

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ЗАКЛАД
«ЛУГАНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА»

Навчально-науковий інститут математики
та інформаційних технологій

Кафедра математики та інформатики

Латкіна Тетяна Валеріївна

РОЗРОБКА ОСВІТНЬОГО ДОДАТКУ ДЛЯ ТЕСТУВАННЯ

Магістерська робота
за спеціальністю
014.09 «Середня освіта. Інформатика»

Особистий підпис – _____

Науковий керівник – _____ ас. Козуб В.Ю.

В.о. зав. кафедри – _____ д.т.н., доцент Козуб Ю.Г.

Старобільськ – 2023

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ.....	3
ВСТУП.....	4
1. ДОСЛІДЖЕННЯ ОСВІТНІХ РЕСУРСІВ ДЛЯ ТЕСТУВАННЯ.....	6
1.1. Огляд освітніх платформ.....	6
1.2. Огляд та порівняння чат-ботів.....	12
Висновки до розділу 1.....	20
2. МЕТОДОЛОГІЯ РОЗРОБКИ ДОДАТКУ.....	21
2.1. Моделі створення програмного забезпечення.....	21
2.2. Вибір мови програмування.....	28
2.3. Форми тестових завдань у середовищі освіти.....	38
Висновки до розділу 2.....	42
3. СТВОРЕННЯ ОСВІТНЬОГО ДОДАТКУ ДЛЯ ТЕСТУВАННЯ.....	43
3.1. Загальні поняття тестових ботів.....	43
3.2. Проектування чат-боту.....	44
3.3. Опис роботи чат-боту для тестування.....	54
3.4. Апробація чат-боту.....	57
Висновок до розділу 3.....	60
ВИСНОВОК.....	61
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	63

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ІТ – інформаційні технології - це система методів, процесів та способів використання обчислювальної техніки і систем зв'язку для створення, збору, передачі, пошуку, оброблення та поширення інформації з метою ефективної організації діяльності людей.

ПЗ – програмне забезпечення.

Фреймворк (англ. *Framework*, *каркас*, *платформа*) — інфраструктура програмних рішень, що полегшує розробку складних систем.

Чат-бот – спеціальна алгоритмічна програма, яка автоматично обробляє повідомлення від користувачів і надає правильні та логічні відповіді.

ТЗ – тестові завдання.

Парсер – програма, що забезпечує парсинг, тобто синтаксичний аналіз контенту в мережі по певній математичній моделі, створеній на одній з мов програмування.

Ітератор — це об'єкт, який дозволяє програмісту обходити елементи колекції, зокрема списку.

ВСТУП

Тема: Розробка освітнього додатку для тестування.

Оскільки події останніх кількох років призвели до того, що діти не мають змоги навчатися у звичайному режимі (карантин, війна) – дистанційна форма навчання стала основою подальшого здобуття освіти. І якщо технічне та програмне забезпечення зробили таке навчання доступним та, взагалі, можливим, залишається ще досить багато не вирішених запитань. Одним із них є рівень якості такого навчання та, відповідно, методи та критерії оцінювання здобутих знань та навичок. Звісно є традиційні контрольні роботи, але перевіряти сфотографовані роботи буває не просто складно, а і не можливо через якість зображення тощо. Є альтернативні варіанти: проєктні завдання, хоча вони більше зорієнтовані на креативне мислення, ніж на засвоєння матеріалу. Тому одним із методів проведення контролю знань залишається тестування: швидкий спосіб отримати результати контролю. Це дає змогу отримати певні результати за досить короткий проміжок часу у порівнянні з іншими видами контролю знань. Тому *об'єктом* дослідження є дистанційна та змішана форми навчання. *Предметом* же дослідження являється чат-бот для месенджера Telegram, який створений для проведення та аналізу тестування учнів у даному середовищі.

Окрім технології дистанційного навчання, досить суттєве місце у житті сучасного здобувача освіти займають соціальні мережі, месенджери, що дозволяють підтримувати зворотній зв'язок без будь-яких перешкод. Telegram вважається одним з провідних месенджерів. Він характеризується зручністю, простотою, надійністю та сучасним середовищем для вільного і, головне, безпечного спілкування. Саме завдяки поєднанню цих характеристик його обирають багато користувачів. І одним з основних інструментів Telegram є чат-бот, який використовується для автоматизації повсякденних процесів. У більшості випадків людям зручніше спілкуватися за допомогою повідомлень, ніж дзвінків. Саме тому чат-боти активно використовуються, адже вони

дозволяють людям робити багато речей онлайн не витрачаючи багато часу на очікування відповідей на їх запити. Враховуючи вищезазначене, використання чат-боту для проведення тестування – це не лише *актуальне* рішення даної проблеми, а й сучасний підхід, який однозначно оцінить кожен учень.

Тому *метою* даної роботи є створення чат-боту в месенджері Telegram який дасть змогу проводити тестування учнівських знань та автоматично видаватиме результати їхньої успішності.

Гіпотеза: використання таких методів оцінювання як тести, не лише допомагають зекономити час при формуванні завдань та їх подальшій перевірці, підрахунку середнього балу результату успішності, але й дозволяє швидко зорієнтуватися, у яких саме питаннях найчастіше виникають проблеми розуміння чи засвоєння матеріалу в учнів, щоб мати змогу внести відповідні корективи у свою роботу при планування уроків.

1. ДОСЛІДЖЕННЯ ОСВІТНІХ РЕСУРСІВ ДЛЯ ТЕСТУВАННЯ

1.1. Огляд освітніх платформ.

В умовах дистанційного та змішаного навчання різко зросла актуальність платформ і сервісів онлайн-навчання, які забезпечують зворотний зв'язок між викладачами та учнями за допомогою тестів і вправ.

