МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«До захисту допущено»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)
« » травня 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня бакалавр

зі спеціальності 122 - Комп'ютерних наук,
освітньо-професійної програми «Інформатика»
на тему: «Інформаційне та програмне забезпечення для вивчення мов»
здобувачки групи ІН-01 Полежай Олесі Ігорівни

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Олеся ПОЛЕЖАЙ
(підпис)

Керівник,
старший викладач кафедри
ком'ютерних наук, к.т.н., доцент Борис КУЗІКОВ _____
(підпис)

Суми – 2024

Сумський державний університет
Центр заочної, дистанційної та вечірньої форм навчання
Кафедра комп'ютерних наук

«Затверджую»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавра

зі спеціальності 122 - Комп'ютерних наук, освітньо-професійної програми «Інформатика»
здобувача групи ІН-01 Полежай Олесі Ігорівни

1. Тема роботи: «Інформаційне та програмне забезпечення для вивчення мов»
затверджую наказом по СумДУ від *«22» квітня 2024 р. No 0414-VI*
2. Термін здачі здобувачем кваліфікаційної роботи *до 29 травня 2024 року*
3. Вхідні дані до кваліфікаційної роботи _____
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)
1) Аналіз існуючих аналогів. 2) Обрати технології для реалізації засобу. 3) Спроектувати систему, що розробляється. 4) Створити інформаційне наповнення уроків для вивчення іноземних мов. 5) Реалізувати та протестувати програмне забезпечення боту
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)_____
6. Консультанти до проекту (роботи), із значенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «____» _____ 20 ___ р.

Завдання прийняв до виконання _____ Керівник _____
(підпис) (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	*Аналіз існуючих аналогів*		
2	*Обрати технології для реалізації засобу*		
3	*Спроектувати систему, що розробляється*		
4	*Створити інформаційне наповнення уроків для вивчення іноземних мов*		
5	*Реалізувати та протестувати програмне забезпечення боту*		

Здобувач вищої освіти _____ Керівник _____
(підпис) (підпис)

АНОТАЦІЯ

Записка: 56 стр., 34 рис., 1 додаток, 23 використаних джерела.

Обґрунтування актуальності теми роботи – тема кваліфікаційної роботи залишається актуальною, оскільки спрямована на розв'язання практичної задачі вивчення іноземних мов шляхом розробки Telegram-бота, який поєднує в собі теоретичну інформацію та можливість перевірки знань. Дана робота пропонує ефективний інструмент для покращення мовних навичок.

Об’єкт дослідження — вивчення іноземних мов із використанням засобів ІТ.

Мета роботи — створити інформаційне та програмне забезпечення Telegram-боту для вивчення іноземних мов.

Методи дослідження — методології створення Telegram-ботів, аналіз схожих рішень.

Результати — розроблено інформаційне та програмне забезпечення для забезпечення користувачів зручною та ефективною можливістю вивчати мови через Telegram додаток. Бот надсилає інформацію з матеріалами для навчання, а також надає можливість перевірити свій рівень засвоєного матеріалу через тестування. Після завершення кожного уроку користувач отримує повідомлення з результатом та мотивуючими словами.

ТЕЛЕГРАМ БОТ, ВИВЧЕННЯ МОВ, JAVA, НАВЧАННЯ, POSTGESQL

ЗМІСТ

ВСТУП

Актуальність. У сучасному світі вивчення мов є надзвичайно актуальне, оскільки відіграє ключову роль у сприянні міжкультурного спілкування, професійного, особистого розвитку та соціальної інтеграції. Наразі багато людей вимушені жити в еміграції, тож шукають засоби для швидкого та ефективного вивчення мов, щоб легше адаптуватися до різноманітних мовних середовищ. Також зростаючий попит на міжнародні контакти та культурний обмін підкреслює необхідність ефективного володіння іноземними мовами. Тому можна зробити висновки, що програми для вивчення мов є важливим і необхідним інструментом у сучасному світі. Вони дозволяють людям набути важливих навичок, які відкривають нові можливості для особистого та професійного розвитку, а для тих хто має досвід переїзду за кордон, можуть стати незамінним інструментом для адаптації до нових умов життя.

Об’єкт дослідження. Вивчення іноземних мов із використанням засобів ІТ.

Предмет дослідження. Теоретичні та практичні аспекти процесу створення інформаційного та програмного забезпечення Telegram-ботів для вивчення іноземних мов.

Методами дослідження є методології створення Telegram-ботів, аналіз схожих рішень.

Мета та завдання роботи полягають у створенні інформаційного та програмного забезпечення Telegram-боту для вивчення іноземних мов. При цьому необхідно провести аналіз існуючих аналогів; обрати технології для реалізації засобу, спроектувати систему, що розробляється; створити інформаційне наповнення уроків для вивчення іноземних мов; реалізувати та протестувати програмне забезпечення боту.

Практична значимість. Використання Telegram як інструменту вивчення мови є інноваційним підходом, оскільки платформа дозволяє створювати інтерактивні курси, які створюють сприятливі умови для ефективного навчання.

Програмне рішення дозволить проходити користувачам короткі, але ефективні інтерактивні уроки.

Структура. Кваліфікаційна дипломна робота охоплює аналітичний огляд, формулювання завдання, вибір програмних засобів для реалізації поставленої мети, проектування як візуальної, так і програмної частини, реалізації та тестування телеграм-боту, висновків, списку використаних джерел у роботі та додаток.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз існуючих реалізацій

В цей час існує велика кількість додатків, які дозволяють вивчати іноземні мови. Вони відрізняються способом реалізації та інформацією, яка надається користувачеві. Тож проведімо аналіз аналогів, він буде включати порівняння функціональності, взаємодії з користувачами та інші фактори. Ось кілька популярних аналогів та їх характеристики:

Додатки Google PlayStore:

ReWord [1] — це додаток, який допомогає зібрати свій словник. Додаток надає як заготовлені слова, так і можна додати власні. Одна з головних переваг ReWord — персоналізовані рекомендації. Додаток аналізує прогрес користувача та його потреби, надаючи індивідуальні поради щодо того, які слова варто вивчати та в якому порядку. Додаток дозволяє вчити слова як додані користувачем, так і слова які заготовлені.

Переваги:

- відстеження прогресу;
- різноманітні методи вивчення слів;
- можна додавати власні слова;
- можна прослухати вимову слова.

Серед мінусів:

- безкоштовний функціонал обмежений;
- відсутня візуалізація слів;
- доступний лише андроїд юзерам;
- потребує значної кількість пам'яті на телефоні;
- відсутній функціонал для вивчення граматики.

Приклад роботи продемонстровано на рисунку 1.1.

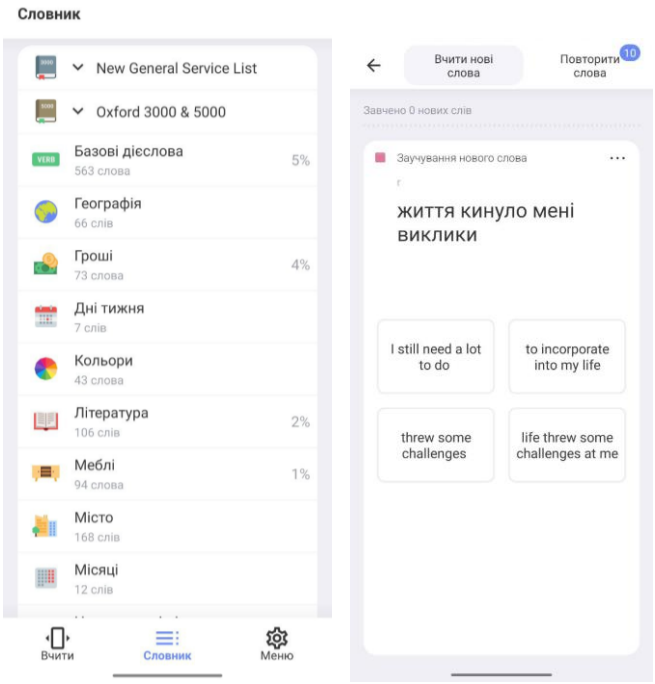

Рисунок 1.1 — Скріншот роботи з «ReWord »

Promova [2] — пропонує різноманітні методи навчання, такі як інтерактивні курси, аудіо- та відеоматеріали, а також можливості познайомитися с носіями мови та займатися у групах. Використано штучний інтелекту для створення навчання.

Переваги:

- цікаві інтерактивні уроки;
- можливість комунікації з носіями мови;
- відстеження прогресу;
- можливість вивчення різними мовами.

Серед мінусів:

- безкоштовний функціонал обмежений;
- обмежена персоналізація;
- потребує значної кількість пам'яті на телефоні;
- обмежений функціонал вивчення граматики.

Приклад роботи з наведений на рисунку 1.2.

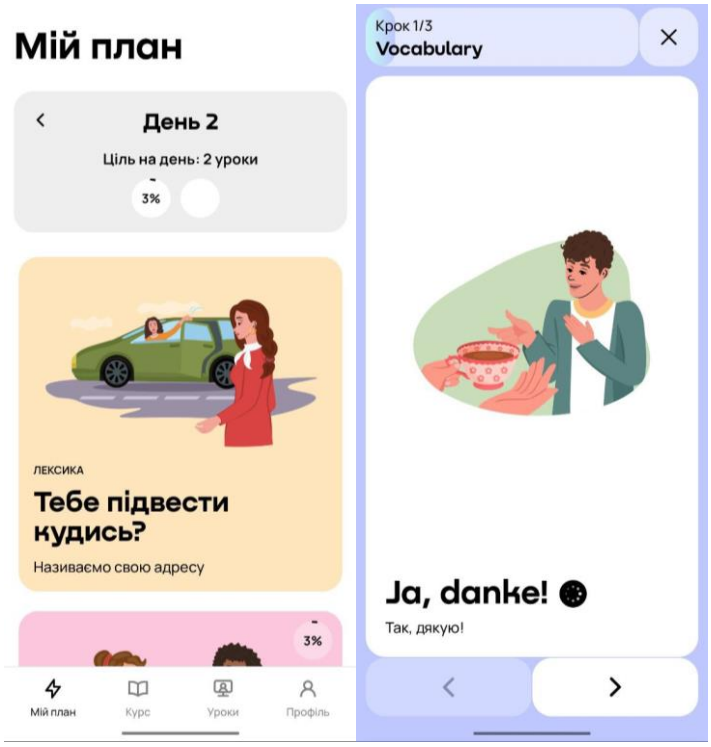

Рисунок 1.2 – Скріншот роботи з додатком «Промова»

Також розглянемо наявні чат-боти та проаналізуємо їх сильні та слабкі сторони.

GermanWithLeoBot [3] — це Telegram-бот, призначений для вивчення німецької мови. За допомогою цього бота користувачі можуть отримати доступ до різноманітних навчальних вправ, які допоможуть закріпити граматику та збільшити словниковий запас. Серед основних можливостей бота — приклади вживання слів та тематичні вправи.

Переваги:

- надає доступ до різноманітних навчальних вправ;
- поступове вивчення;
- надає матеріал за допомогою відео-, аудіо- та текстових повідомлень;
- часткове транслювання тексту англійською мовою.

Недоліки:

- не безкоштовний;
- не доступний українською мовою;
- не зручний для користувачів, які тільки починають свій шлях у вивченні мови;
- інтерфейс не завжди оптимальний для користування.

Приклад роботи з Telegram-ботом наведений на рисунку 1.3.

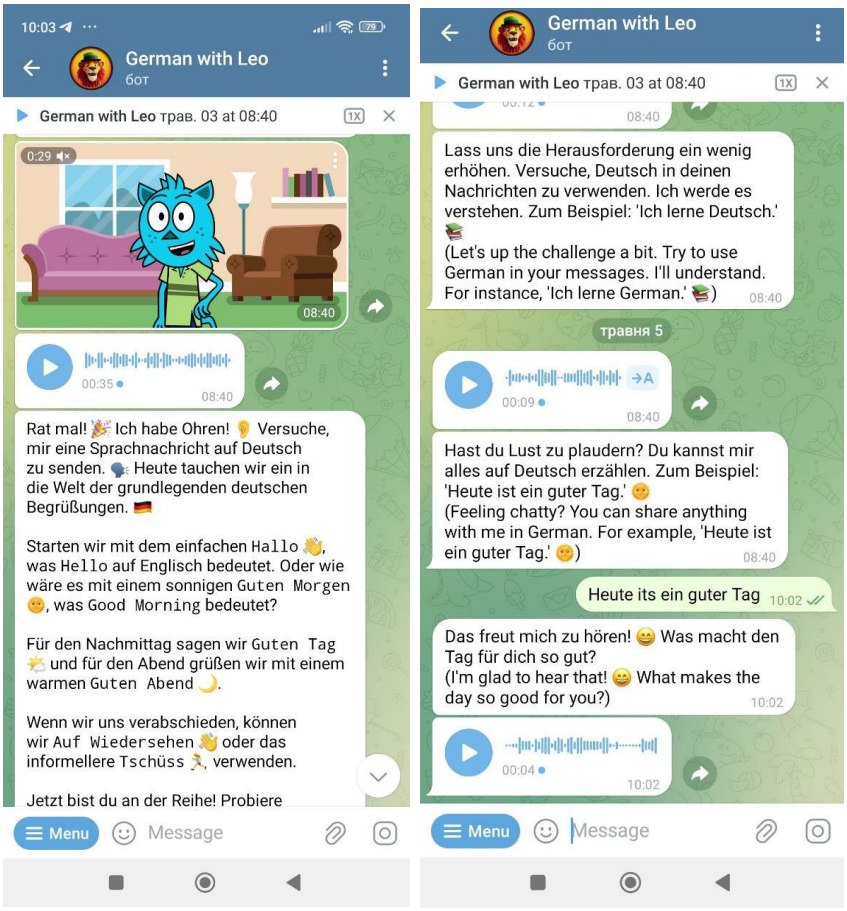

Рисунок 1.3 – Скріншот роботи з «GermanWithLeoBot»

GermanGrammarBot [4] — спеціалізується на допомозі користувачам з різними видами граматики, наприклад: відмінювання іменників, використання часів, спряження дієслів. Ключовим елементом у роботі мови є штучний інтелект. Він аналізує текст, що надіслав користувач, і виявляє можливі граматичні помилки,

надаючи користувачеві інформацію про помилки та поради.

