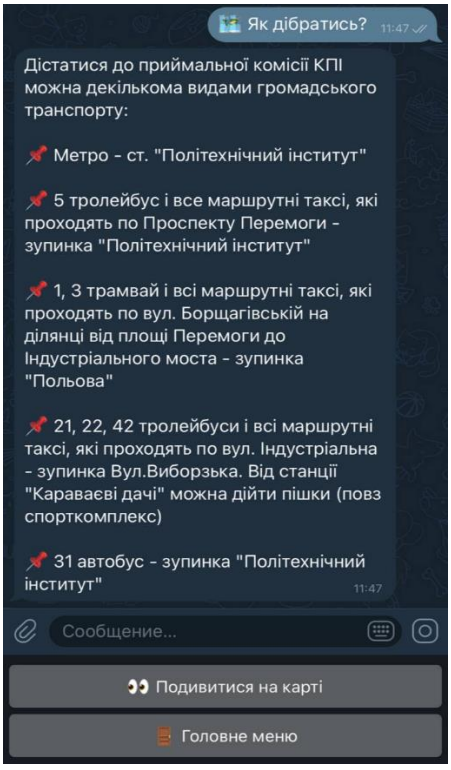


Рис. 4.10 Відображення кнопки «Як дібратись» в Telegram

Кнопка «Популярні питання» відображає питання які найчастіше задаються (рис. 4.11) і користувач може відразу отримати на них відповідінатиснувши на питання яке цікавить. Питання відображаються по 6 на кожній сторінці.

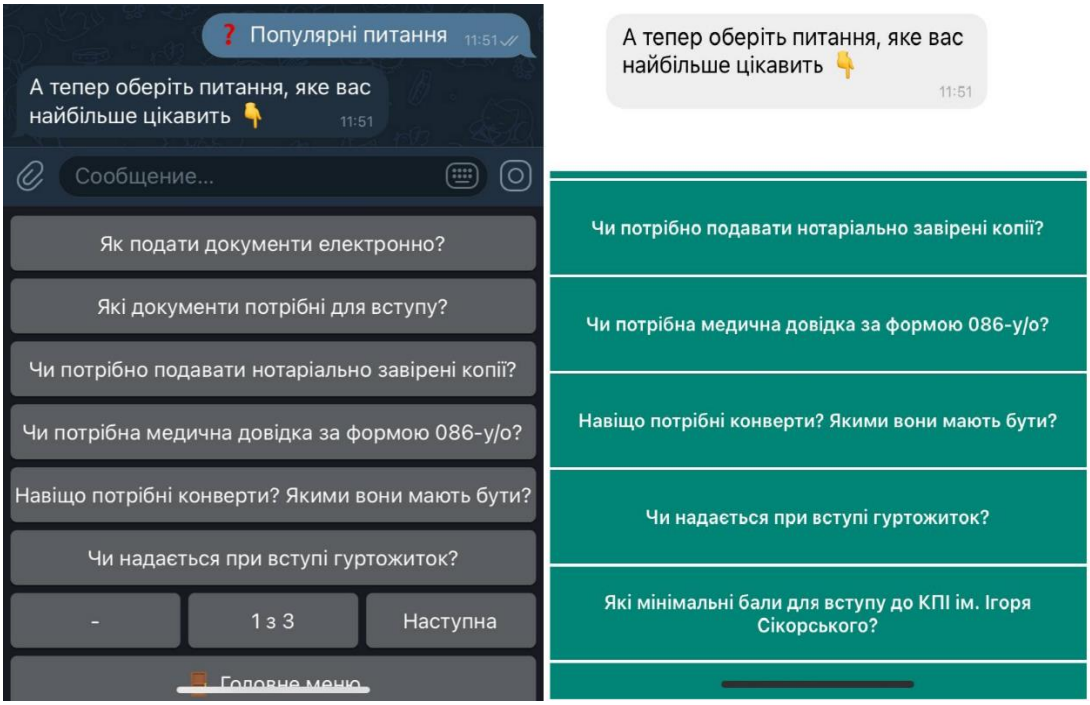


Рис. 4.11 Приклад відображення популярних питань

Кнопка «Залишити відгук» потрібна для того щоб абітурієнти могли залишити своє враження від університету та приймальної комісії (рис. 4.12).

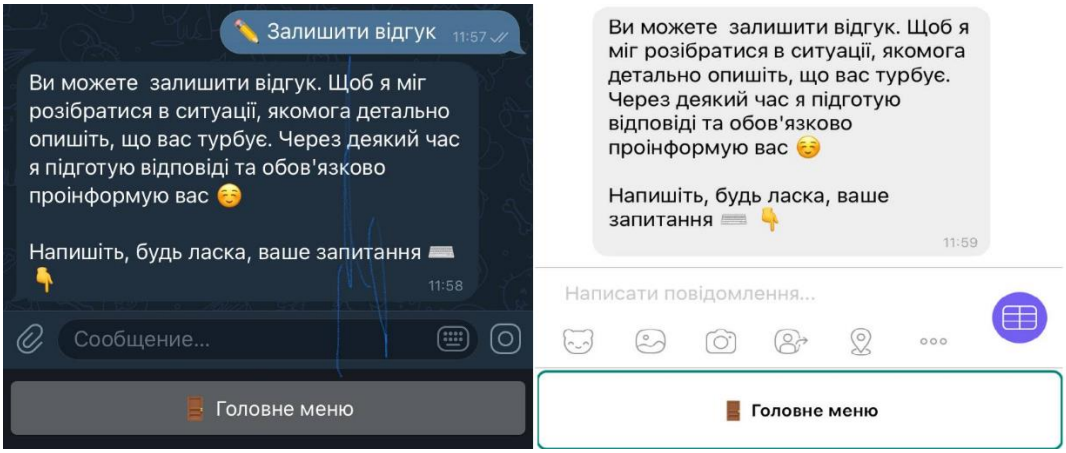


Рис. 4.12 Функціонал кнопки «Залишити відгук»

Після того як користувач написав свій відгук , він відображається в таблиці з такою інформацією: дата, ім’я, телефон, відгук. Вигляд таблиці зображений на рис. 4.13.