Тестові опитування надзвичайно популярні. Адже вони є швидким та ефективним способом перевірки рівня засвоєння здобувачами освіти навчального матеріалу та автоматичного формування статистики успішності учнів за результатами виконаних робіт. Підбираючи навчальні матеріали та готуючи завдання до уроку, вчителі повинні враховувати такі рекомендації:

- Обсяг, інтенсивність та рівень складності завдання має бути вдвічі меншим, ніж на звичайному уроці;
- Менше навантаження означає коротший урок. Наприклад, для молодших школярів традиційний 45-хвилинний урок слід розбити на дві сесії;
- Вчителі повинні постійно отримувати зворотний зв'язок від учнів. Це означає, що учень має знати, чи сподобалося йому завдання, що було незрозумілим, що було найскладнішим і т. д.;
- Необхідне тестування складності завдання і, за необхідності, зміна його на більш легке та зрозуміле;
- Необхідність структурування навчального матеріалу, створення папок із завданнями на Google Діску чи інших ресурсах;
- Подача теорії у вигляді презентації або використання відео;
- Використання різних типів завдань, таких як тести, розгорнуті відповіді, вікторини, ігри, спільні проєкти тощо.

Сьогодні численні платформи електронного навчання мають схожі переваги та недоліки, і це розмаїття дуже ускладнює для пересічного викладача вибір найбільш підходящого інструменту для реалізації дистанційних режимів навчання. У більшості випадків керівництво закладу відповідає за організацію такого режиму роботи та має навчити науково-

педагогічних працівників навичкам роботи з платформою дистанційного навчання.

У галузі освіти багато дослідників України [1; 2; 3] та за кордоном [4; 5; 6; 7] вивчали питання використання інформаційно-комунікаційних технологій та веб-технологій. Однак проблеми навчання учнів з використанням сучасних інформаційно-комунікаційних технологій не отримали належного розкриття, що й визначило актуальність даного дослідження.

Як свідчать результати аналізу даного дослідження, навчальні заклади в основному використовують безкоштовні ресурси іноземних розробників програм та компаній. На разі найбільш поширеними є платформи дистанційного навчання:

- Google Classroom;
- MOODLE;
- Kiddom;
- Edmodo.

В основу вибірки для аналізу покладено ідею порівняння організаційних та функціональних характеристик кількох різних платформ, якими користуються викладачі різних типів акредитованих навчальних закладів, згідно зі статистичними даними.

Розглянемо найбільш корисні безкоштовні онлайн-платформи.

Kahoot (рис. 1.1). Це навчальна платформа, яка дозволяє проводити інтерактивні уроки та перевіряти знання учнів за допомогою онлайн-тестів. У ній можуть брати участь до 50 дітей.



рис. 1.1 Логотип Kahoot

Вчителі можуть створювати власні завдання або вибирати готові з колекції запитань. Також є банк зображень, які можна використовувати для візуалізації завдань, додавати до запитань або використовувати як відповіді. Платформа дозволяє створити діаграму прогресу академічної групи та дізнатися, як кожен здобувач освіти відповідав на запитання.

Використовуючи безкоштовну версію платформи, можна створювати два типи запитань: вікторини, тобто запитання з множинним вибором, де школярі мають вибір і обирають одну або кілька правильних відповідей, та запитання типу "істина-хибність", де надаються два взаємовиключні варіанти відповідей.

Quizizz - це сервіс для створення вікторин і тестів. Він використовує подібні до Kahoot методи навчання. Quizizz (рис.1.2) можна використовувати в реальному часі в класі (групах), а також як самостійну позакласну роботу для учнів, які обрали режим "Грати наживо" або "Домашнє завдання".



рис. 1.2. Логотип Quizizz

Сервіс дозволяє відстежувати прогрес кожного учня та експортувати дані в Excel. Вчителі можуть використовувати готові тести з бібліотеки Quizizz або створювати власні тести, ігри чи опитування.

Quizlet (рис. 1.3) – це сервіс для розробки цікавих тестів та флеш-карт. Тут можна обирати тип навчання із запропонованого списку – тести, письмо,

запам'ятовування, флеш-картки, орфографія, гравітація, встановлення відповідності тощо, і, окрім того, завантажити відповідний матеріал для тесту.



рис. 1.3. Логотип Quizlet

OnlineTestpad (рис.1.4) – це зручний і безкоштовний багатофункціональний сервіс для онлайн-навчання. За допомогою цього конструктора є можливість створювати кросворди, логічні ігри та діалогові тренажери, у тому числі і тести.



рис. 1.4 Логотип Online Test Pad

Classtime (рис.1.5) – онлайн-сервіс, який дозволяє створювати власні завдання різних типів, отримувати доступ до бази даних готових запитань з різних предметів та проводити швидке тестування за допомогою смартфона. Він може відстежувати прогрес кожного учня, створювати уроки та експортувати звіти у форматі Excel та PDF.



рис. 1.5 Логотип Classtime

Moodle (Modular Object-Oriented Dynamic Learning Environment) - це модульне об'єктно-орієнтоване динамічне навчальне середовище, також відоме як система управління навчанням (LMS), система управління курсами (CMS), віртуальне навчальне середовище (VLE) або просто навчання Moodle і є найсучаснішою та найпопулярнішою системою в Україні та світі. Наразі Moodle налічує 129 мільйонів користувачів по всьому світу і продовжує зростати набагато швидшими темпами, ніж його конкуренти.

Moodle (рис.1.6) – це безкоштовна система з відкритим кодом. Вона не лише безкоштовна, але й не потребує жодного платного програмного забезпечення для роботи. Це означає, що всі навчальні заклади можуть впровадити сучасну та повністю ліцензовану систему, не витрачаючи жодної копійки на програмне забезпечення. При цьому вони можуть модифікувати код відповідно до власних потреб.