Переваги:

- швидкі і корисні поради;
- доступний в будь-який момент;
- постійне вдосконалення завдяки застосуванню машинного навчання.

Недоліки:

- може надавати неправильні поради;
- не зручна взаємодія;
- може бути неефективний при обробці складних запитань.

Приклад взаємодії з ботом наведений на рисунку 1.4.

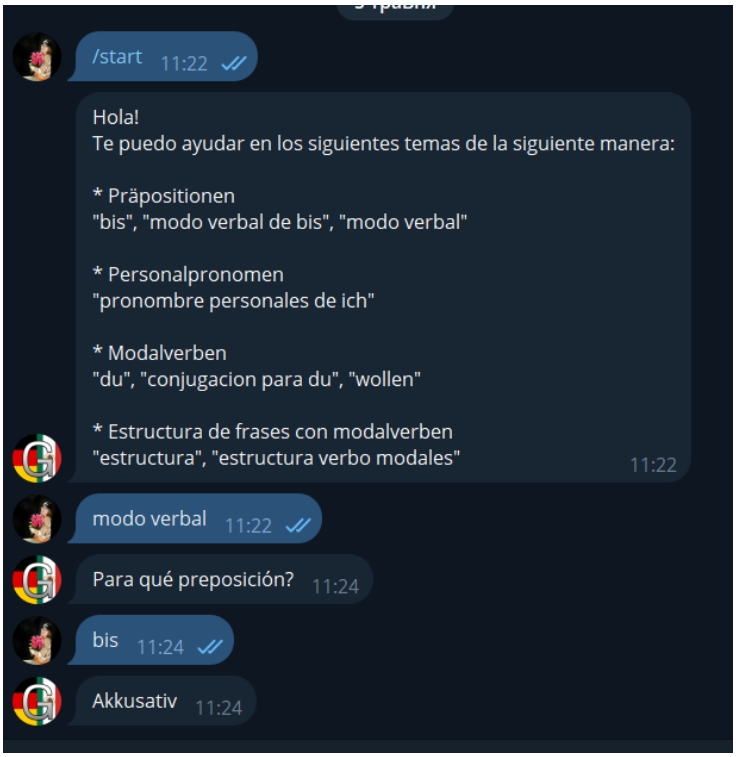

Рисунок 1.4 – Скріншот роботи з «GermanGrammarBot»

Andyrobot [5] — Telegram-бот, завдяки якому користувачі за допомогою текстового інтерфейсу можуть взаємодіяти з ботом. Він надає цікаві вправи, корисні поради та може пояснити правила з граматики. Також надає цікаві інтерактивні ігри.

Переваги:

- персоналізована допомога;
- інтерактивність;
- швидкі та корисні поради.

Недоліки:

- взаємодія тільки через текстові повідомлення ;
- для більш детального вивчення пропонує завантажити додаткові програми;
- може надавати неправильні поради;
- інтерфейс не завжди забезпечує зручність використання.

Результат користування «andyrobot» наведений на рисунку 1.5.

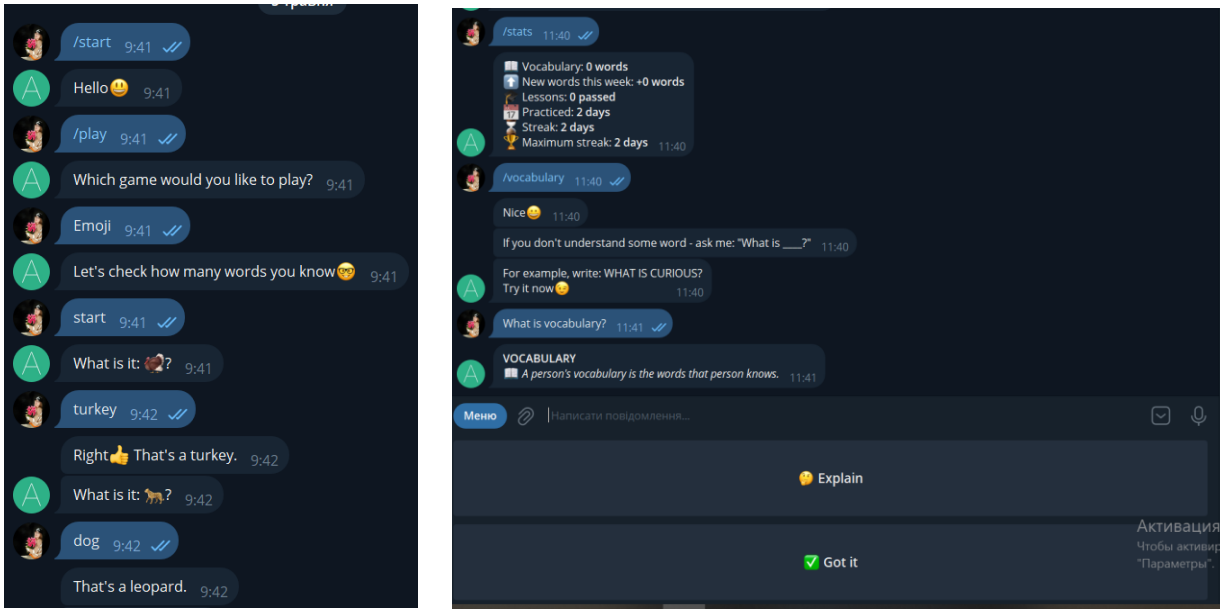

Рисунок 1.5 — Скріншот роботи з «andyrobot»

В результаті аналізу аналогів було визначено переваги та недоліки функцій додатків та ботів пов’язаних з вивченням мови. Вибрано вектор майбутнього напрямку розвитку функцій інформаційної системи, тобто система повинна надавати можливість:

- поступового вивчення;
- надавати користувачу теоретичний матеріал;
- сконцентруватися на вивченні граматики та збільшенні словникового запасу;
- відстеження прогресу;
- можливість виконувати різноманітні вправи;
- візуалізація до завдань.

Також було прийнято рішення розробляти Telegram-бот, оскільки дана технологія не потребує встановлення додатка, достатньо мати обліковий запис у месенджері Telegram. Крім того, перевага ботів Telegram полягає в їх універсальності та інтеграції з різними платформами. Вони працюють на будь-якому пристрої, який підтримує Telegram.

1.2 Постановка задачі

Після дослідження актуальності та аналізу існуючих програм, які спеціалізуються на вивченні мов було прийнято рішення розробити телеграм бот. Дане дослідження виконується в рамках в рамках кваліфікаційної роботи бакалавра за спеціальністю 122 «Комп’ютерні науки».

Проєкт який розробляється повинен реалізувати наступні функції:

- бот повинен надсилати теоретичний матеріал користувачу;
- користувач повинен мати можливість після кожної теоретичної частини проходити перевірку засвоєного матеріалу;
- користувач повинен мати можливість після завершення тесту пройти тестування ще раз;
- бот повинен відображати загальні результати тестування;
- бот має відображати коректну відповідь у випадку неправильної відповіді;
- бот повинен зберігати інформацію про користувачів.

Для реалізації поставлено задачі потрібно виконати наступні завдання:

1. провести аналіз існуючих аналогів;
2. обрати технології для реалізації засобу,
3. спроектувати систему, що розробляється;
4. створити інформаційне наповнення уроків для вивчення іноземних мов;
5. реалізувати та протестувати програмне забезпечення боту.

Практичною вагомістю даного боту є можливість проходити міні уроки у зручний час, не завантажуючи додаткових програм.

2 ВИБІР МЕТОДУ РІШЕНЬ

2.1 Вибір мови та фреймворку

Перш ніж перейти до розробки, потрібно визначитися з мовою програмування. Для цього проаналізуємо найпопулярніші мови та їх фреймворки, які широко використовуються для розробки Telegram-ботів.

Першою розглянемо одну з найпопулярніших мов програмування у світі — Python. Вона застосовується в різних сферах, таких як веброзробка, машинне навчання, обчислення, штучний інтелект, автоматизація завдань. Python порівняно з іншими мовами має велику кількість фреймворків для розроблення ботів, розглянемо декілька з них.

Telebot — це простий та зручний фреймворк для створення Telegram ботів на Python. Він дозволяє швидко розробляти прості боти. Даний інструмент має вичерпну документацію та підтримує багато функціональних можливостей, таких як реакцію на різні команди, відповіді на повідомлення, робота з клавіатурою і багато іншого, цей інструмент має популярність в розробці та має активну спільноту[6].

Python-telegram-bot — потужний фреймворк для розробки Telegram ботів на Python з широким функціоналом та активною спільнотою. Бібліотека надає простий інтерфейс та високорівневі класи для Python. Обгортка може бути безкоштовно завантажена на офіційному вебсайті GitHub [7]. Однією з переваг використання обгортки є те, що вона дозволяє багатопотоковість між викликами обробників. Таким чином, продуктивність і ефективність кодування можуть бути збільшені, оскільки можуть виконуватися два або більше виконання одночасно. Обгортка також підтримує різноманітні функції, такі як Telegram Inline кнопка, Telegram Inline клавіатура, кнопка клавіатури Telegram та багато іншого [8].

Наступною розглянемо Java це одна з найстаріших мов, але завдяки постійному вдосконаленню й адаптації до сучасних вимог ринку — одна з

найпопулярніших мов, яка застосовується майже у всіх сферах програмування, а саме в розробці ігор, вебзастосунків, корпоративних додатків. Вона дозволяє запускати програми на будь-якій платформі, що підтримує віртуальну машину Java. Мова також має велику кількість сторонніх бібліотек і фреймворків.

Одним з найкращих рішень для створення Telegram-ботів на Java є TelegramBots від іспанського розробника Рубена Бермудеса. TelegramBots спрощує розробку та управління ботами, надає можливість створювати боти з різним рівнем складності. Бібліотека підтримує різні функції Telegram API, що дозволяє створювати ботів з різним функціоналом. TelegramBots в Java використовуються для розробки різних ботів, від особистих помічників до бізнес-ботів і ботів для розваг [9].

Перейдемо до мови Go, яка є легка у використанні та фактично може допомогти створити потужну технологію, як-от канали даних і вебсервер. Також вона використовується для створення програм машинного навчання та штучного інтелекту [10]. Ця мова також має потужні бібліотеки для розроблення Telegram-ботів.

Golang bindings for the Telegram Bot API [11] – має простий і зрозумілий інтерфейс для взаємодії з API Telegram, що дозволяє швидко й ефективно створювати ботів для різноманітних завдань. Фреймворк підтримує всі основні функції Telegram Bot API, а також надає можливість асинхронних запитів, таким чином покращуючи продуктивність бота. Він активно підтримується спільнотою розробників і дозволяє вам легко розширювати функціональні можливості бота за потреби.

Наступною розглянемо мову Ruby – на сьогодні Ruby стала однією з найпопулярніших мов для веброзробки, використовуючи Ruby on Rails. Є кілька способів виконання одного й того ж завдання (принцип Ruby), що розуміється як гнучкість коду, яка є однією з основних переваг Ruby. Програміст може визначити свій синтаксис для написання коду. Це робить програмування більш цікавим, що призводить до його популярності серед програмістів [12].

Один з найкращих фреймворків для написання Telegram-ботів на Ruby є

Telegram-bot-ruby [13]. Він має простий і зрозумілий синтаксис, який дозволяє швидко розвивати функціональність вашого бота. Завдяки гнучкості Ruby реалізація різноманітних роботів стає зручнішою. Однак його слабкі сторони включають обмежену швидкість виконання та можливі проблеми з масштабованістю у великих проєктах. Крім того, бібліотека може не підтримувати всі функції Telegram API, що може призвести до обмеження функціональності бота.

Для кращого розуміння створимо таблицю, де опишемо переваги кожного рішення (див. табл. 1.1).

Таблиця 1.1 − Порівняння мов програмування для створення Telegram-ботів

Фреймворк	**Мова програмування**	**Основні переваги**	**Основні недоліки**
Telebot	Python	Простота, хороша документація, широкий набір функцій	Обмежена функціональність для складних проектів
python-telegram-bot	Python	Добра структура, підтримка асинхронності, активний розвиток	Складність архітектури, час для вивчення
TelegramBots	Java	Підходить для великих проектів, багатопоточність	Складність для вивчення, менша гнучкість
Golang bindings for the Telegram Bot API	Go	Висока продуктивність, паралельне виконання	Складність архітектури
telegram-bot-ruby	Ruby	Зрозумілий синтаксис, гнучкість мови, активна спільнота	Менша швидкість, складність у маштабуванні

Після аналізу рішень було обрано TelegramBots з огляду на стабільність, зручність при розробці, гарну документацію, велику спільноту користувачів та функціональність.