	A	B	C	D	E
1	Дата	Ім'я	Телефон	Роль	Відгук
2	30.05.2021	Іванов Іван Іванович	380987866642	Абітурієнт	Все добре!
3					

Рис. 4.13 Відображення відгуків в Google Sheets

4.3 Функціонал студента

Меню студента зображено на рис. 4.14. налічує в собі 10 кнопок, кожна з яких відповідаю за окремий функціонал.

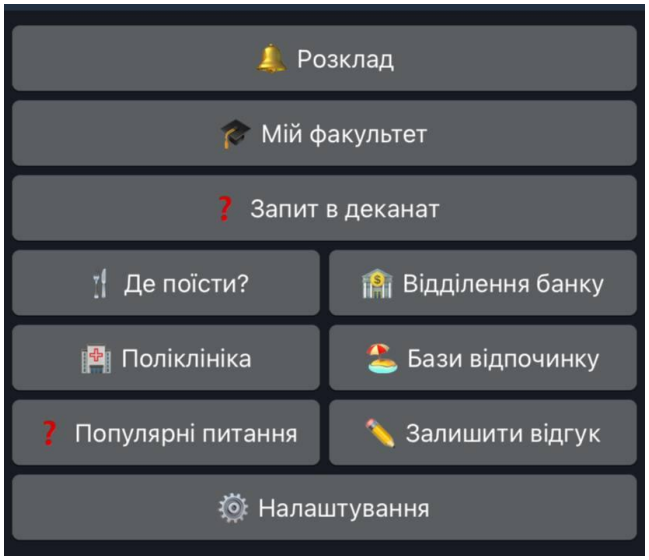


Рис. 4.13 Вигляд меню студента

Кнопка «Розклад» потрібна для відображення графіку та розкладу пар. Розклад пар шукається по групі, яку студент вказав на етапі реєстрації. Графік пар відображається в текстовому вигляді (рис. 4.15).

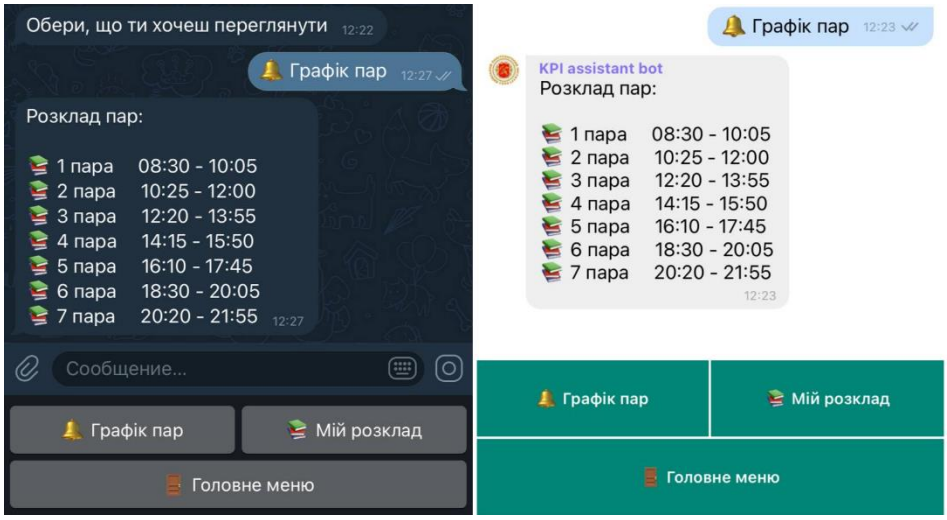


Рис. 4.15 Відображення графіку пар

Кнопка «Де поїсти» потрібна щоб студент за пару кліків зміг найти заклад де він може поїсти. При нажатті на кнопку меню, користувачу вибирає який тип закладу (буфет чи столова) він хоче відвідати. Також можна подивитися схему розташування всіх закладів на території університету , натиснувши на кнопку «Схема розташування об’єктів харчування», після чого з’явиться фото території з відміченими точками з закладами. Якщо користувач обрав тип закладу, пропонуємо йому обрати корпус який студент бажає відвідати. Після вибору корпусу відображаємо всю потрібну інформацію для ознайомлення з закладом, а саме : фото, опис, розташування, графік роботи, орієнтовна вартість обіду, посилання на меню. Етапи вибору закладу зображені на рис.4.16.