рис. 1.6 Логотип Moodle

Важливою особливістю проєкту Moodle є веб-сайт. Останній є централізованим джерелом інформації про систему, а також форумом для обговорення та співпраці між користувачами Moodle (системними адміністраторами, викладачами, дослідниками, дизайнерами та розробниками) [22]. Завдяки цьому Moodle підтримує інтерфейси більш ніж 80 мовами, включаючи українську. Перевагою платформи електронного навчання Moodle також є той факт, що вона неодноразово модифікувалася і доповнювалася новими рішеннями та інструментами з моменту її запуску. Програмне забезпечення платформи написано на мові PHP з використанням безкоштовних загальнодоступних баз даних (MySQL, PostgreSQL).

Платформа Moodle може бути встановлена на будь-яку операційну систему (MS Windows, Unix, Linux). Платформа відповідає всім основним критеріям, що пред'являються до системи онлайн навчання, у тому числі:

- Функціональність – доступні різні рівні функціональності (форуми, чат, аналіз активності студентів, управління курсами та навчальними групами тощо);
- Надійність – зручність управління та адміністрування навчання, простота оновлення контенту на основі існуючих шаблонів, захист користувачів від зовнішніх маніпуляцій тощо;
- Стабільність – висока стабільність системи за різних режимів роботи та активності користувачів;
- Вартість – сама система є безкоштовною, а витрати на впровадження, розробку та підтримку курсу мінімальні;
- Необмежена кількість студентських ліцензій;
- Модульність – набори блоків матеріалів навчального курсу можуть бути використані в інших курсах;
- Наявність вбудованих інструментів для розробки та редагування навчального контенту; інтеграція різних матеріалів для різних цілей;
- Підтримка міжнародного стандарту SCORM (Sharable Content Object Reference Model) – основа для обміну електронними курсами та гарантія перенесення ресурсів в інші системи;
- Наявність системи онлайн-тестування та оцінювання знань здобувачів освіти (тести, завдання, моніторинг активності на форумах);
- Зручність і простота використання та навігації – інтуїтивно зрозумілі прийоми навчання (можливість легко знаходити довідкові меню, легкість переходу від одного розділу до іншого, спілкування з тьюторами тощо).

Але при всіх суттєвих перевагах платформ для тестування учнів, існує і певна кількість недоліків, які впливають на їхню ефективність та прийняття:

а). Відсутність контексту: тестові завдання можуть бути відокремленими від реального життя та позбавленими контексту. Учні можуть виявляти велику успішність на тесті, але не завжди мати можливість застосувати свої знання в реальних ситуаціях.

б). Недостатній зворотній зв'язок: деякі платформи можуть надавати обмежений зворотній зв'язок щодо результатів тестування, що ускладнює для учнів розуміння своїх помилок та вдосконалення знань.

в). Залежність від технологій: освітні платформи для тестування можуть бути залежними від технічних аспектів, таких як доступ до Інтернету та комп'ютерів. Це може створювати нерівності в доступі до можливостей тестування між учнями.

г). Можливість обману: технологічні платформи можуть стикатися з проблемою обману або шахрайства, коли учні використовують заборонені джерела під час тестування.

Тому все більше привертають увагу альтернативні варіанти у цифровому середовищі навчання дітей. Одним з таких варіантів є застосування чат-ботів.

1.2. Огляд та порівняння чат-ботів.

Дистанційна освіта не є новим етапом у розвитку освіти. Піонером вважається Оксфордський університет, який запровадив першу дистанційну освіту в 1780 році [6]. Його наступником став І. Пітман у 1840 році, який запропонував британським студентам навчання поштою.

На разі найпоширенішою формою дистанційної освіти є комп'ютерна, з різними освітніми можливостями залежно від використовуваної конфігурації (текстові файли, мультимедійні технології, відеоконференції).

Варто проаналізувати переваги та недоліки цього виду освіти. Безперечно, головною перевагою є можливість вибору тривалості навчання та темпу засвоєння знань відповідно до індивідуальних потреб дитини. Учні мають можливість вирішувати, коли і як вони хочуть навчатися. Крім того, навіть найбільш невпевнений і сором'язливий учень може ставити запитання

через електронну пошту, чат і дискусійні форуми. на додаток до багатьох інших переваг використання електронного навчання, важливим аспектом є постійно зростаюча кількість додаткових матеріалів, доступних для навчання. Дослідження показують, що значна кількість дітей скептично ставиться до комп'ютерної техніки або має труднощі з її використанням[3]. Це є основною перешкодою для засвоєння нових навичок. У той же час, спілкування у месенджерах та соціальних мережах стає, фактично, частиною повсякденного життя. І сучасні реалії змушують більшість людей проводити багато часу в спілкуванні онлайн. І, насправді багато завдань можна вирішити, не звертаючись безпосередньо до оператора. Це допомагає економити час. Це завдання покладено на чат-боти. Це потужний інструмент для підтримки та побудови ефективних відносин з користувачами[8]. Тож розглянемо більш детально існуючі чат-боти і їх застосування у різних сферах.

Механізм роботи чат-ботів базується на заздалегідь визначених сценаріях спілкування і може швидко реагувати на дії співрозмовника. Іншими словами, у чат-ботах користувач спілкується не з живою людиною, а з роботом [23].

При створенні чат-боту вирішальне значення має вибір типу. Існує два типи чат-ботів: складні та прості. Складні чат-боти оснащені штучним інтелектом, тому він здатен розпізнавати складні фрази та запитання різними мовами. Прості чат-боти працюють за інструкцією: вони можуть відповідати лише на ті запитання, які вже прописані в системі; месенджери, зокрема Telegram, використовують саме простих ботів, і саме цей тип буде проаналізовано в подальшому.

Для того, щоб визначитися з вибором методу розробки програмного забезпечення, необхідно розглянути переваги та недоліки основних чат-ботів.