2.2 Структурно-функціональне моделювання

Для моделювання процесів використаємо методологію IDEF0. Основна перевага даної методології в здатності відображення взаємодії між різними компонентами системи. Також дана методологія стандартизована з погляду на синтаксис та семантику, тому моделі добре визначені, добре структуровані, прості для розуміння, модифікації та використання та можуть бути розширені до будь-якої глибини деталей. Модель IDEF0 також є гнучкою та масштабованою для адаптації до різноманітних ситуацій та умов [14].

Після постановки задачі та вибрання мови реалізації було визначено перелік даних для побудови контекстної діаграми А-0. Цей список містить усі ключові елементи, необхідні для чіткого розуміння та моделювання взаємодії між системою та зовнішнім середовищем, що дозволяє точніше представити структуру та функціональність системи.

Вхідними даними є інформація про користувача та команда від користувача.

Вихідними даними є результат тестування, матеріали уроку та оновлення даних про користувача у БД.

Елементами управління було обрано: вимоги до бота, документація PostgeSQL, документація Telegram Bot API.

Механізмами є Java, Telegram API, PostgresSQL, користувач.

На рисунку 2.1 зображена контекстна діаграма.

Рисунок 2.1 - Контекстна IDEF0 діаграма

Так як було збудовано контекстну діаграму, яка містить лише одну основну функцію на наступному етапі необхідно її деталізувати. Для цього використаємо декомпозицію – процес розбиття системи на складові частини, визначення функцій і визначення того, як вони взаємодіють для досягнення цілей системи. Метою функціональної декомпозиції є спрощення складних систем на менші, більш керовані та зрозумілі частини. Функціональна декомпозиція полегшує проектування, аналіз і впровадження складних систем [15].

Під час поділу контекстної діаграми на компоненти, досліджуваний процес було виділено на п’ять етапів:

- користувач натискає «/start»;
- ознайомлення з теоретичною частиною уроку;
- проходження тестування;
- повторне проходження тестування;
- перехід до наступного уроку.

На рисунку 2.2 представлено декомпозицію IDEF0-діаграми.

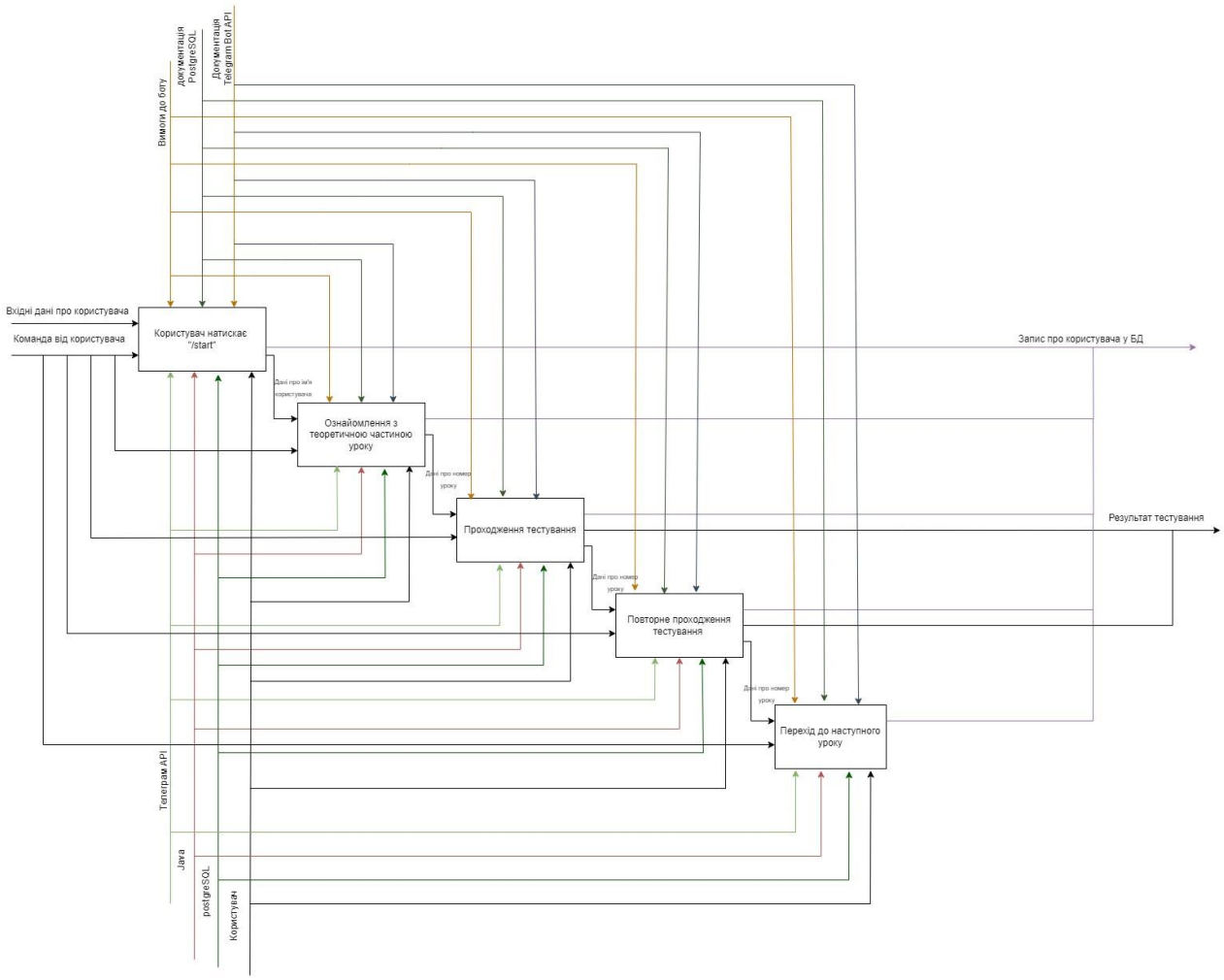

Рисунок 2.2 - Діаграма декомпозиції IDEF0

2.3 Діаграма варіантів використання

В теперішній час існує широкий спектр методів і засобів для відображення та моделювання бізнес-процесів. Для моделювання розробленого додатка використано діаграму варіантів використання. Ця діаграма візуально зображає різноманітні сценарії взаємодії між акторами (користувачами) та випадками використання (прецедентами). Крім того, вона дозволяє описати функціональні аспекти системи, такі як бізнес-логіка, та зрозуміти, як система повинна працювати

у різних ситуаціях [16].

Діаграм варіантів відображає, що система повинна робити не вказуючи на конкретні метод для використання.

Для реалізації діаграми виділені наступні актори: «користувач», «адміністратор» та «БД».

Діаграма повинна включати наступні варіанти використання:

- команда старт;
- почати урок;
- перейти до наступного уроку;
- пройти тестування;
- пройти тестування ще раз;
- результати тестування;
- авторизація у телеграм боті для адміністратора;
- додавання нових матеріалів для навчання та редагування існуючих.

Діаграма варіантів використання (рис. 2.3) наведена нижче:

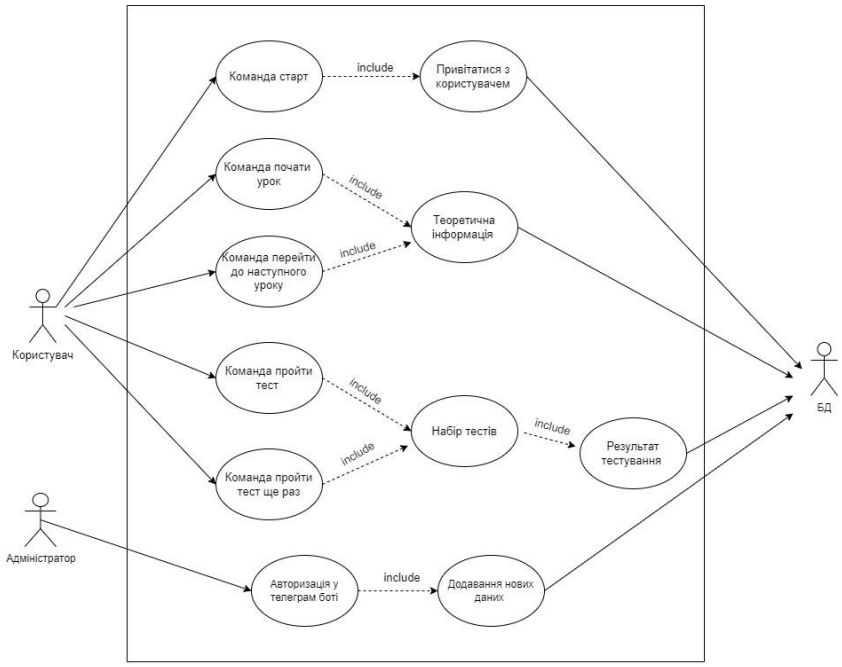

Рисунок 2.3 - Діаграма варіантів використання

2.4 Проектування моделі бази даних

Сучасні інформаційні системи обертаються навколо баз даних і систем керування ними. Ці системи відіграють вирішальну роль в організації та зберіганні даних. Нині на світовому ринку доступна велика різноманітність СУБД, які обслуговують реляційні бази даних. Це стосується як комерційних, так і вільно розповсюджуваних систем.

Для розробки Telegram-бота нам знадобитися система управління базами даних, яка допоможе зберігати, організовувати та звертатися до даних, що використовуються ботом. У якості СУБД для виконання бакалаврської роботи було обрано PostgreSQL. PostgreSQL це об'єктно-реляційна система управління базами даних (СУБД), що вільно поширюється. Основними можливостями, що надаються PostgreSQL є надійність, багатоверсійність, цілісність даних, відкритість кодів, продуктивність, розширюваність [17].

PostgreSQL досить популярна СУБД, але, важливіше те, що вона більш гнучка, ніж її аналоги. Дана СУБД дозволяє налаштовувати все, що потрібно для конкретної задачі, це така собі «зброя», якою потрібно вміти користуватися [18]. Попри те, що система на ринку понад 37 років, вона постійно вдосконалюється і на сьогодні за версією DB-Engines PostgreSQL названа системою управління базами даних року [19].

Для проєктування бази даних побудуємо ER-модель (Entity-relationship model або Entity-relationship diagram) – модель сутності-зв’язку надає схематичне представлення та метод проєктування бази даних шляхом визначення її структури перед збереженням інформації. З іншого боку, реляційна модель надає точне математичне визначення для зберігання інформації у формі таблиць (зв’язків), пов’язаних між собою зовнішніми ключами, базова структура яких визначається моделлю ER [20].

Для розробки чат системи, створено 3 сутності, а саме: таблиця, яка міститиме інформацію про користувачів, а також таблиці які зберігатимуть матеріали уроків

та дані для перевірки засвоєних знань (рис 2.4).

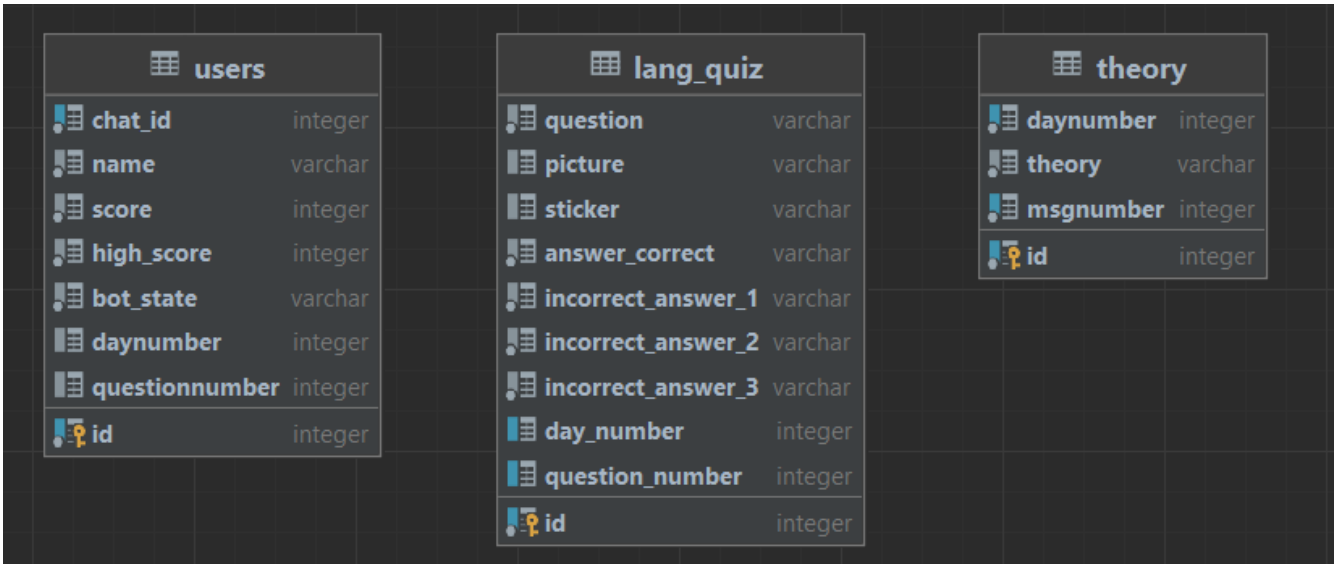

Рисунок 2.4 - ERD для ІС

Перейдемо до реалізації структури БД. Для цього представимо імена необхідних таблиць, атрибутів, типів, їх призначення та обмеження (див. табл. 2.1).