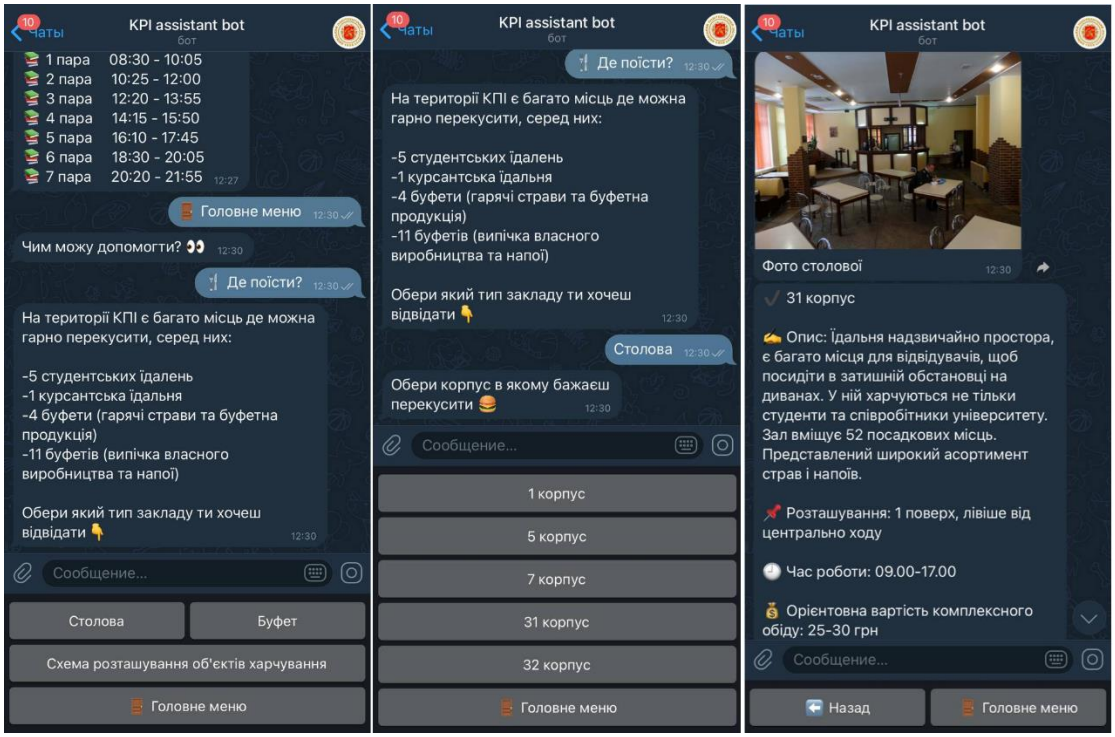


Рис. 4.16 Етапи вибору закладу харчування

Кнопка «Відділення банку» потрібна для того щоб знайти найближче відділення банку.. Спочатку користувач обирає банк, відділення якого треба знайти. Далі відображається список вулиць, на яких є банкомати цього банку. Після вибору вулиці, студенту приходить місцезнаходження банкомату до якого він може прокласти маршрут (рис.4.16).

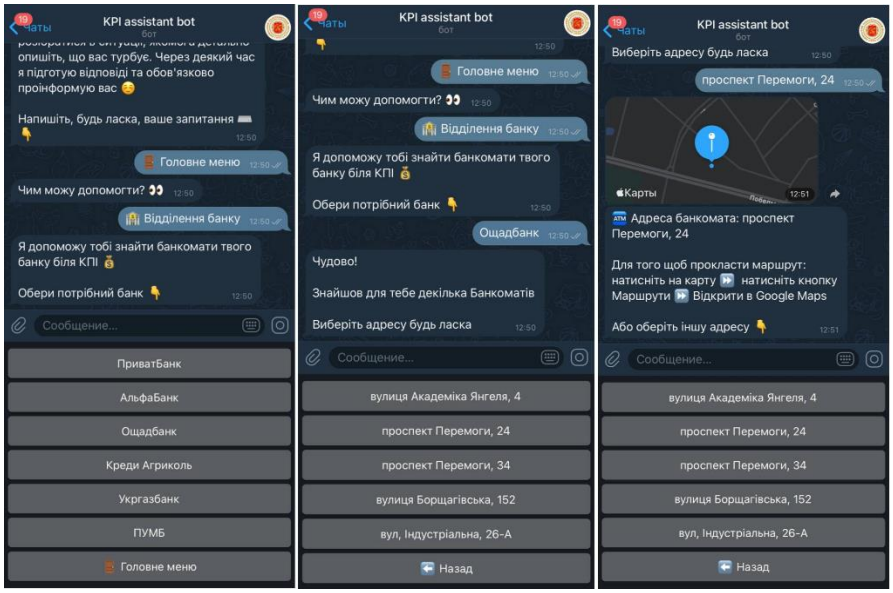


Рис. 4.16 Етапи вибору відділення банку

Кнопка «Поліклініка» відображає контактні дані та графік роботи студентської поліклініки (рис. 4.17).

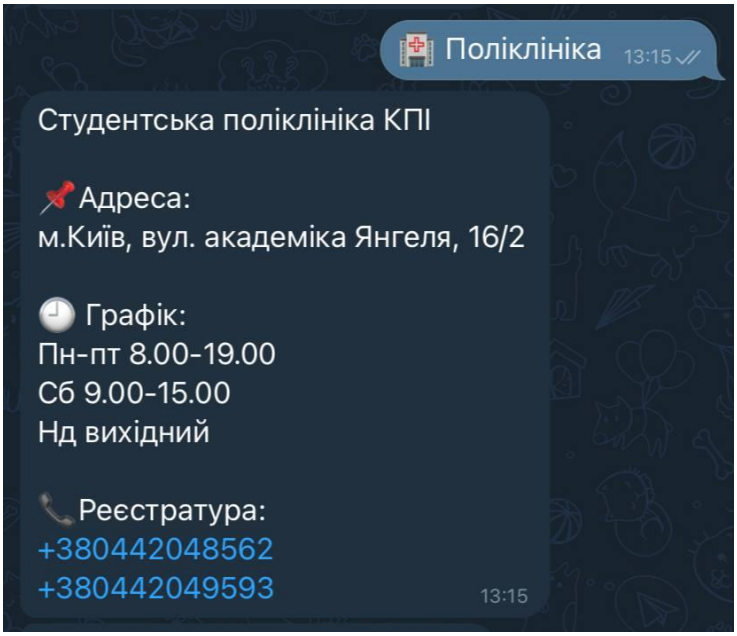


Рис. 4.17 Інформація про студентську поліклініку

Кнопка «Бази відпочинку» відображає кнопки з базами відпочинку. При нажатті на кнопку користувач переходить за посиланням на веб-сайт де можна замовити путівку в обрану базу (рис.4.18).