Перевагами є:

- Цілодобова доступність і простота використання;
- Миттєва реакція;
- Інтерфейс простий для розуміння та використання;

- Легкість у використанні;
- Високий рівень безпеки особистої інформації;
- Підтримка декількох користувачів одночасно;
- Оптимізація процесів;
- Зручність для користувача;
- Автоматизація простої комунікації.

Класифікація чат-ботів.

У нашій країні чат-боти використовуються в багатьох державних, транспортних і фінансових установах. Також вони популярні в медицині, різних месенджерах і соціальних мережах. Найчастіше вони інтегровані з такими месенджерами: Viber, Telegram, WhatsApp, Facebook Messenger тощо. Деякі також привертають увагу клієнтів для вирішення конкретних завдань: їх застосовують у сервісах електронної комерції, колл-центрах, ігровій індустрії тощо.

З точки зору реалізації ботів розрізняють дві категорії чат-ботів: бізнес-класифікацію та технічну класифікацію.

Бізнес-класифікація чат-ботів включає в себе розмовні чат-боти, які не мають конкретного призначення і створюються для спілкування; та чат-боти із заздалегідь визначеною метою. Вони використовують запитання для отримання даних, необхідних для досягнення заданих цілей. Третя група – це Q&A чат-боти, що призначені для відповідей на запитання; вони працюють за принципом надання однієї відповіді на одне запитання. Такі боти можуть замінити FAQ (поширені запитання) на різних сайтах.

Так само існує три класи чат-ботів за технічною класифікацією: на основі бізнес-правил, на основі штучного інтелекту та гібридні.

Перший клас ботів має деревоподібну структуру діалогу. Скрипти, створені для ботів, не дозволяють їм відхилятися від заздалегідь визначених розробником шляхів. Якщо користувач відповідає у форматі, якого немає в скрипті бота, така розмова не підтримується. У такому випадку бот надає альтернативу у вигляді кнопки з конкретним вибором.

Другий клас ботів повністю використовує штучний інтелект. Тут немає початково визначеного сценарію для підтримки розмови. Бот самостійно вирішує, які питання ставити і як на них відповідати, на основі попередніх взаємодій, що використовуються в машинному навчанні. Такі боти потребують великого набору вхідних даних, щоб діалог з користувачем мав логічну послідовність. Наразі існують лише прототипи таких чат-ботів.

Гібридні чат-боти – це комбінація перших двох типів. Вони використовують як початково написані сценарії для спілкування, так і штучний інтелект для розпізнавання намірів користувача та вилучення цінних персональних даних (наприклад, прізвище, ім'я, дата народження) з розмов користувачів. Такі чат-боти використовуються в різноманітних програмних комерційних додатках і є найбільш поширеними.

Запорукою успішної розробки продукту є якісне дослідження ринку та аналіз конкуренції. Тому перед тим, як почати проектувати і розробляти бот необхідно розглянути вже існуючі програми.

Для визначення унікальності ідеї чат-бота для тестування, було проведено пошук в інтернеті на предмет наявності подібних ботів. У результаті точних аналогів не було знайдено, оскільки тема чат-ботів продовжує розвиватися. Розглянемо найпопулярніші чат-боти в Telegram.

Andy English Bot – це чат-бот, призначений для вивчення та практики англійської мови (рис.1.7 та рис. 1.8). Він дозволяє спілкуватися, виконувати вправи, вивчати граматику та грати в різноманітні ігри. З ботом можна практикувати повсякденні теми для розмов: робота, погода, знайомство та багато іншого.

Програма розроблена таким чином, що є можливість використовувати англійську в листуванні, відповідаючи на запитання та вивчаючи нову лексику та граматику. Крім того, є щоденне проходження коротких уроків граматики, щоб поглибити розуміння англійської мови.

Якщо трапляється незрозуміле слово, Andy надішле його визначення та приклад речення, щоб допомогти закріпити знання. Чат-бот також може виправляти помилки під час розмови.



Andy English Bot

@andyrobot

рис. 1.7 Andy English Bot

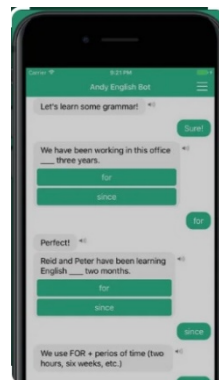


рис. 1.8 Вигляд боту Andy English Bot

Skeddy – це чат-бот, який надійно нагадує вам про завдання, які потрібно виконати у визначений час (рис.1.9). Все, що потрібно зробити, це написати повідомлення і обрати час, коли потрібно отримати сповіщення.



Skeddy

@skeddybot

Skeddy is a simple yet powerful reminder tool that can help you create and manage your reminders.

<https://skeddy.me/>

рис. 1.9. Телеграмм бот Skeddy

Бот має наступні функції:

- Створює прості, одноразові нагадування;
- Створює нагадування, які повторюються через певний проміжок часу;
- Перегляд списку нагадувань;
- Зміна нагадувань (наприклад, часу, інформації);
- Вимкнення та ввімкнення одного і того ж нагадування;
- Створення звичайного нагадування без встановлення часу.

"СтопСуржик" – чат-бот, створений для перекладу суржиком українською мовою. Слова, яких немає в словнику, можна додавати вручну.

Необхідно написати слово суржиком і бот видає його український відповідник. Це корисний тренажер для вдосконалення мови. Бот ще навчається, тому якщо не вдалося знайти певне слово суржиком, можна додати його український аналог до словника бота (рис. 1.10).