Таблиця 2.1 – Опис змінних у структурі БД

Таблиця	Поле	Зміст	Тип	Ключі / обмеження
Users	id	Ідентифікатор користувача	Integer	Primary Key
	chart_id	Ідентифікатор в телеграмі	Integer	Unique, not null
	name	Ім’я користувача	Varchar	not null
	score	Результат проходження тесту	Integer	not null (default 0)
	high_score	Найкращий результат	Integer	not null (default 0)
	Day_number	Номер уроку	Integer	(default 0)
	Question_number	Номер тесту, який проходить користувач	Integer	(default 0)

Продовження таблиці 2.1

Lang_quiz	id	Ідентифікатор питання	Integer	Primary Key
	Picture	Посилання на картинку для завдання	Varchar	
	sticker	Ідентифікатор стікера	Varchar	
	answer_correct	Правильна відповідь	Varchar	not null
	incorrect_answer_1	Неправильна відповідь	Varchar	not null
	incorrect_answer_2	Неправильна відповідь	Varchar	not null
	incorrect_answer_3	Неправильна відповідь	Varchar	not null
	day_number	Номер уроку	Integer	Not null unique (day_number, question_number)
	question_number	Номер питання	Integer	Not null unique (day_number, question_number)
Theory	id	Ідентифікатор теорії	Integer	Primary Key
	Theory	Текст теорії	varchar	Not null
	dayNumber	Номер уроку	integer	Not null, unique (dayNumber, msgNumber)
	msgNumber	Номер повідомлення	integer	Not null, unique (dayNumber, msgNumber)

На рис. 2.5 – 2.6 відображені фрагменти таблиць з інформацією про теоретичний матеріал для уроків та питання для перевірки засвоєного матеріалу.

	daynumber	theory	msgnumber
7	1	👻 Hallo! Ich freue mich dich hier zu sehen😍 Сьогодні я поясню тобі тему Komposita	1
8	1	📌Komposita є складними іменниками у німецькій мові, … Наприклад: die Küche + der Schrank:	2
9	2	WEG vs LOS	1
0	2	🔲 Прислівник LOS має таке ж значення, як і <… Ich muss leider los. (На жаль, мені потрібно йти.)…	2

Рисунок 2.5 – Теоретична частина уроків

	id	question	sticker	answer_correct	option1	option2	option3	day_number	question_number
1	148	Яке з наступн…	CAACAgIAAxkB…	Wir müssen jetzt l…	Mein Hund …	Mein Schlüs…	Der Kuchen…	2	1
2	149	Заповніть про…	CAACAgIAAxkB…	los/weg, los/weg	los, weg	weg, los	os, los	2	2
3	150	Вставте пропу…	CAACAgIAAxkB…	weg, weg	weg, los	los, weg	los, los	2	3
4	151	Утвори правил…	CAACAgIAAxkB…	der Wasserhahn	das Wasser…	der Hahnwas…	das Hahnwa…	1	1
5	152	PS. Що означа…	CAACAgIAAxkB…	der Löwenzahn	das Löwenz…	der Löwezahn	die Löwenz…	1	2
6	153	Подумай над з…	CAACAgIAAxkB…	die Sonnenbrille	die Sonneb…	die Brilles…	die Brille…	1	3
7	154	PS. Що означа…	CAACAgIAAxkB…	die Handschuhe	die Schuhe…	die Händesc…	die Schuhe…	1	4
8	155	Утвори правил…	CAACAgIAAxkB…	der Regenbogen	der Bogenr…	der Bogereg…	der Regebo…	1	5

Рисунок 2.6 – Дані для тестування

3 РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 Реалізація взаємодії з базою даних

Для зручного додавання нових уроків було впроваджено новий функціонал в боті, який забезпечує зручну взаємодію з базою даних. Тепер, щоб додати новий матеріал, достатньо ввести команду «/add_materials», і бот автоматично відправить вам приклад повідомлення, яке потрібно надіслати адміністратору для додавання даних (рис. 3.1).

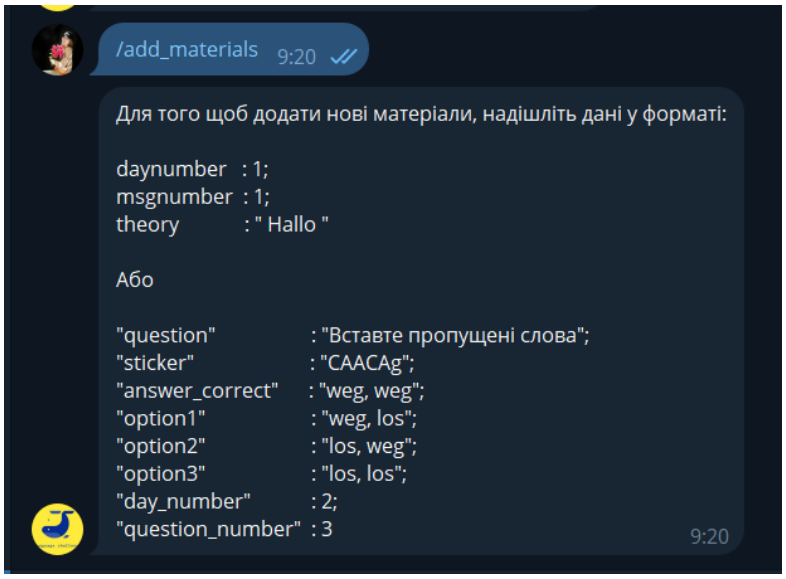

Рисунок 3.1 – Загальна інформація

На рисунку 3.2 показано приклад відповіді бота в разі правильного надсилання повідомлення з теоретичним результатом.

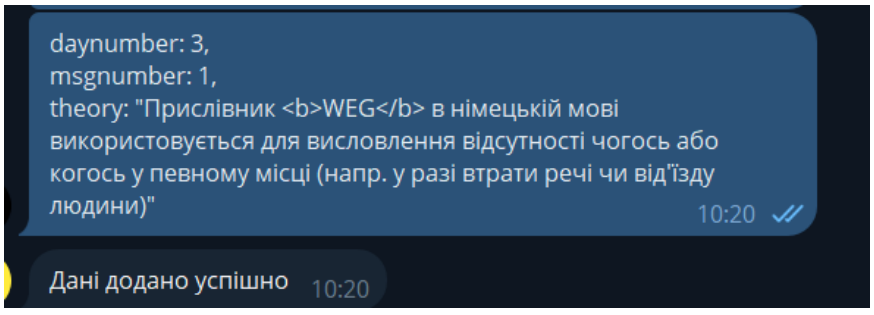

Рисунок 3.2 – Успішне додавання нового матеріалу

Для того щоб додати питання, яке містить стікер, потрібно мати його ідентифікатор. Для отримання інформації було використано бот «LeadConverterToolkitBot»[21]. Для одержання необхідно надіслати боту стікер, і отримати ідентифікатор. Приклад роботи можна переглянути на рисунку 3.3.

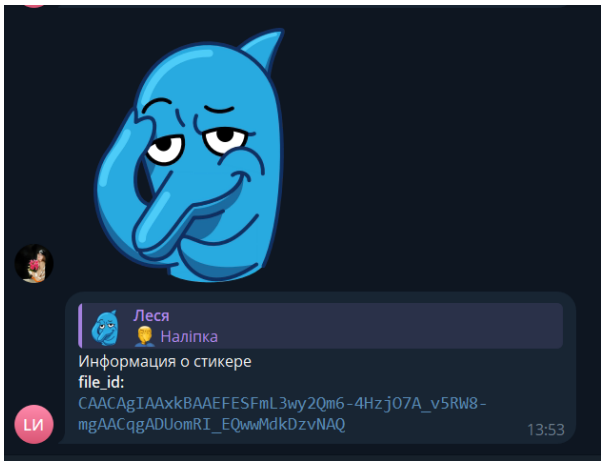

Рисунок 3.3 – Приклад отримання коду стікера

На рисунку 3.4 показано приклад успішного додавання питання для перевірки засвоєного матеріалу. А на рисунку 3.5 показано обробку помилки, якщо адміністратор намагається додати матеріал, який вже був доданий раніше.

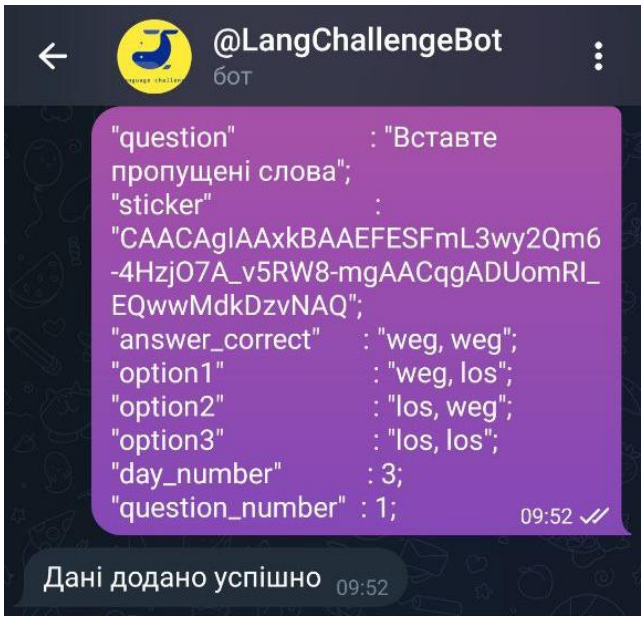

Рисунок 3.4 – Успішне додавання питання

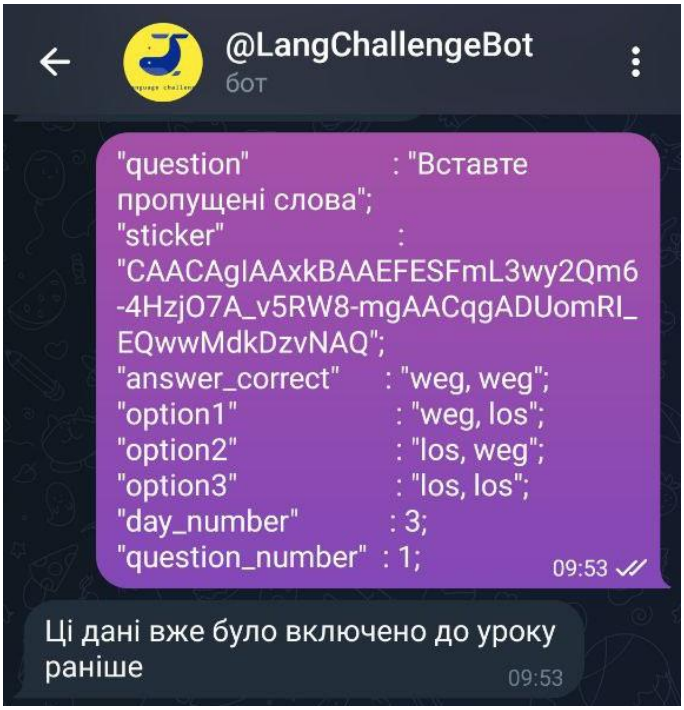

Рисунок 3.5 – Оброблення помилки

3.2 Реєстрація Telegram бота

Перед тим як перейти до розробки Telegram-бота зареєструємо його у самій програмі обміну повідомленнями використавши офіційний Telegram-бот «BotFather» [22]. Після виконання даної процедури отримуємо унікальний токен, який забезпечую взаємодію між ботом та Telegram API. На рисунку 3.6 представлена початкова сторінка бота, яка містить загальну інформацію та посилання на більш детальну інструкцію, що надає детальні пояснення для налаштування бота.

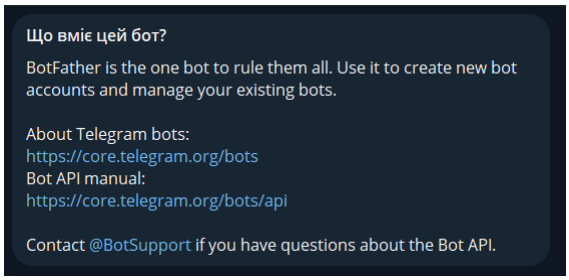

Рисунок 3.6 – Загальна інформація

Після початку роботи з’являється меню, яке відображаю команди для роботи з ботом (рис 3.7).

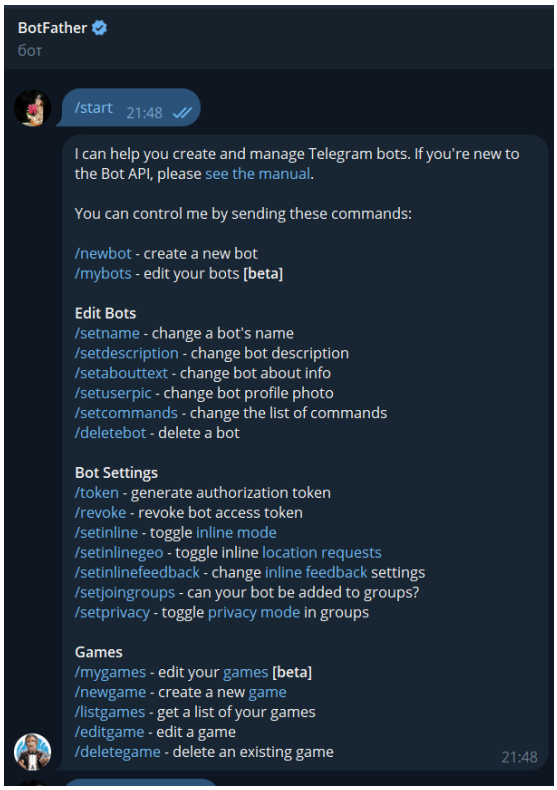

Рисунок 3.7 – Меню для керування BotFather

Для подальшої роботи натискаємо на команду /newbot. Та обирає ім’я для бота (рис 3.8).