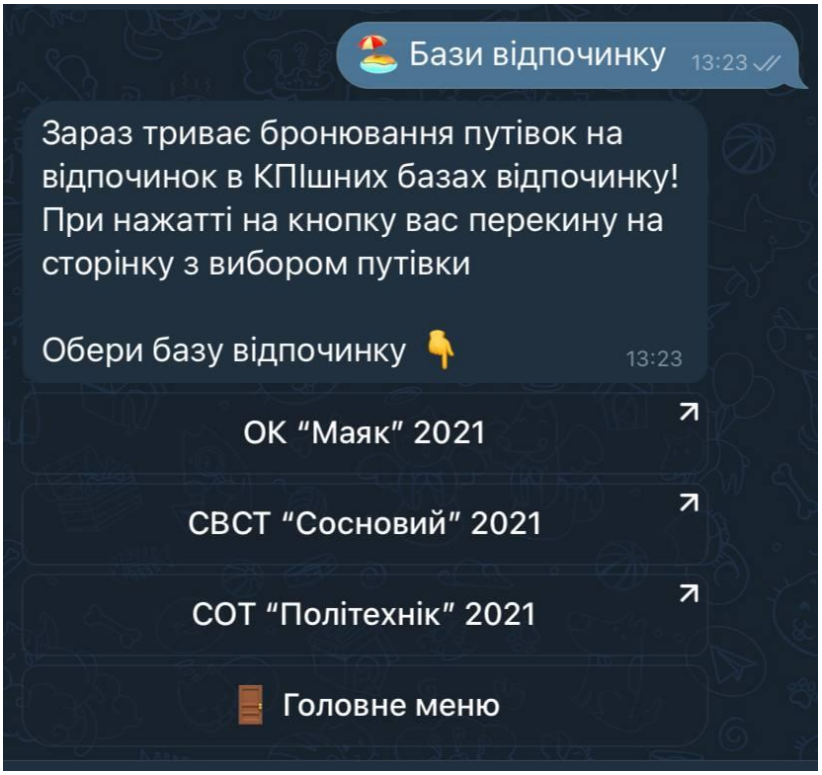


Рис. 4.18 Функціонал кнопки «Бази відпочинку»

4.4 Функціонал адміністратора

Функціонал адміністратора складається з трьох кнопок зображених на рис. 4.19.

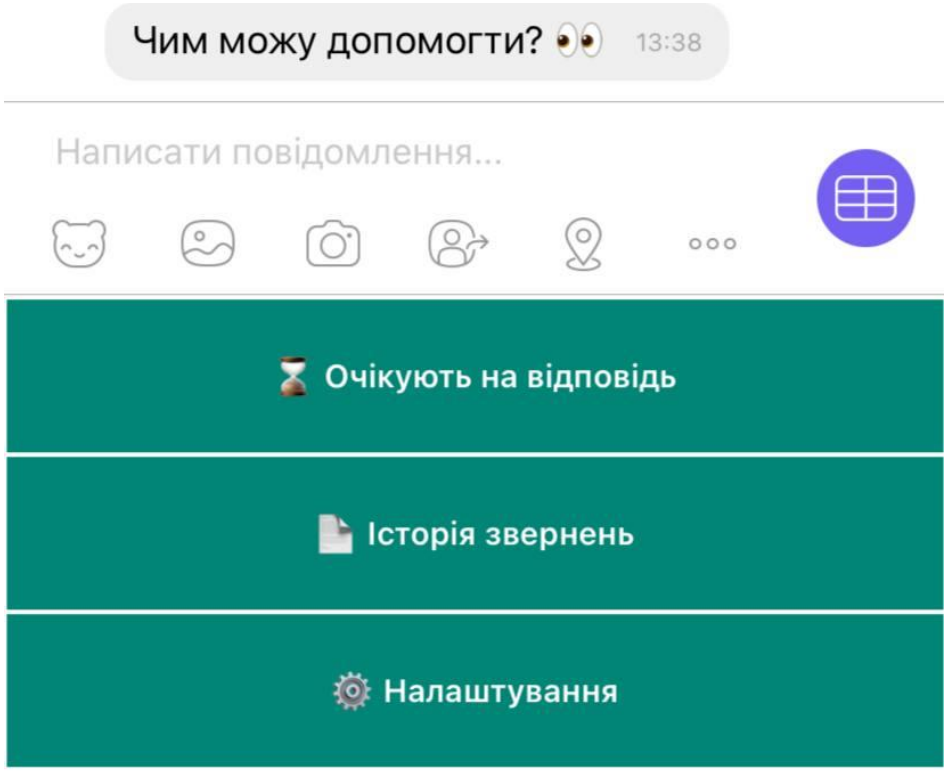


Рис. 4.19 Вигляд меню адміністратора

До функціоналу адміністратора відноситься перегляд нерозглянутих запитів в деканат та перегляд минулих запитів. Запити поступають після того як студент за допомогою кнопки меню «Запит до деканату» відправив свій запит. В цей же час запит потрапляє до списку нерозглянутих. Вигляд нерозглянутих запитів в панелі адміністратора зображений на рис. 4.18.