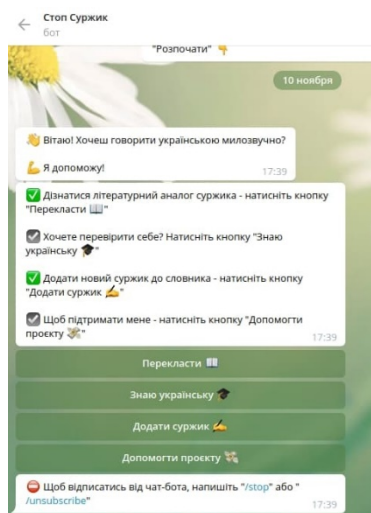


рис. 1.10. Телеграмм бот «СтопСуржик»

«Баба Катря» - бот, який допомагає покращити знання української мови, особливо потренуватися в акцентуванні. Одразу після початку спілкування бот запитує, як користувач хоче навчатися. Є два варіанти: "навчити мене" або "відправити мене на курси онучок". Якщо користувач обирає першу відповідь, «Баба Катря» (рис. 1.11) одразу пояснює, як його навчити. Цікавим є те, що Баба Катря по-різному реагує на правильні та неправильні відповіді (рис. 1.12).



рис. 1.11. Телеграмм бот «Баба Катря»



рис. 1.12. Вигляд чат-боту «Баба Катря»

Якщо ж обирається другий варіант "Відправте мене на курс для онучок", одразу з'являється посилання на сайт озвученого курсу.

І список подібних «помічників» постійно розширюється.

Загальні переваги та функції чат-ботів.

Чат-боти відповідають на усні та письмові запити, знаходять відповіді на запитання та підбирають необхідний контент. Вони використовують машинне навчання, щоб відповідати на запити та створювати сценарії спілкування; підтримувати розмови з клієнтами цілодобово; детально аналізувати комунікації; оптимізувати витрати та підвищувати залученість аудиторії. Це основні плюси, які привертають увагу до даних розробок.

Програми, вбудовані у боти, впливають на їхній машинний інтелект. Чат-боти на основі машинного навчання не просто виконують команди, а й можуть розпізнавати мову. Вони краще розуміють запити і тому можуть надавати відповідні результати. Вони також можуть надсилати масові електронні листи, дозволяючи охопити ширшу аудиторію.

Можливості чат-ботів.

1. Обслуговування та підтримка клієнтів. Запити клієнтів обробляються 24 години на добу. Це включає повідомлення, відповіді на запитання та інтерактивні опитування. Це значно зменшує навантаження на службу підтримки та підвищує продуктивність.

2. Інструменти для маркетингу.

Чат-боти допомагають охопити цільову аудиторію, збирати дані про клієнтів для подальшого аналізу та координації маркетингових активностей, отримувати зворотній зв'язок, інформувати про знижки, акції та нові продукти, надсилати маркетингові листи.

3. Асистенти у сфері страхування та охорони здоров'я.

Чат-боти існують не лише для стандартизованого інформування клієнтів, але й для допомоги клієнтам. Наприклад, перший в Україні медичний бот Марта, доступний у Viber і Telegram, дозволяє клієнтам записуватись на

прийом до лікаря онлайн, ставити запитання оператору (за потреби), бронювати консультації, отримувати інформацію про вакцинацію від COVID-19, права пацієнтів, безоплатні медичні послуги, допомагає отримати та оцінити свій результат після візиту до лікаря.

4. Інструменти електронної комерції.

Це інструменти, якіможуть допомогти мотивувати людей робити покупки. Прикладом такого чат-ботає онлайн-помічник компанії Farmasi, міжнародного виробника косметики та товарів для краси. Телеграм-бот @FarmasiDigital_bot спочатку запитує, чи ви вже є клієнтом компанії. Якщо ні, то пропонує ознайомитися з актуальним каталогом, отримати знижку на перше замовлення та дізнатися про безліч бізнес-можливостей.

5. Асистенти для сфери освіти.

Можуть продавати освітні послуги, відповідати на запити потенційних студентів, отримувати зворотній зв'язок від студентів, надавати навчальні матеріали та розміщувати посилання на навчальні платформи.

Висновки до розділу 1

Тож стає досить очевидним, що використання тестів для оцінювання знань учнів має значну кількість переваг. Це і ефективність, оскільки тести дають можливість оцінити знання досить широкого спектру тем за короткий період часу. Це особливо важливо при масовому оцінюванні, наприклад, під час екзаменів. Це також стандартизація (можна порівнювати результативність учнів, які проходять один і той же тест).

Окрім того, тести є зручними для аналізу: результати тестів можуть бути швидко оброблені та аналізовані, що дозволяє ефективно визначити, на яких темах учні можуть потребувати додаткової допомоги, на які питання слід звернути уваги і, можливо розібрати останні додатково.

А дослідження різних інтернет-джерел показує, що саме чат-боти є досить потужним інструментом, який набуває все більшої популярності завдяки цілому ряду переваг. Telegram-боти стрімко з'являються і замінюють людину в спілкуванні.

Чат-боти можуть легко обробляти запити користувачів у будь-який час, інформувати їх про новини компанії, проводити інтерактивні опитування або відповідати на запитання. Боти можуть підвищити задоволеність клієнтів, оскільки вони завжди доступні для відповіді. Це значно підвищує продуктивність роботи.

Такі програми можна використовувати віддалено, що є доречним, особливо в питаннях дистанційного чи змішаного навчання.

2. МЕТОДОЛОГІЯ РОЗРОБКИ ДОДАТКУ

2.1. Моделі створення програмного забезпечення

Розробка відмінного програмного продукту починається із формування його життєвого циклу. Це чіткий план дій, за яким розробники розуміють, що їм потрібно робити, як досягти результатів і які методи використовувати. Методологія розробки програмного забезпечення-це набір перевірених методів і практик для створення цифрових продуктів точно і ефективно В ІТ є кілька основних методологій розробки програмного забезпечення, на які сьогодні варто звернути увагу.