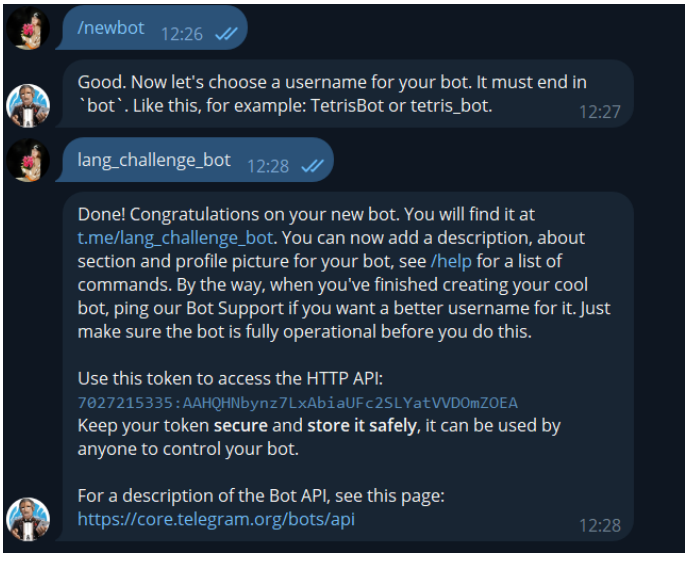

Рисунок 3.8 - Налаштування для Telegram-бота

Далі за допомогою додатку «logomaster.ai» [23] згенеровано логотип та встановлено його за допомогою команди «/setuserpic» (рис 3.9).

Рисунок 3.9 - Встановлення логотипу

За допомогою команди «/setdescription» додаємо опис до Telegram-бота. (рис. 3.10).

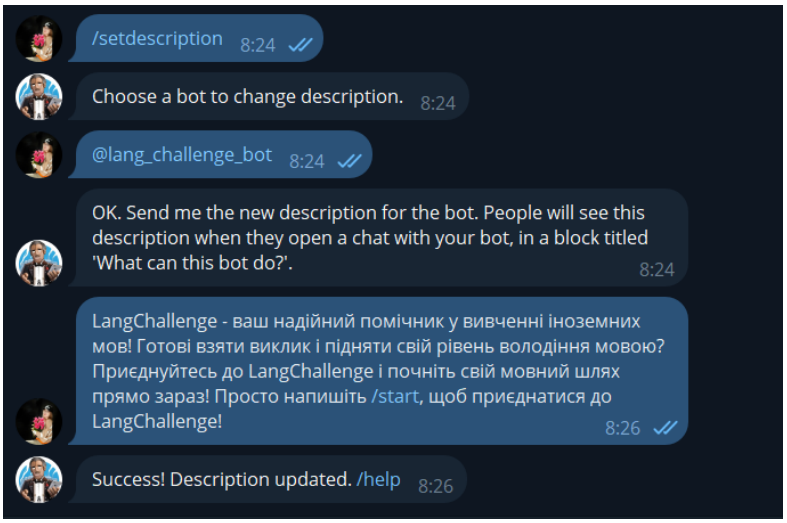

Рисунок 3.10 - Встановлення опису

Після реєстрації бота можна переходити до його розробки.

3.3 Реалізація Telegram бота

Для демонстрації результатів роботи нижче наведено скріншоти із основними можливостями Telegram-бота.

Під час першого відкриття Telegram-бота «LangChallengeBot», користувач може ознайомитися з загальну інформацію про можливості бота (рис 3.11).

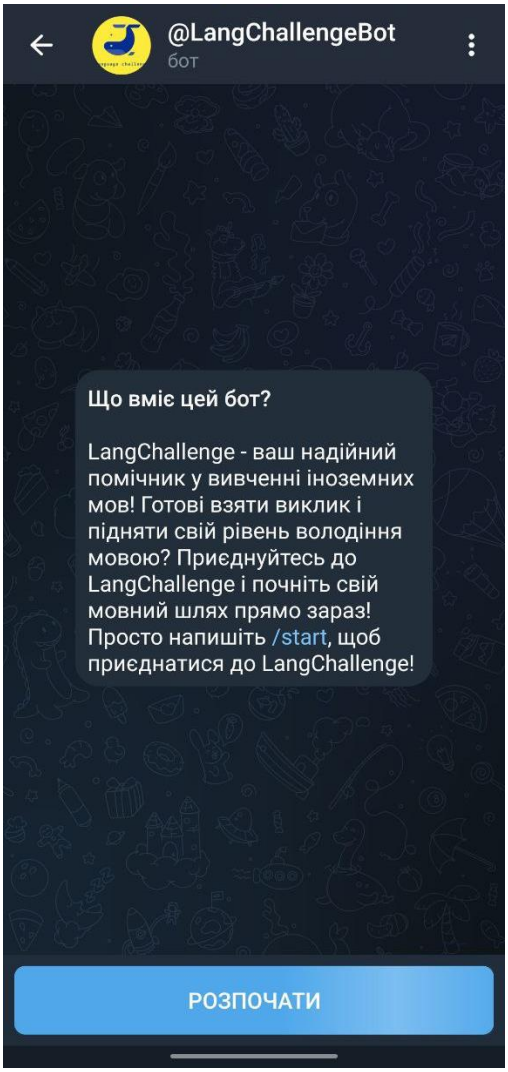

Рисунок 3.11 – Відображення опису Telegram-бота.

Для того щоб почати користуватися ботом, потрібно ввести команду «/start» або натиснути кнопку «Розпочати». Після цього дані про користувача автоматично заносяться до бази даних, а йому відправляється привітальне повідомлення. Переглянути отриману інформацію можна на рисунку 3.12.

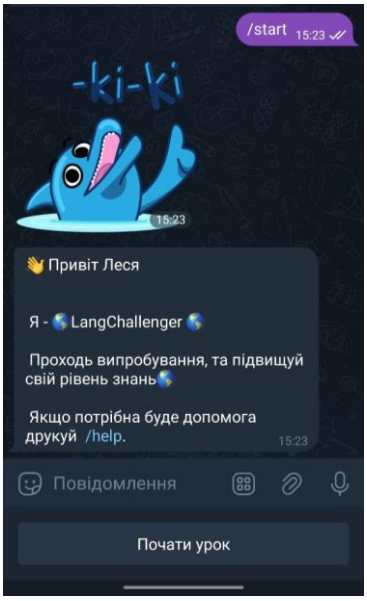

Рисунок 3.12 – Відображення привітального повідомлення

Для того щоб почати навчання, необхідно натиснути кнопку «Почати урок». Бот відправить інформацію з матеріалами для навчання, вона буде подана поетапно, з затримкою декілька секунд. Це необхідно для того, щоб користувач міг поступово ознайомитися з кожним повідомленням перед отриманням наступного (рис 3.13).

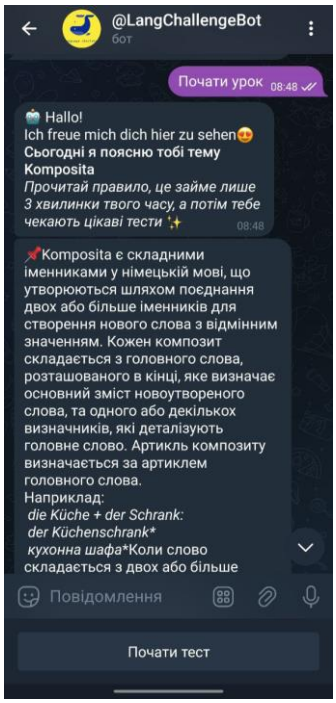

Рисунок 3.13 – Теоретична частина уроку

Після вивчення теорії користувач може перевірити свій рівень засвоєного матеріалу, пройшовши тестування. Кожен урок має своє тестування, і кількість тестів може відрізнятися. Кожен тест складається зі стікера або картинки, запитання та чотирьох варіантів відповідей. Приклад тесту ви можете переглянути на рисунку 3.14.

Рисунок 3.14 – Приклад тесту

Після відповіді користувача на запитання, якщо відповідь правильна, бот надішле повідомлення, підтверджуюче це. Крім того, залежно від характеру завдання, може бути надіслане візуальне представлення правильної відповіді (рис. 3.15).

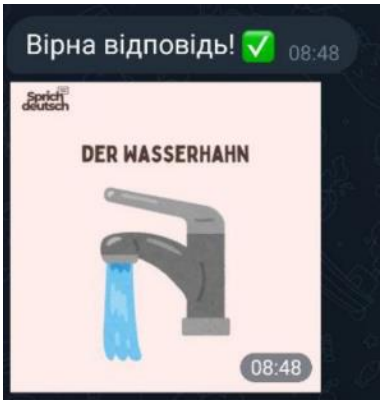

Рисунок 3.15. – Відображення повідомлення, якщо відповідь правильна

Якщо ж натиснуто неправильну відповідь, то буде надіслано повідомлення про помилку разом з вірною відповіддю. Крім того, в залежності від типу завдання, користувач може отримати візуальне відображення правильної відповіді (рис 3.16).

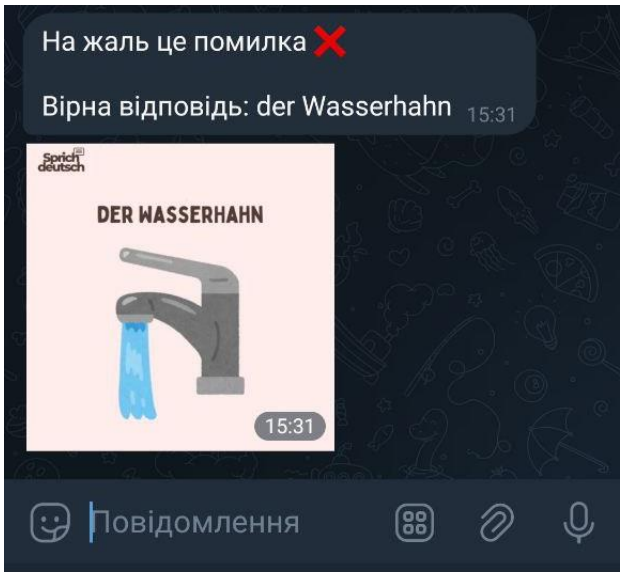

Рисунок 3.16 – Відображення повідомлення, якщо відповідь неправильна

Після проходження всіх тестів призначених для певного уроку. Користувач отримує текстове повідомлення з результатом, мотивуюче повідомлення та стікер. Якщо відсоток правильних відповідей менший або дорівнює 35%, то бот надішле повідомлення, зображене на рисунку 3.17.

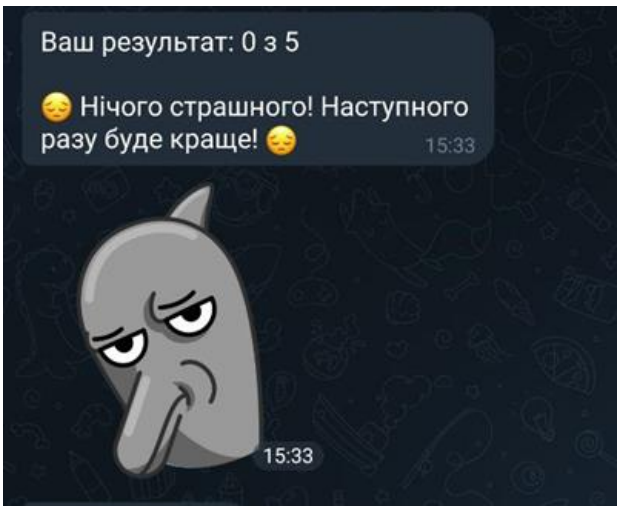

Рисунок 3.17 – Результат тестування

Якщо ж відсоток правильних відповідей від 35 до 55 включно, користувач отримає інше повідомлення, а також відповідне зображення (рис 3.18).

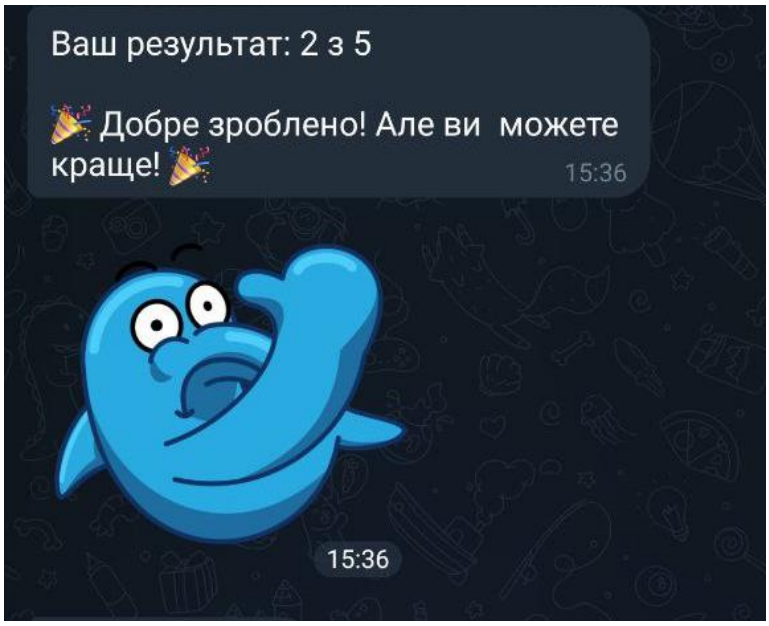

Рисунок 3.18 – Результат тестування

Якщо ж діапазоні 55% - 80%, користувач отримає своє сповіщення та відповідне зображення (рис. 3.19).