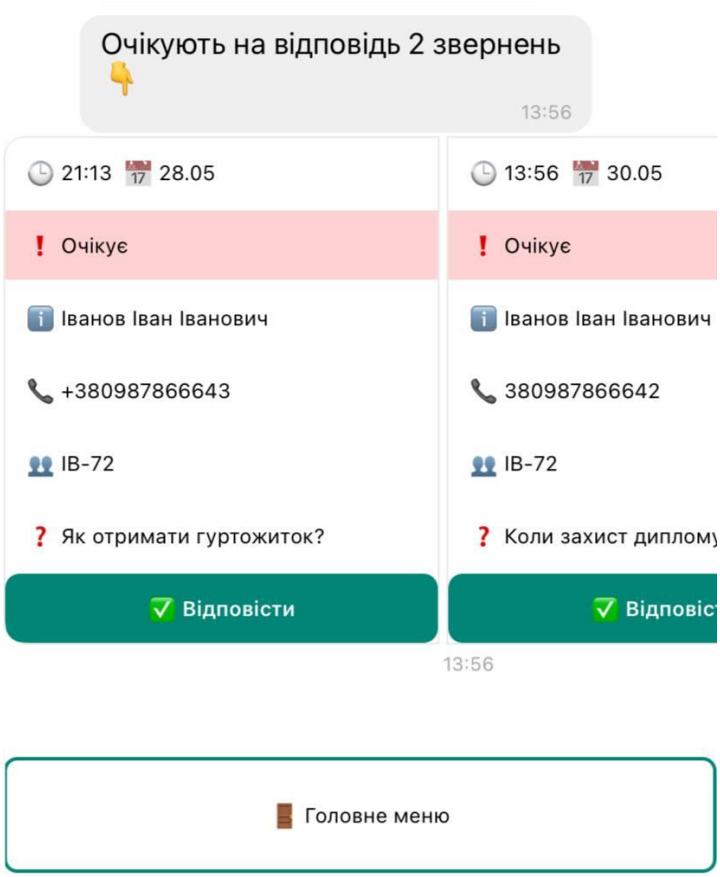


Рис. 4.19 Нерозглянуті запити студентів

Як видно на рисунку, в запиті вказується дата та час коли прийшла анкета, статус анкети,ім'я, номер та група студента, а також саме питання. Для того щоб відповісти на питання, потрібно натиснути на кнопку «Відповісти». Після чого адміністратору запропонується написати відповідь. Після відправки відповіді вона відразу приходить студенту(рис. 4.20) в бот,а її статус змінюється на вирішений.

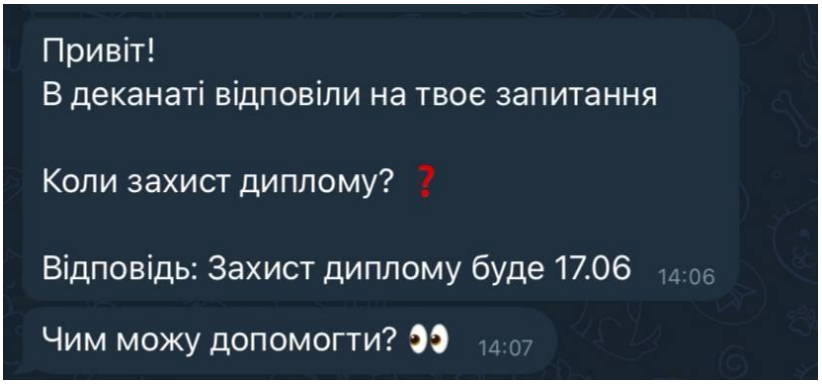


Рис. 4.20 Відповідь на запит студента

Після того як на запит відповіли, його можна побачити натиснувши на кнопку меню «Історія звернень». Можна переглянути питання яке було задано, та відповідь на нього (рис.4.21).

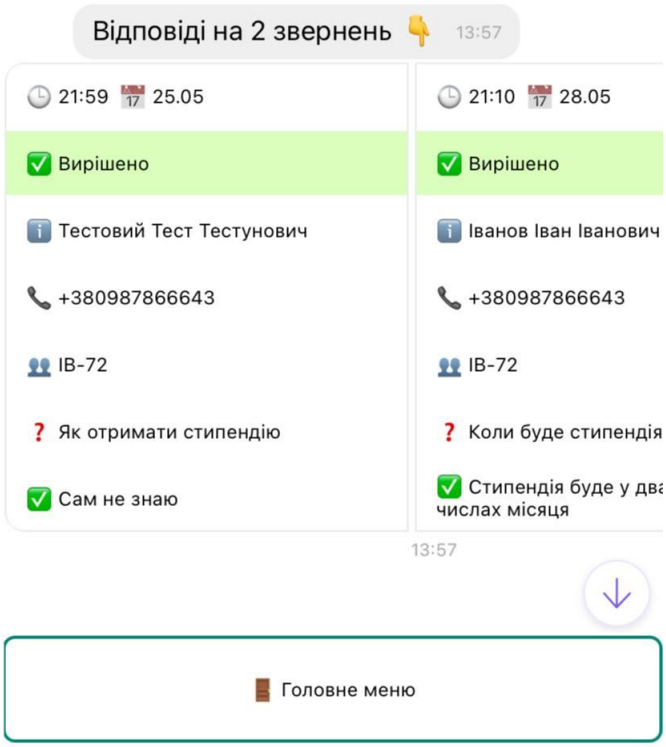


Рис. 4.20 Історія звернень до адміністратора

Окрім цього в панелі адміністратора є кнопка «Налаштування» , вона зроблена для того щоб в будь-який момент переглянути як функціонал абітурієнта, так і студента, для тог щоб перевірити справність функціоналу.

ВИСНОВОК ДО РОЗДІЛУ 4

В цьому розділі розглянуто приклади використання системи для комунікації у вищому навчальному закладі на основі чат-боту. Було представлено реєстрацію в чат-боті, рольову модель та всі можливі сценарії в чат-боті. Розглянуто панель адміністратора та принцип взаємодії адімінстратора та студента.

Таким чином цей розділ демонструє, що система, реалізована в третьому розділі повністю справна, має весь зазначений фукціонал, має зрозумілий інтерфейс та швидкий час відповіді від бота.