Кожна модель розробки ПЗ має свої унікальні характеристики, сильні та слабкі сторони. Досить складно визначити, яка з них краща, оскільки різні завдання, продукти та ідеї вимагають різних принципів розробки. Нижче наведені основні типи, що застосовуються в програмуванні.

«Waterfall Model» (каскадна модель).

Це одна з основних моделей, яку використовували на початку ери розвитку, але активно застосовують і до сьогодні. Принцип її роботи достатньо простий: наступний етап здійснюється лише після остаточного завершення попереднього. Етапи чітко розділені, а технологія реалізується каскадно, покроково переходячи від першого етапу до останнього [24].

Така модель розробки програмного забезпечення досить конкретна і має жорсткі правила. Терміни виконання кожного етапу чітко визначені. Однак є й недоліки. Внесення змін в існуючий проєкт є дуже дорогим і проблематичним. Цей метод підходить тільки для чітко визначених проєктів, де повністю зрозуміло, що будується, для чого будується і які вимоги були поставлені. Цей метод можна використовувати, якщо проєкт має детальний прототип або існуючий додаток.

«Incremental Model» (інкрементна модель).

Модель поетапної розробки програмного продукту підходить, якщо є чітко встановлені всі аспекти роботи, але продукт потрібно запустити досить швидко і внести зміни пізніше. Її суть полягає в тому, що з самого початку

розплановані дії розбиваються на невеликі завдання. Крім того, кожен блок сегментується відповідно до традиційної каскадної моделі. Спочатку виготовляється "базовий" продукт з мінімальними, але основними функціями. І надалі він доповнюється інших компонентів, так званими *інкрементами*. Цей процес повторюється до тих пір, покине буде повністю зібрана єдина система.

Кожна окрема частина опрацювання такого проєкту є готовим елементом. Іноді її можна застосовувати індивідуально. Як правило, її розробляють, щоб не повторювати спробу пізніше. Тому каскадна модель використовується в рамках інкрементальної моделі.

Що надає така методологія розробки ПЗ. Перш за все, ризик зводиться до мінімуму і гарантується швидкий випуск і запуск продукту. Крім того, основні функції вже працюють, а зміни та доповнення можна вносити поступово із часом.

«Iterative Model» (ітеративна або ітераційна модель).

Цей підхід дещо схожий за дизайном з попереднім підходом. Принцип ітеративної методології створення програмного продукту полягає в створенні і поступовому поліпшенні базових функцій. Поки що він схожий на попередню версію, але з певними відмінностями. Покрокову модель своєю сутністю схожа на пазл, де всі елементи складаються один за іншим і таким чином утворюють єдине зображення.

Цей метод є актуальним для великих проєктів, де визначені головні напрямки і є основне уявлення про те, що має статися. Але в той же час не чітко визначені деталі, і не зовсім зрозуміло, як працює та чи інша функція. Такий підхід дає змогу доопрацьовувати програму за рахунок нових компонентів і покращення вже існуючих.

«Agile Model» (гнучка методологія).

Гнучка методологія розробки ПЗ – відмінне рішення для створення продуктів, які неповністю сформовані з урахуванням цієї дії. Відмінною особливістю цього методу є те, що клієнти можуть швидко відстежувати зміни в розробці і координувати дії. Це можливо шляхом визначення сегмента

спринту, на якому буде виконуватися завдання. В ході обговорення ставляться цілі і визначається час сегмента, протягом якого виконується завдання. Потім проходить обговорення, пропонуються корегувальні дії і визначаються нові сегменти.

Даний метод являється досить ефективним, але у нього також є певні недоліки. Через причини неможливості визначення точного результату і відсутності розуміння, скільки часу потрібно на реалізацію цілей, заздалегідь визначити точну вартість не представляється можливим.

«Spiral Model» (спіральна модель).

Спіральна модель підходить для масивних проєктів, де допущення помилки може призвести до значних втрат і неприємних наслідків. Сутність методу полягає в наступному:

На кожному етапі виконуються чотири основні дії: планування, аналіз ризиків, розробка і конструювання, оцінка результатів і зворотній зв'язок. Якщо всі чотири етапи будуть успішно завершені, робота перейде на новий етап. Перехід на новий рівень аналогічний інкрементальній моделі, в якій кожен блок розробляється індивідуально і бере участь в базових функціях, створених на початку. Відмінною особливістю такого підходу є те, що на аналіз, розрахунок і оцінку результатів витрачається більше часу, ніж безпосередньо на саму розробку. Іноді процес впровадження функції відбувається набагато швидше, ніж досліджуються аналітичні відгуки.

«RAD Model» (швидка розробка).

Даний метод підходить у тих випадках, коли на меті стоїть запуск продукції у найкоротші терміни. Ідея полягає в тому, щоб розділити етап розробки програми на кілька окремих блоків, кожним з яких займається окрема людина або навіть команда.

Невеликі модулі, над якими ведеться робота, потім збираються в єдину систему, утворюючи робочу версію, яку можна запускати на апробацію. Потім, завдяки зворотному зв'язку, вносяться зміни. Цей метод робить

можливим створення навіть складних продуктів за короткий проміжок часу, проте має деякі особливості:

- є необхідність постійно контролювати результати;
- метод є складним (а в деяких випадках і дорогим) через потребу залучення певної кількості людей чи навіть цілої команди;
- для кожного модуля визначена своя задача, яку необхідно чітко розуміти, щоб отримати відповідні результати для заданого блоку.

Такий підхід дозволяє швидко генерувати ідеї, додавати нові продукти в короткі терміни і створювати потужні додатки з багатим функціоналом.

Об'єктно-орієнтоване програмування.

ООП – це об'єктно-орієнтований підхід до розробки програмного забезпечення. Тобто програма розділена на набір об'єктів, які взаємопов'язані один з одним.