Рисунок 3.19 – Результат тестування

У випадку, коли відсоток правильних відповідей перевищує 80%, користувач отримає відповідне повідомлення та зображення (рис. 3.20).

Після відображення результат тестування, користувачу надається меню яке містить дві кнопки: «Пройти тест ще раз» і «Перейти до наступного уроку». Меню можна переглянути на рисунку 3.21.

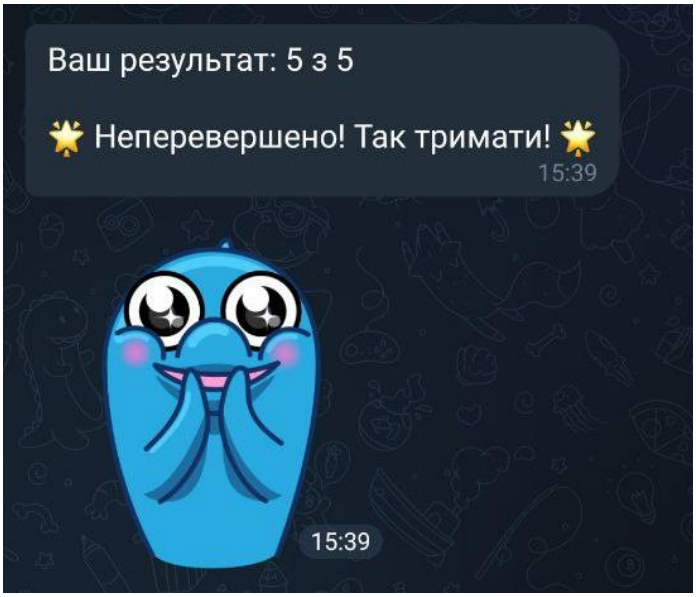

Рисунок 3.20 – Результат тестування

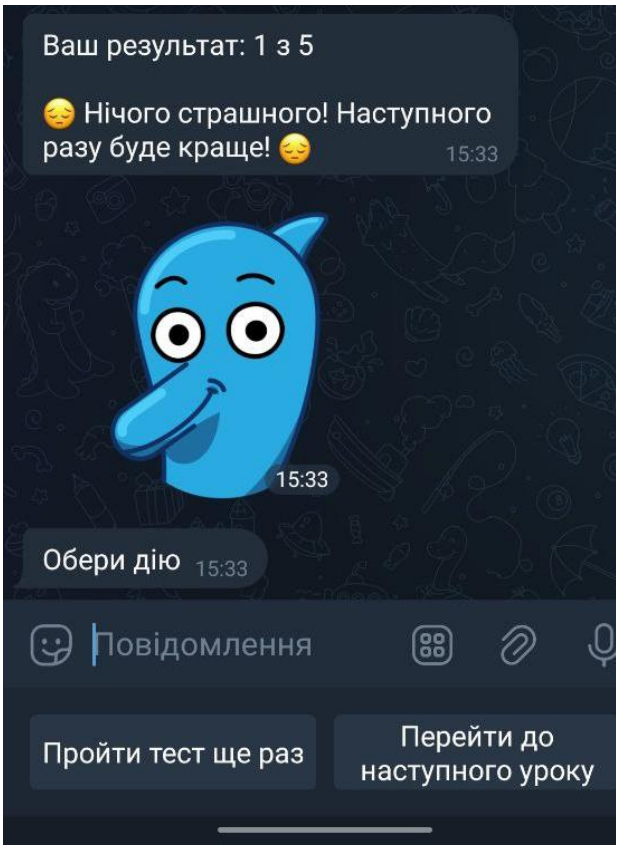

Рисунок 3.21 – Відображення кнопок

При натисканні на кнопку «Пройти тест ще раз» відображатимуться тести в тій самій послідовності, але варіанти відповідей будуть відображено в іншому порядку.

Якщо ж користувач натисне кнопку «Перейти до наступного уроку», то відбудеться, той самий сценарій, що й після натискання на кнопку «Почати урок». Користувач отримає повідомлення з новим теоретичним матеріалом (рис. 3.22.)

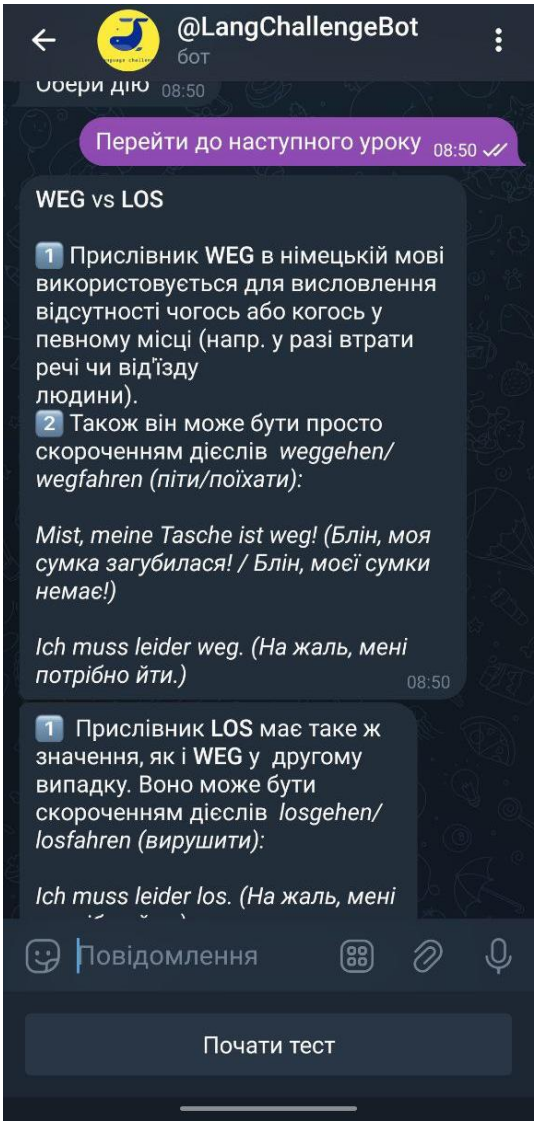

Рисунок 3.22 – Відображення теоретичного матеріалу

Якщо ж нові уроки знаходяться на стадії розробки, то після натискання кнопки «Перейти до наступного уроку». Користувач отримає повідомлення відображене на рисунку 3.23

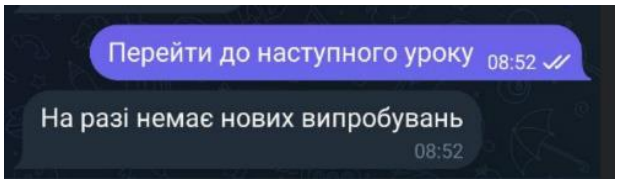

Рисунок 3.23 – Повідомлення про відсутність нових уроків.

ВИСНОВКИ

В результаті бакалаврської роботи було розроблено «LangChallengeBot», який забезпечує користувачів зручною та ефективною можливістю вивчати мови через Telegram додаток. Бот надсилає інформацію з матеріалами для навчання, а також надає можливість перевірити свій рівень засвоєного матеріалу через тестування. Кожен урок супроводжується зручним тестуванням з використанням стікерів або картинок. Після завершення кожного уроку користувач отримує повідомлення з результатом та мотивуючими словами. Також для адміністратора системи було розроблено функціонал, за допомогою якого можна додавати нові матеріали для уроків та перевірки знань.

У ході виконання кваліфікаційної роботи було виконано такі завдання:

1. Проведено глибокий аналіз сучасних сервісів, які використовуються для вивчення мов, визначено їх переваги та недоліки, на основі яких було визначено вектор розробки Telegram-бота.
2. Проаналізовані сучасні інструменти для розробки Telegram-ботів та створено порівняльну таблицю (див. таблицю 1.1). На основі аналізу мова Java та фреймворк TelegramBots обрані для реалізації проекту.
3. Для моделювання було використано наступні методології:
   - Для моделювання процесів використано IDEF0. Було визначено входи, виходи, елементи управління та механізми (рис 2.1). Для досягнення більшої деталізації застосовано декомпозицію, під час якої було виділено п'ять основних етапів роботи системи (рис 2.2).
   - Для моделювання взаємодії користувачів з системою та внутрішніх процесів самої системи використано діаграму варіантів використання. Визначено, що технологічне рішення має трьох основних акторів: користувача, адміністратора та базу даних. Результат моделювання відображений на рисунку 2.3.

- Для управління базою даною було обрано PostgreSQL. Крім того, для побудови бази даних, що є основою для розробки чат-системи, використано ER-модель. Ця модель включає три основні сутності: таблицю користувачів, таблицю матеріалів уроків та таблицю перевірки знань (рис 2.4).

4. Створено інформаційне наповнення уроків для ефективного вивчення німецької мови.
5. Розроблено ефективний бот, який допоможе користувачам у вивченні мови через інтерактивне середовище месенджера. На додочу роботу системи було ретельно перевірено.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. ReWord [Електронний ресурс].– Режим доступу до ресурсу: https://reword.app/uk/en
2. Promova [Електронний ресурс].– Режим доступу до ресурсу: https://promova.com/uk
3. German With Leo Bot [Електронний ресурс].– Режим доступу до ресурсу: https://t.me/GermanWithLeoBot
4. German Grammar Bot [Електронний ресурс].– Режим доступу до ресурсу: https://t.me/GermanGrammarBot
5. Andy English Bot [Електронний ресурс].– Режим доступу до ресурсу: https://t.me/AndyRobot
6. pyTelegramBotAPI. *PyPI*. URL: https://pypi.org/project/pyTelegramBotAPI/ (date of access: 08.05.2024).
7. GitHub - python-telegram-bot/python-telegram-bot: We have made you a wrapper you can't refuse. *GitHub*. URL: https://github.com/python-telegram-bot/python-telegram-bot (date of access: 08.05.2024).
8. *Journal of Islamic, Social, Economics and Development*. URL: http://www.jised.com/PDF/JISED-2021-36-05-88.pdf (дата звернення: 08.05.2024).
9. GitHub - rubenlagus/TelegramBots: Java library to create bots using Telegram Bots API. *GitHub*. URL: https://github.com/rubenlagus/TelegramBots (date of access: 08.05.2024).
10. undefined. The Best Programming Languages to Learn in 2024. *City Lit | Adult Education | Courses Online & in London*. URL: https://www.citylit.ac.uk/blog/best-programming-languages-learn-this-year#go (date of access: 08.05.2024).

11. GitHub - go-telegram-bot-api/telegram-bot-api: Golang bindings for the Telegram Bot API. *GitHub*. URL: https://github.com/go-telegram-bot-api/telegram-bot-api (date of access: 08.05.2024).
12. Conceptual Modeling / ed. by G. Goos et al. Berlin, Heidelberg : Springer Berlin Heidelberg, 1999. URL: https://doi.org/10.1007/3-540-48854-5 (date of access: 08.05.2024).
13. GitHub - atipugin/telegram-bot-ruby: Ruby wrapper for Telegram's Bot API. *GitHub*. URL: https://github.com/atipugin/telegram-bot-ruby (date of access: 08.05.2024).
14. Automation of strategy using IDEF0 – A proof of concept. *Operations Research Perspectives*. 2015. Vol. 2. P. 106–113.
15. What Is Functional Decomposition? | Baeldung on Computer Science. *Baeldung on Computer Science*. URL: https://www.baeldung.com/cs/functional-decomposition (date of access: 08.05.2024).
16. УКРАЇНСЬКІ СТУДІЇ В ЄВРОПЕЙСЬКОМУ КОНТЕКСТІ : зб. наук. пр. / ред. Ш. Т.М. 7-ме вид. Київ : Держ. наук. установа «Ін-т модернізації змісту освіти», 2023. 462 с.
17. Clark L. PostgreSQL named DBMS of the year by DB-Engines. *The Register: Enterprise Technology News and Analysis*. URL: https://www.theregister.com/2024/01/03/opensource_postgresql_named_dbms_of/ (date of access: 06.05.2024).
18. *DSpace at West Ukrainian National University: Головна сторінка*. URL: http://dspace.wunu.edu.ua/bitstream/316497/31976/1/Мєлєнтьєва_Лебедєва.pdf (дата звернення: 06.05.2024).

19. *DSpace Repository :: Electronic Kyiv-Mohyla Academy Institutional Repository*. URL: https://ekmair.ukma.edu.ua/server/api/core/bitstreams/5e5cd9d2-2da2-46d6-a621-27fe586361e6/content (дата звернення: 06.05.2024).

20. Hytowitz A. N. Review of using the Dyop optotype for acuity and refractions per the article: https://www.sciencedirect.com/science/article/pii/S1888429622000656. *Journal of Optometry*. 2023. URL: https://doi.org/10.1016/j.optom.2022.12.002 (date of access: 06.05.2024).