ВИСНОВКИ

Метою дипломного проекту є розробка системи для комунікації у вищому навчальному закладі на основі чат-боту. Існує багато чат-ботів,кожний з яких виконує окрему функцію, але в цьому чат-боті поєднаний весь функціонал.

В першому розділі було проаналізовано проблему комунікації у вищих навчальних закладах, визначено причини а способи подолання цієї проблеми. Розглянуто харакетеристики месенджерів Telegram та Viber. Розглянуто приклади чат-ботів у різних сферах бізнесу та проблеми які вони покривають

В другому розділі було описано вибрані технології та мови програмування для розробки системи для комунікації. Головним інструментом для розробки є інструмент візуального програмування Node-RED. В якості мови програмування використовується JavaScript. Також в цьому розділі описано методи роботи з Telegram Bot API та Viber Bot API.

Третій розділ розкриває хід розробки чат-боту для Telegram та Viber, реалізацію універсальної бот платформи, інтеграції з Google Sheets, розробка структури бази данних, функціоналу абітурієнта, студента та панель адміністратора. Окрім цього було описано реалізацію сервісу для розсилки інформаційний повідомлень та опитувань.

Четвертий розділ демонструє зовнішній вигляд чат-боту. Було представлено реєстрацію в чат-боті, рольову модель та всі можливі сценарії в чат-боті. Розглянуто панель адміністратора та принцип взаємодії адімінстратора та студента. Показано, що система повністю справна, має весь зазначений фукціонал та має зрозумілий інтерфейс.

Отже, описана в дипломному проекті система для комунікації у вищих навчальних закладах на основі чат-боту відповідає поставленим вимогам, оптимізує пошук потрібної інформації та полегшує комунікацію всередині навчального закладу за рахунок інфомаційних розсилок, опитувань та можливості студентам давати зворотній зв’язок.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Bots: An introduction for developers [Електронний ресурс] – Режим доступу до ресурсу: https://core.telegram.org/bots
2. Viber REST API [Електронний ресурс] – Режим доступу до ресурсу: https://developers.viber.com/docs/api/rest-bot-api/
3. What is an API [Електронний ресурс] – Режим доступу до ресурсу: https://www.mulesoft.com/resources/api/what-is-an-api
4. HTTP Request Methods [Електронний ресурс] – Режим доступу до ресурсу: https://www.mulesoft.com/resources/api/what-is-an-api
5. MongoDB [Електронний ресурс] – Режим доступу до ресурсу: https://www.mongodb.com/what-is-mongodb
6. Node-RED [Електронний ресурс] – Режим доступу до ресурсу: https://nodered.org/
7. Everything you need know about Node-RED [Електронний ресурс] – Режим доступу до ресурсу: https://www.opensourceforu.com/2017/09/node-red/
8. Send messages using Telegram and Node-RED [Електронний ресурс] – Режим доступу до ресурсу: https://www.iotwithus.com/send-messages-using-telegram-and-node-red/
9. How to Create Telegram Bots Using Webhooks [Електронний ресурс] – Режим доступу до ресурсу: https://levelup.gitconnected.com/
10. An Introduction to MongoDB [Електронний ресурс] – Режим доступу до ресурсу: https://www.sitepoint.com/an-introduction-to-mongodb/
11. Local Installs and Managing Node-Red Projects [Електронний ресурс] – Режим доступу до ресурсу: https://stevesnoderedguide.com/managing-node-red-projects

ДОДАТОК 1

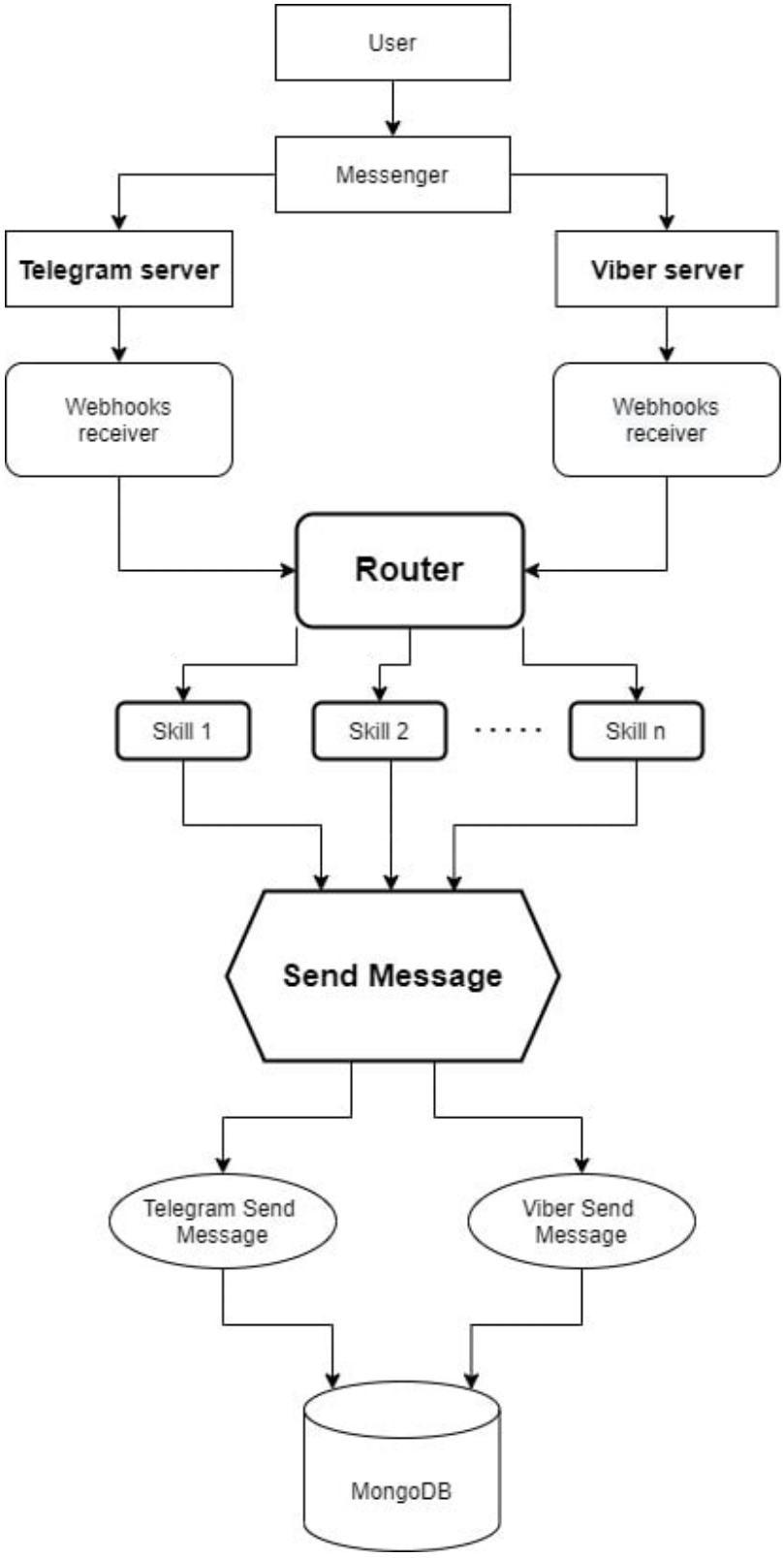
Система для комунікації у вищому навчальному закладі на основі чат-боту