Об'єкти, класи, атрибути та методи складають структуру даного принципу розроблення ПЗ.

Об'єкт – це екземпляр класу, що представляє реальну або абстрактну сутність.

Клас – шаблон для утворення об'єктів, що визначаються атрибутами і методами.

Атрибут – це дані, які зберігають стан об'єкту.

Метод – це функція, що змінює стан об'єкту або виконує певні дії.

Підхід об'єктно-орієнтованого програмування базується на чотирьох основних принципах: інкапсуляція, успадкування, поліморфізм та абстракція.

Інкапсуляція. Всі висхідні дані та методи, що пов'язані із ними, зберігаються в об'єкті. Це дозволяє приховати внутрішню реалізацію та виводить у доступ інтерфейс, що необхідний для співпраці. Це дозволяє створювати надійні та безпечні програми та запобігати несанкціонованому доступу та внесенню змін сторонніми людьми.

Успадкування. ООП дозволяє створювати нові класи на основі існуючих. Похідним являється новостворений клас, а вже існуючий – носить назву

базовий або батьківський. При успадкуванні похідний клас отримує всі атрибути та методи базового класу і може додавати власні. Це дозволяє повторно використовувати один і той же код, утворюючи ієрархію "від загального до приватного" для реалізації складних схем [9]. Це допомагає підвищити ефективність розробки та забезпечити більш логічну організацію коду. Немає потреби постійно переписувати однієї ті ж властивості для різних об'єктів, досить просто успадкувати їх від одного "батька".

Поліморфізм. Об'єкти різних класів можуть мати аналогічні методи, проте реалізовуватимуть їх по-різному. Програма працює з усіма об'єктами, використовуючи загальний інтерфейс, що робить код гнучкішим і більш універсальним.

Абстракція. Абстракція фокусується на тому, що найважливіше, замість того, щоб детально описувати кожен частину системи окремо. Це дозволяє не акцентувати увагу на певній складності їх реалізації, а зосередити увагу на ключових аспектах об'єкту.

Перевагами ООП є його модульність та можливість повторного використання коду. В даному випадку є змога розділити додаток на невеликі, зрозумілі частини, які відповідають за певні функції. Можна безперешкодно повторно використовувати модулі, класи та об'єкти для спрощення розробки, налагодження та підтримки програмного забезпечення.

Окрім того, варто наголосити на гнучкості та масштабованості додатків, оскільки такий підхід дозволяє додавати нові класи та методи без необхідності переписувати весь код, таким чином адаптуючи програму до будь-яких змін у вимогах. Простота обслуговування говорить сама за себе: кожен об'єкт має власні функції та дані, що полегшує пошук та виправлення помилок. Програмне забезпечення із захищеним інкапсульованим кодом важко розшифрувати.

Досить проста інтеграція – ще одна перевага даного методу. Швидка інтеграція різних компонентів додатку, новостворених об'єктів, що взаємодіють один з одним, полегшує розробку більш складних систем.

Але у будь-якій методиці завжди є певні недоліки. Даний метод не є виключенням. Більш складне навчання у порівнянні з процедурним програмуванням, що вимагає більшої кількості часу і зусилля, щоб повністю засвоїти та практикувати концепцію ООП.

Можливе дублювання коду та надмірність. У випадку, коли дизайн класів та об'єктів неправильний, може виникнути ситуація, що деякі частини коду реалізують один і той же функціонал, що призведе до надмірності та проблем. Крім того, до перевантаження пам'яті призводить наявність додаткової інформації, такої як методи та властивості.

Принцип об'єктно-орієнтованого програмування широко використовується у різних областях розробки.

Веб-розробка. Класи, об'єкти, успадкування та поліморфізм дозволяють створювати різні типи користувачів, продуктів, замовлень та інших сутностей на основі загальних шаблонів. Розробка ігор. Створення нових персонажів, додавання предметів, світів та інших елементів із загальними характеристиками. І таких прикладів можна навести безліч.

Подійно-орієнтоване програмування.

Подійно-орієнтоване програмування – це парадигма програмування, в якій виконання програми визначається подіями-діями користувача (клавіатура, миша), повідомленнями від інших програм або потоків, подіями операційної системи (наприклад, надходженням мережевих пакетів).

ПОП також можна визначити як спосіб побудови комп'ютерної програми, при якому в коді (як правило, в межах основної функції програми) явно виділяється основний програмний цикл, тіло якого складається з двох частин: отримання повідомлень про події та обробка подій.

В принципі, в реальній задачі неприпустимо запускати обробники подій протягом тривалого часу, оскільки програма не зможе реагувати на інші події. У зв'язку з цим при написанні програм, орієнтованих на події, часто використовується автоматичне програмування.

Програмування, кероване подіями, використовується в серверних додатках для вирішення проблеми масштабування до 10000 і більше одночасних з'єднань.

Сервери, побудовані за моделлю "один потік на з'єднання", мають проблеми з масштабуванням з наступних причин:

- Накладні витрати на структури даних операційної системи (сегменти стану завдань, стеки), необхідні для визначення одного завдання, дуже високі;
- Занадто багато накладних витрат при перемиканні між контекстами.

У подійно-орієнтованому програмуванні серверні додатки реалізуються на мультиплексованих системних викликах, які отримують повідомлення про події від багатьох ідентифікаторів одночасно. При обробці подій використовуються лише неблокуючі операції вводу/виводу, так що жоден дескриптор не блокує операції подій від інших дескрипторів [10].