21. Lead Converter Toolkit Bot [Електронний ресурс].– Режим доступу до ресурсу: https://t.me/LeadConverterToolkitBot

22. Bot Father [Електронний ресурс].– Режим доступу до ресурсу: https://t.me/BotFather

23. logomaster [Електронний ресурс].– Режим доступу до ресурсу: https://app.logomaster.ai/ru/proposals

ДОДАТОК

```
@Component
public class QuizHandler extends BotCommand {

  private final UserController userController;
  private final QuestionController questionController;

  public QuizHandler(@Value(MessageTest.StartCompetition) String commandIdentifier,
      @Value("") String description,
      UserController userController, QuestionController questionController) {
   super(commandIdentifier, description);
   this.userController = userController;
   this.questionController = questionController;
  }

  @Override
  public void execute(AbsSender absSender, User user, Chat chat, String[] strings) {

   com.example.langchallenge2.bot.model.User user1 = new com.example.langchallenge2.bot.model.User(
       user.getId(), chat.getFirstName());

   int day_number = userController.getDayByChatId(chat.getId());
   userController.incrementQuestionNumber(user1);
   int question_number = userController.getQuestionNumberByChartIc(chat.getId());
   int count_question = questionController.getCountQuestionInDay(day_number);

   if (day_number == 1 && question_number == 1) {
    SendMessage sendMessage = new SendMessage(chat.getId().toString(),
        MessageTest.MessageQuestionForFirstDays);
    sendMessage.enableHtml(true);
    try {
     absSender.execute(sendMessage);
     Thread.sleep(2000);
    } catch (TelegramApiException | InterruptedException e) {
     throw new RuntimeException(e);
    }
   }

   if (question_number <= count_question) {
    Question question = questionController.getQuestionInDay(day_number, question_number);
    try {
     if (question.getSticker() != null) {
      InputFile sticker = new InputFile(question.getSticker());
      absSender.execute(new SendSticker(chat.getId().toString(), sticker));
     } else if (question.getPicture() != null) {

      InputFile photo = new InputFile(new File(question.getPicture()));
      SendPhoto sendPhoto = new SendPhoto(chat.getId().toString(), photo);
      absSender.execute(sendPhoto);
     }
     absSender.execute(sendQuestion(chat.getId().toString(), question));
```

```
    } catch (TelegramApiException e) {
      e.printStackTrace();
    }
  } else {
    try {
      printTheTestResult(absSender, user, chat);
    } catch (TelegramApiException e) {
      throw new RuntimeException(e);
    }
  }
}

public SendMessage sendQuestion(String chatId, Question question) {
  SendMessage message = new SendMessage();
  message.setChatId(chatId);
  message.setText(question.getQuestion());

  InlineKeyboardMarkup markupInline = new InlineKeyboardMarkup();

  List<List<InlineKeyboardButton>> rowsInline = new ArrayList<>();

  List<InlineKeyboardButton> rowInline1 = new ArrayList<>();

  List<String> options = new ArrayList<>();
  options.add(question.getCorrectAnswer());
  options.add(question.getOptionOne());
  options.add(question.getOptionTwo());
  options.add(question.getOptionThree());
  Collections.shuffle(options);

  InlineKeyboardButton inlineKeyboardButton1 = new InlineKeyboardButton();
  inlineKeyboardButton1.setText(options.get(0));
  inlineKeyboardButton1.setCallbackData(
     Objects.equals(options.get(0), question.getCorrectAnswer()) ? "/test_correct"
        : "/test_incorrect1"
  );
  rowInline1.add(inlineKeyboardButton1);

  List<InlineKeyboardButton> rowInline2 = new ArrayList<>();

  InlineKeyboardButton inlineKeyboardButton2 = new InlineKeyboardButton();
  inlineKeyboardButton2.setText(options.get(1));
  inlineKeyboardButton2.setCallbackData(
     Objects.equals(options.get(1), question.getCorrectAnswer()) ? "/test_correct"
        : "/test_incorrect2"
  );
  rowInline2.add(inlineKeyboardButton2);

  List<InlineKeyboardButton> rowInline3 = new ArrayList<>();

  InlineKeyboardButton inlineKeyboardButton3 = new InlineKeyboardButton();
  inlineKeyboardButton3.setText(options.get(2));
```

```
 inlineKeyboardButton3.setCallbackData(
    Objects.equals(options.get(2), question.getCorrectAnswer()) ? "/test_correct"
       : "/test_incorrect3"
 );
 rowInline3.add(inlineKeyboardButton3);

 List<InlineKeyboardButton> rowInline4 = new ArrayList<>();

 InlineKeyboardButton inlineKeyboardButton4 = new InlineKeyboardButton();
 inlineKeyboardButton4.setText(options.get(3));
 inlineKeyboardButton4.setCallbackData(
    Objects.equals(options.get(3), question.getCorrectAnswer()) ? "/test_correct"
       : "/test_incorrect4"
 );
 rowInline4.add(inlineKeyboardButton4);

 rowsInline.add(rowInline1);
 rowsInline.add(rowInline2);
 rowsInline.add(rowInline3);
 rowsInline.add(rowInline4);

 markupInline.setKeyboard(rowsInline);
 message.setReplyMarkup(markupInline);

 return message;
}

public void correctAnswer(AbsSender absSender, User user, Chat chat, String[] strings)
   throws TelegramApiException, InterruptedException {
 com.example.langchallenge2.bot.model.User user1 = new com.example.langchallenge2.bot.model.User(
    user.getId(), chat.getFirstName());

 userController.incrementScore(user1);

 int day_number = userController.getDayByChatId(chat.getId());
 if (day_number == 1) {
  sendPhotoAnswer(absSender, chat);
 }
 execute(absSender, user, chat, strings);
}

public void incorrectAnswer(AbsSender absSender, User user, Chat chat, String[] strings)
   throws TelegramApiException, InterruptedException {

 int day_number = userController.getDayByChatId(chat.getId());
 int question_number = userController.getQuestionNumberByChartIc(chat.getId());

 Question question = questionController.getQuestionInDay(day_number, question_number);

 String correctAnswer =
    MessageTest.IncorrectAnswer + MessageTest.IncorrectAnswer2 + question.getCorrectAnswer();
```

```
  SendMessage sendMessage = new SendMessage(chat.getId().toString(),
      correctAnswer);
  absSender.execute(sendMessage);

  if (day_number == 1) {
    sendPhotoAnswer(absSender, chat);
  }

  execute(absSender, user, chat, strings);
}

public void printTheTestResult(AbsSender absSender, User user, Chat chat)
    throws TelegramApiException {

  com.example.langchallenge2.bot.model.User user1 = new com.example.langchallenge2.bot.model.User(
      user.getId(), chat.getFirstName());

  int score = userController.getScoreByChatId(user1.getChatId());
  int lesson = userController.getDayByChatId(chat.getId());
  int countQuestion = questionController.getCountQuestionInDay(lesson);

  userController.resetScore(user1);
  userController.resetQuestionNumber(user1);

  String resultText = MessageTest.ResultMessage + score + " з " + countQuestion;
  float accuracy = (float) score / countQuestion;
  InputFile sticker;
  if (accuracy < 0.35) {
    sticker = new InputFile(MessageTest.StickerBadResult);
    resultText += MessageTest.MessageBadResult;
  } else if (accuracy < 0.55) {
    sticker = new InputFile(MessageTest.StickerGoodResult);
    resultText += MessageTest.MessageGoodResult;
  } else if (accuracy < 0.8) {
    sticker = new InputFile(MessageTest.StickerVeryGoodResult);
    resultText += MessageTest.MessageVeryGoodResult;
  } else {
    sticker = new InputFile(MessageTest.StickerAmazingResult);
    resultText += MessageTest.MessageAmazingResult;
  }

  absSender.execute(new SendMessage(chat.getId().toString(), resultText));
  absSender.execute(new SendSticker(chat.getId().toString(), sticker));

  navigateNextOrRetry(absSender, user, chat);

}

public void navigateNextOrRetry(AbsSender absSender, User user, Chat chat)
    throws TelegramApiException {

  SendMessage messageTheory;
```

```
    messageTheory = new SendMessage(chat.getId().toString(),
        MessageTest.MessageChooseButton);
    ReplyKeyboardMarkup keyboardMarkup = new ReplyKeyboardMarkup();
    List<KeyboardRow> keyboard = new ArrayList<>();
    KeyboardRow row = new KeyboardRow();
    KeyboardButton button = new KeyboardButton(MessageTest.MessageTryAgain);
    KeyboardButton button2 = new KeyboardButton(MessageTest.MessageStartNewLesson);
    keyboardMarkup.setResizeKeyboard(true);
    row.add(button);
    row.add(button2);
    keyboard.add(row);
    keyboardMarkup.setKeyboard(List.of(keyboard.toArray(new KeyboardRow[keyboard.size()])));
    //keyboardMarkup.setOneTimeKeyboard(true);
    messageTheory.setReplyMarkup(keyboardMarkup);
    absSender.execute(messageTheory);
  }

  public void resetTest(AbsSender absSender, User user, Chat chat, String[] strings) {
    com.example.langchallenge2.bot.model.User user1 = new com.example.langchallenge2.bot.model.User(
        user.getId(), chat.getFirstName());
    userController.resetScore(user1);
    userController.resetQuestionNumber(user1);
    execute(absSender, user, chat, strings);
  }

  public void sendPhotoAnswer(AbsSender absSender, Chat chat) throws TelegramApiException {

    int question_number = userController.getQuestionNumberByChartIc(chat.getId());

    SendMessage sendMessage = new SendMessage(chat.getId().toString(), "Наступне питання");
    if (question_number == 1) {
      InputFile sticker = new InputFile(
          "CAACAgIAAxkBAAEFDtRmLqkF0jd-MwrvA2pcHs7O3BpdOAAC_0sAAoLncElPfNFAxETvszQE");
      absSender.execute(new SendSticker(chat.getId().toString(), sticker));
    } else if (question_number == 2) {
      InputFile sticker = new InputFile(
          "CAACAgIAAxkBAAEFDtZmLqkGCSqX2HwLImaUF1n1acZcaQACF1EAAtx0cEn-
seKWchHvlTQE");
      absSender.execute(new SendSticker(chat.getId().toString(), sticker));
    } else if (question_number == 3) {
      InputFile sticker = new InputFile(

"CAACAgIAAxkBAAEFDthmLqkIOO4_JCnNYichu6e1RKeWjwACOEgAAsEjeEnUacqGVRqbjzQE");
      absSender.execute(new SendSticker(chat.getId().toString(), sticker));
    } else if (question_number == 4) {
      InputFile sticker = new InputFile(

"CAACAgIAAxkBAAEFDtpmLqkJtsVXSFGBAaDBB884v_UwLwACLUYAArGLeEmSKCGexAd8FTQE");
      absSender.execute(new SendSticker(chat.getId().toString(), sticker));
    } else if (question_number == 5) {
      InputFile sticker = new InputFile(
          "CAACAgIAAxkBAAEFDuRmLqkRVIXSFEsRP4VzolQMoVSDagAC4U4AAi8ucElJc05gJAaItTQE");
```

```
      absSender.execute(new SendSticker(chat.getId().toString(), sticker));
    }

    try {
      Thread.sleep(2000);
      if (question_number != 5) {
        absSender.execute(sendMessage);
      }
    } catch (InterruptedException e) {
      throw new RuntimeException(e);
    }
  }
}
@Component
public class StartLessonHandler extends BotCommand {

  private final UserController userController;
  private final QuestionController questionController;
  private final TheoryController theoryController;

  public StartLessonHandler(@Value(MessageTest.StartLesson) String commandIdentifier,
      @Value("") String description,
      UserController userController, QuestionController questionController,
      TheoryController theoryController) {
    super(commandIdentifier, description);
    this.userController = userController;
    this.questionController = questionController;
    this.theoryController = theoryController;
  }

  @Override
  public void execute(AbsSender absSender, User user, Chat chat, String[] strings) {
    SendMessage messageTheory;

    com.example.langchallenge2.bot.model.User user1 = new com.example.langchallenge2.bot.model.User(
        user.getId(), chat.getFirstName());

    int lesson = userController.getDayByChatId(chat.getId());

    if (questionController.getCountQuestionInDay(lesson + 1) != 0) {
      userController.incrementDay(user1);
      lesson = userController.getDayByChatId(chat.getId());
      System.out.println(lesson);
      int i = 1;
      System.out.println(theoryController.checkData(lesson, i));
      while (theoryController.checkData(lesson, i) == 1) {
        Theory theory = theoryController.getQuestionInDay(lesson, i);
        SendMessage sendMessage = new SendMessage(chat.getId().toString(),
            theory.getTheory());
        sendMessage.enableHtml(true);
        try {
          absSender.execute(sendMessage);
```

```
            Thread.sleep(1000);
           } catch (TelegramApiException | InterruptedException e) {
            throw new RuntimeException(e);
           }
           i++;
          }
          messageTheory = new SendMessage(chat.getId().toString(),
              MessageTest.MessageStartTest);
          ReplyKeyboardMarkup keyboardMarkup = new ReplyKeyboardMarkup();
          List<KeyboardRow> keyboard = new ArrayList<>();
          KeyboardRow row = new KeyboardRow();
          KeyboardButton button = new KeyboardButton(MessageTest.MessageButtonStartTest);
          keyboardMarkup.setResizeKeyboard(true);
          row.add(button);
          keyboard.add(row);
          keyboardMarkup.setKeyboard(keyboard);
          keyboardMarkup.setOneTimeKeyboard(true);
          messageTheory.setReplyMarkup(keyboardMarkup);
        } else {
          messageTheory = new SendMessage(chat.getId().toString(),
              MessageTest.MessageNotAvailableLesson);
        }
        try {
          messageTheory.enableHtml(true);
          absSender.execute(messageTheory);
        } catch (TelegramApiException e) {
          throw new RuntimeException(e);