Схема структурна

ІАЛЦ.467100.004 Д1

Аркушів 1

Київ – 2021 р.



					ІАЛЦ.467200.004 Д1	
Зм.	*Арк.*	*№ докум.*	*Підпис*	*Дата*		
Розробив		*Тимко А.О.*			*Система для комунікації в навчальному закладі на основі чат-боту* *Схема структурна*	*Літ.* / *Аркуш* 1 / *Аркуш ів* 1
Перевірив		*Ткаченко В.В.*				
Реценз.						*НТУУ «КПІ», ФІОТ, ІВ-72*
Н. Контр.		*Сімоненко В.П.*				
Затв.		*Стіренко С.Г.*				

ДОДАТОК 2

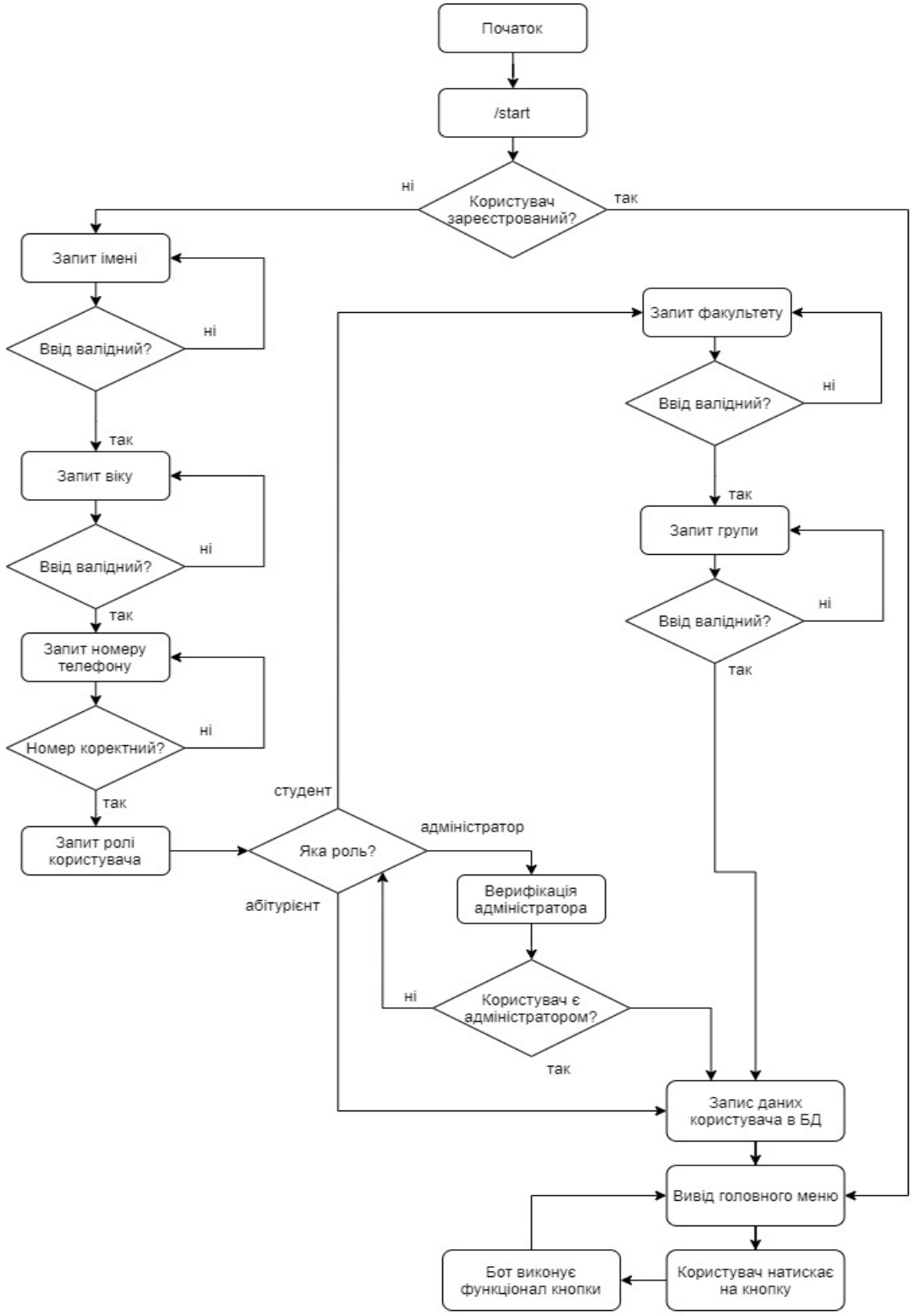
Система для комунікації у вищому навчальному закладі на основі чат-боту

Схема функціональна

ІАЛЦ.467100.005 Д2

Аркушів 1

Київ – 2021 р.