        }
      }
    }

    @RestController
    public class QuestionController {
      private final QuestionRepository questionRepository;

      public QuestionController(QuestionRepository questionRepository) {
        this.questionRepository = questionRepository;
      }

      public Question getRandomQuestion(){
       return questionRepository.getRandomQuestion();
      }

      public Integer getCountQuestionInDay(int day){
        return questionRepository.getCountQuestionInDay(day);
      }
     public Question getQuestionInDay(int day, int numberQuestion){
        return questionRepository.getQuestionInDay(day, numberQuestion);
      }

      public void setQuestion(Question question){
        questionRepository.save(question);
```

```
  }

  public int checkQuestion (int day, int numberQuestion){
    return questionRepository.checkQuestion( day, numberQuestion);
  }
}

@RestController
public class TheoryController {

  private final TheoryRepository theoryRepository;

  public TheoryController(TheoryRepository theoryRepository) {
    this.theoryRepository = theoryRepository;
  }

  public void setTheory(Integer dayNumber, Integer msgNumber, String theory){
    theoryRepository.setTheory(dayNumber, theory, msgNumber);
  }
  public int checkData(Integer dayNumber, Integer msgNumber){
    return theoryRepository.checkData(dayNumber, msgNumber);
  }

  public Theory getQuestionInDay (Integer dayNumber, Integer msgNumber){
    return  theoryRepository.getQuestionInDay(dayNumber, msgNumber);
  }
}

@RestController
public class UserController {

  private final UserRepository userRepository;

  public UserController(UserRepository userRepository) {
    this.userRepository = userRepository;
  }

  public void addOneEmployee(User user) {
    Optional<User> existingUser = userRepository.findByChatId(user.getChatId());
    existingUser.ifPresent(userRepository::delete);
    userRepository.save(user);
  }

  public void incrementScore(User user) {
    Optional<User> existingUserOptional = userRepository.findByChatId(user.getChatId());
    if (existingUserOptional.isPresent()) {
      User existingUser = existingUserOptional.get();
      existingUser.setScore(existingUser.getScore() + 1);
      userRepository.save(existingUser);
    } else {
      userRepository.save(user);
    }
```

```
}

public void resetScore(User user) {
  Optional<User> existingUserOptional = userRepository.findByChatId(user.getChatId());
  if (existingUserOptional.isPresent()) {
    User existingUser = existingUserOptional.get();
    existingUser.setScore(0);
    userRepository.save(existingUser);
  } else {
    userRepository.save(user);
  }
}

public void resetQuestionNumber(User user) {
  Optional<User> existingUserOptional = userRepository.findByChatId(user.getChatId());
  if (existingUserOptional.isPresent()) {
    User existingUser = existingUserOptional.get();
    existingUser.setQuestionNumber(0);
    userRepository.save(existingUser);
  } else {
    userRepository.save(user);
  }
}

public void incrementDay(User user) {
  Optional<User> existingUserOptional = userRepository.findByChatId(user.getChatId());
  if (existingUserOptional.isPresent()) {
    User existingUser = existingUserOptional.get();
    existingUser.setDayNumber(existingUser.getDayNumber() + 1);
    existingUser.setQuestionNumber(0);
    existingUser.setScore(0);
    userRepository.save(existingUser);
  } else {
    userRepository.save(user);
  }
}

public void incrementQuestionNumber(User user) {
  Optional<User> existingUserOptional = userRepository.findByChatId(user.getChatId());
  if (existingUserOptional.isPresent()) {
    User existingUser = existingUserOptional.get();
    existingUser.setQuestionNumber(existingUser.getQuestionNumber() + 1);
    userRepository.save(existingUser);
  } else {
    userRepository.save(user);
  }
}

public int getQuestionNumberByChartIc(long chatId) {
  Optional<User> userOptional = userRepository.findByChatId(chatId);
  if (userOptional.isPresent()) {
    User user = userOptional.get();
```

```
      return user.getQuestionNumber();
    } else {
      return -1;
    }
  }

  public int getScoreByChatId(long chatId) {
    Optional<User> userOptional = userRepository.findByChatId(chatId);
    if (userOptional.isPresent()) {
      User user = userOptional.get();
      return user.getScore();
    } else {
      return -1;
    }
  }

  public int getDayByChatId(long chatId) {
    Optional<User> userOptional = userRepository.findByChatId(chatId);
    if (userOptional.isPresent()) {
      User user = userOptional.get();
      return user.getDayNumber();
    } else {
      return -1;
    }
  }
}
public class MessageTest {

  public static final String StickerWelcome = "CAACAgIAAxkBAAEEzJFmICZkvauEj6gZiDHm3a0-
olqosQACuAADUomRI0m50GWI4c3YNAQ";
  public static final String WelcomeMessage1 = "\uD83D\uDC4B Привіт ";
  public static final String WelcomeMessage2 = "\n Я - \uD83C\uDF0ELangChallenger \uD83C\uDF0E\n" +
      "\n Проходь випробування, та підвищуй свій рівень знань\uD83C\uDF0E\n" +
      "\n Якщо потрібна буде допомога друкуй  /help.";
  public static final String StartCompetition = "/start_test";
  public static final String StartLesson = "/start_lesson";
  public static final String CorrectAnswer = "Вірна відповідь! ✅";
  public static final String IncorrectAnswer = "На жаль це помилка ❌ \n\n";
  public static final String IncorrectAnswer2 = "Вірна відповідь: ";
  public static final String ResultMessage = "Ваш результат: ";
  public static final String StickerAmazingResult = "CAACAgIAAxkBAAEE-
3ZmKlllhAN1UXiytRkbWQmzbjPZoQACsQADUomRIz3MSWUEgKd-NAQ";
  public static final String MessageAmazingResult = "\n\n🌟 Неперевершено! Так тримати! 🌟";
  public static final String StickerVeryGoodResult = "CAACAgIAAxkBAAEE-2hmKllC-kZ-
QNEQi3IviNrWaMD3qAACogADUomRI4SKHQ65_D0mNAQ";
  public static final String MessageVeryGoodResult = "\n\n👍 Дуже добре! Продовжуйте в тому ж дусі! 👍";
  public static final String StickerGoodResult = "CAACAgIAAxkBAAEE-
2pmKllFtC0dcH5kz0wjbMTudzq18wACpAADUomRI68ST7EFwyc9NAQ";
  public static final String MessageGoodResult = "\n\n🎉 Добре зроблено! Але ви  можете краще! 🎉";
  public static final String StickerBadResult =
"CAACAgIAAxkBAAEFARxmK47oW1CnCIjCnUCoTpKZDOieZQACqQADUomRI9qLNpy7V6HXNAQ";
```

```
  public static final String MessageBadResult = "\n\n😔 Нічого страшного! Наступного разу буде краще!
😔";
  public static final String MessageButtonStartTest = "Почати тест";
  public static final String MessageStartTest = "Для початку тестування натисніть на кнопку";
  public static final String MessageButtonStartLesson = "Почати урок";
  public static final String MessageNotAvailableLesson = "На разі немає нових випробувань";
  public static final String MessageChooseButton = "Обери дію";
  public static final String MessageTryAgain = "Пройти тест ще раз";
  public static final String MessageStartNewLesson = "Перейти до наступного уроку";
  public static final String MessageQuestionForFirstDays =
     "Утвори правильний <b>Komposita</b> з двох іменників. \n"
        + "PS. Що означає це слово ти відразу побачиш на картинці. <i>Viel Spaß</i> \uD83D\uDE09";
  public static final String MessageAddTheoryError = "Ці дані вже було включено до уроку раніше";
  public static final String MessageAddTheoryError2 = "Дані введено не вірно.";
  public static final String MessageAddTheorySuccess = "Дані додано успішно";
  public static final String MessageAddMaterials =
     "Для того щоб додати нові матеріали, надішліть дані у форматі: \n\n" +
        "daynumber   : 1; \n" +
        "msgnumber  : 1; \n" +
        "theory            : \" Hallo \" \n " +
        "\nАбо \n\n" +
        "\"question\"                : \"Вставте пропущені слова\";\n" +
        "\"sticker\"                  : \"CAACAg\";\n" +
        "\"answer_correct\"      : \"weg, weg\";\n" +
        "\"option1\"                  : \"weg, los\";\n" +
        "\"option2\"                  : \"los, weg\";\n" +
        "\"option3\"                  : \"los, los\";\n" +
        "\"day_number\"           : 2;\n" +
        "\"question_number\"  : 3";
}

@Entity
@Table(name = "theory")
@Getter
public class Theory {

  @Id
  @GeneratedValue(strategy = GenerationType.AUTO)
  private Integer theoryID;

  @Column(name = "theory")
  private String theory;

  @Column(name = "dayNumber")
  private Integer dayNumber;

  @Column(name = "msgNumber")
  private Integer msgNumber;

}

public interface TheoryRepository extends JpaRepository<Theory, Integer> {
```

```
  @Modifying
  @Transactional
  @Query(nativeQuery = true, value =
     "INSERT INTO theory (dayNumber, theory, msgNumber) values ( :dayNumber, :theory, :msgNumber)")
  void setTheory(@Param("dayNumber") int dayNumber, @Param("theory") String theory,
     @Param("msgNumber") int msgNumber);

  @Query(nativeQuery = true, value =
     "SELECT CASE WHEN EXISTS (SELECT 1 FROM theory WHERE dayNumber = :dayNumber AND
msgNumber = :msgNumber) THEN 1 ELSE 0 END")
  int checkData(@Param("dayNumber") int dayNumber, @Param("msgNumber") int msgNumber);

  @Query(nativeQuery = true, value = "SELECT * from theory where daynumber = :dayNumber AND
msgNumber = :msgNumber")
  Theory getQuestionInDay(@Param("dayNumber") int dayNumber, @Param("msgNumber") int msgNumber);
}
public interface QuestionRepository extends JpaRepository<Question, Integer> {

  @Query(nativeQuery = true, value = "SELECT *  FROM lang_quiz ORDER BY random() LIMIT 1")
  Question getRandomQuestion();

  @Query(nativeQuery = true, value = "SELECT count(*) from lang_quiz where day_number = :dayNumber")
  Integer getCountQuestionInDay(@Param("dayNumber") int dayNumber);

  @Query(nativeQuery = true, value = "SELECT * from lang_quiz where day_number = :dayNumber AND
question_number = :questionNumber")
  Question getQuestionInDay(@Param("dayNumber") int dayNumber,
     @Param("questionNumber") int questionNumber);

  @Query(nativeQuery = true, value = "SELECT count(*) from lang_quiz where day_number = :dayNumber
AND question_number = :questionNumber")
  int checkQuestion(@Param("dayNumber") int dayNumber, @Param("questionNumber") int questionNumber);
}
@Component
public class MyTelegramBot extends TelegramLongPollingCommandBot {

  private final String username;
  @Autowired
  private QuizHandler quizHandler;
  @Autowired
  private StartLessonHandler startLessonHandler;

  @Autowired
  private AddLearningMaterialsHandler addLearningMaterialsHandler;


  public MyTelegramBot(@Value("${bot.token}") String botToken,
     @Value("${bot.username}") String username) {
    super(botToken);
    this.username = username;
  }
```

```
  @Override
  public String getBotUsername() {
    return username;
  }

  @Override
  public void processNonCommandUpdate(Update update) {
    if (update.hasMessage() && update.getMessage().hasText()) {
      String messageText = update.getMessage().getText();
      if(messageText.equals(MessageTest.MessageButtonStartTest)){
        quizHandler.execute(this, update.getMessage().getFrom() ,update.getMessage().getChat(), null);
      }
      if(messageText.equals(MessageTest.MessageButtonStartLesson)){
       startLessonHandler.execute(this, update.getMessage().getFrom() ,update.getMessage().getChat(), null);
      }
      if(messageText.equals(MessageTest.MessageTryAgain)){
        quizHandler.resetTest(this, update.getMessage().getFrom() ,update.getMessage().getChat(), null);
      }
      if(messageText.equals(MessageTest.MessageStartNewLesson)){
        startLessonHandler.execute(this, update.getMessage().getFrom() ,update.getMessage().getChat(), null);
      }
      if(messageText.contains("theory")){
        try {
          addLearningMaterialsHandler.addTheory(this, update.getMessage().getFrom()
,update.getMessage().getChat(), messageText);
        } catch (TelegramApiException e) {
          throw new RuntimeException(e);
        }
      }
      if(messageText.contains("option1")){
        try {
          addLearningMaterialsHandler.addTest(this, update.getMessage().getFrom()
,update.getMessage().getChat(), messageText);
        } catch (TelegramApiException e) {
          throw new RuntimeException(e);
        }
      }
    }
    if (update.hasCallbackQuery()) {
      CallbackQuery callbackQuery = update.getCallbackQuery();
      String callbackData = callbackQuery.getData();
      Long chatId = callbackQuery.getMessage().getChatId();

      try {
        if ("/test_correct".equals(callbackData)) {
          execute(new SendMessage(chatId.toString(), MessageTest.CorrectAnswer));
          Thread.sleep(1000);
          quizHandler.correctAnswer(this, callbackQuery.getFrom(),
              callbackQuery.getMessage().getChat(), null);
        } else if ("/test_incorrect1".equals(callbackData) || "/test_incorrect2".equals(callbackData)
        || "/test_incorrect3".equals(callbackData) || "/test_incorrect4".equals(callbackData)) {
```

```
          quizHandler.incorrectAnswer(this, callbackQuery.getFrom(),
              callbackQuery.getMessage().getChat(), null);
          }
        } catch (TelegramApiException | InterruptedException e) {
          e.printStackTrace(); }}}

  @Override
  public void onUpdatesReceived(List<Update> updates) {
    super.onUpdatesReceived(updates);
  }
}
```