					ІАЛЦ.467200.004 Д2		
Зм.	*Арк.*	*№ докум.*	*Підпис*	*Дата*			
Розробив		*Тимко А.О.*			*Система для комунікації в навчальному закладі на основі чат-боту*	*Літ.*	*Аркуш*	*Аркуш ів*
Перевірив		*Ткаченко В.В.*					*1*	*1*
Реценз.					*Схема функціональна*		
Н. Контр.		*Сімоненко В.П.*				*НТУУ «КПІ», ФІОТ, ІВ-72*	
Затв.		*Стіренко С.І.*					

ДОДАТОК 3

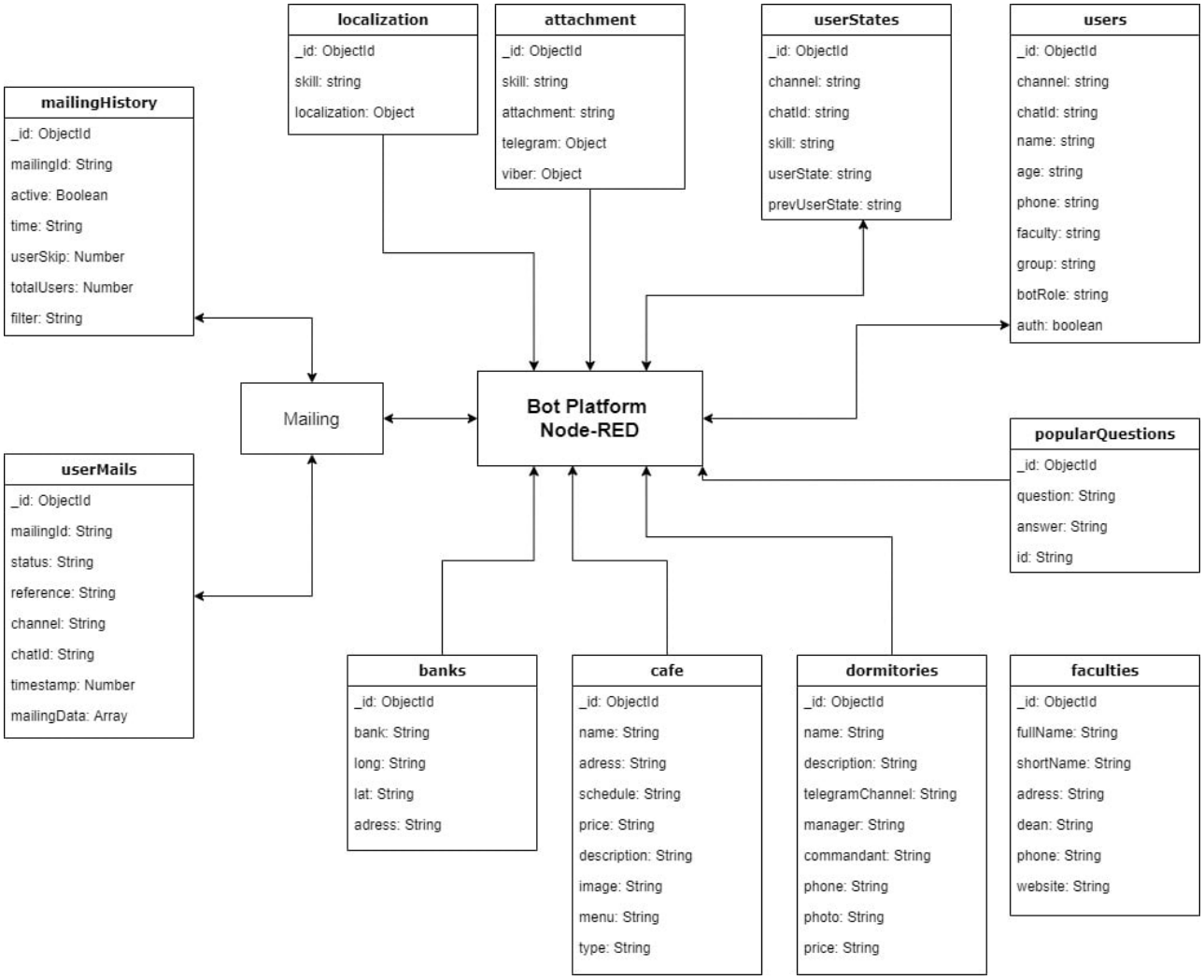
Система для комунікації у вищому навчальному закладі на основі чат-боту

Схема принципова

ІАЛЦ.467100.006 Д3

Аркушів 1

Київ – 2021 р.



					ІАЛЦ.467200.004 Д3		
Зм.	*Арк.*	*№ докум.*	*Підпис*	*Дата*			
Розробив		*Тимко А.О.*			*Система для комунікації в навчальному закладі на основі чат-боту*	*Літ.*	*Аркуш*	*Аркушів*
Перевірив		*Ткаченко В.В.*					*1*	*1*
Реценз.					*Схема принципова*		
Н. Контр.		*Сімоненко В.П.*				*НТУУ «КПІ», ФІОТ, ІВ-72*	
Затв.		*Стіренко С.Г.*					