У сучасних мовах програмування події та обробники подій є центральною частиною реалізації графічних інтерфейсів користувача. Наприклад, програма взаємодіє з подіями миші. При натисканні правої кнопки миші генерується системне переривання і запускається процедура в операційній системі. Ця процедура шукає вікно під курсором миші. Якщо вікно знайдено, ця подія надсилається до черги повідомлень цього вікна. Залежно від типу вікна, можуть генеруватися додаткові події. Наприклад, якщо вікно є кнопкою (у Windows всі графічні елементи є вікнами), додаткова подія генерується на натискання кнопки. Різниця між останніми подіями полягає в тому, що вони є більш абстрактними, тобто не містять координат курсору, а лише вказують на те, що кнопка була натиснута.

Обробник події може виглядати наступним чином (на прикладі C #):

```
private void button1_Click (object sender, EventArgs e)
{
    MessageBox.Show ("Була натиснута кнопка");
}
```

Тут обробник події – це процедура, якій передається параметр-відправник, зазвичай вказівник на джерело події. Це дозволяє одній процедурі обробляти події від декількох кнопок і розрізняти їх за цим параметром.

2.2. Вибір мови програмування

Вибір мови програмування для написання освітнього додатку залежить від ряду факторів, таких як функціональні вимоги, масштаб проєкту, вимоги до продуктивності та інші аспекти [11].

C# (вимовляється як "C sharp") - це мова програмування, розроблена компанією Microsoft. Вона була випущена у 2000 році і стала однією з основних мов для платформи .NET. C# є повністю об'єктно-орієнтованою мовою програмування, що означає, що все в мові є об'єктами та статично типізованою мовою, що дозволяє визначити тип змінної під час компіляції. Це сприяє виявленню помилок на ранніх етапах розробки. Мова має вбудовані механізми безпеки, такі як перевірка меж для масивів та автоматичне виправлення помилок вказівників.

Хоча C# була розроблена для використання на платформі Windows, існують ініціативи, такі як .NET Core та .NET 5+, які дозволяють використовувати C# на різних платформах, таких як Linux та macOS. Також C# інтегрована зі середовищами розробки, такими як Visual Studio, що робить розробку більш зручною та продуктивною.

C# використовується для розробки різноманітних програм, включаючи десктопні застосунки, веб-додатки, ігри, мобільні додатки для платформи Windows та Android, а також корпоративні рішення.

Java є мовою програмування, яка стала вкрай популярною завдяки своїй платформонезалежності, об'єктно-орієнтованому підходу та високому рівню безпеки. Java використовує віртуальну машину Java (JVM), яка дозволяє виконувати код на будь-якому пристрої, який має JVM, іншими словами код, написаний цією мовою, може бути використаний на різних операційних системах без будь-яких змін. Синтаксис Java подібний до мов C та C++, що робить його досить зрозумілим для розробників, які вже мають досвід роботи

з даними мовами. Окрім того, у даної мови високий рівень безпеки, так як вбудовані механізми дозволяють управляти доступом до ресурсів та забезпечують безпеку віртуальної машини. Розробникам легко створювати програми, які можуть виконувати багато завдань одночасно, тому що Java підтримує мультипоточність та має велику та активну систему з багатьма бібліотеками, фреймворками та іншими інструментами, що полегшують розробку [25].

Java використовується у різних областях, включаючи веб-розробку (застосунки на основі JavaServer Pages), мобільну розробку (застосунки Android), корпоративні системи (застосунки на основі Java EE), наукові дослідження та багато іншого. Java визначається своєю надійністю, широкою розповсюдженістю та зручністю використання, що робить її однією з найпопулярніших мов програмування у світі.

Скриптова мова програмування *JavaScript* використовується як мова сценаріїв для веб-сторінок, що виконується безпосередньо у браузері користувача. Її синтаксис, схожий до інших мов програмування, зокрема Java та C і включає в себе такі конструкції як умови, цикли, функції та об'єкти [12]. Тип змінної може змінюватися під час виконання програми, при цьому він підтримує типи даних, наприклад, рядки, числа, масиви та об'єкти.

JavaScript є об'єктно-орієнтованою мовою програмування, де програма складається з об'єктів, які можуть включати в себе властивості та методи. Дана мова дозволяє використовувати функції як об'єкти першого класу, тобто їх можна передавати як аргументи, повертати з інших функцій та зберігати в змінних. JavaScript підтримує обробку подій та асинхронне програмування. Це особливо важливо в веб-розробці, де може виникати потреба в відповідях на події користувача без блокування виконання коду. Це середовище програмування використовується в основному для розробки веб-додатків і підтримується практично на всіх сучасних браузерах. Також воно може бути використаним на серверному боці за допомогою платформ, таких як Node.js.

JavaScript відомий своєю гнучкістю та великою кількістю можливостей, що робить його однією з основних мов програмування для розробки веб-додатків та веб-сайтів.

Окремо хотілося б розглянути середовище Node.js.

Node.js - це серверна платформа для виконання JavaScript-коду на стороні сервера. Вона дозволяє розробникам використовувати ту саму мову програмування як на стороні клієнта, так і на стороні сервера, що полегшує обмін даними та розробку з боку повного стеку. Однією з основних особливостей Node.js є асинхронна обробка подій [13]. Вона використовує один потік для обробки запитів, використовуючи зворотні виклики (callbacks) та події. Це дає змогу досягати високої ефективності, особливо в операціях вводу/виводу.

Node.js вбудовує систему модулів, яка дозволяє розробникам організовувати свій код в окремі модулі. Кожен модуль може бути відокремленим файлом або папкою, яку можна використовувати в інших проєктах.

Node.js широко використовується для створення серверів, API, мережесх застосунків, інструментів розробки та інших видів програмного забезпечення. Дане середовище є потужним інструментом для розробки серверних застосунків, особливо для тих, де важлива висока ефективність та обробка багатьох одночасних підключень.

Ruby - це динамічна, об'єктно-орієнтована мова програмування, яку розробив Японець Юкихіро Мацумото (Yukihiro Matsumoto). Ruby була вперше випущена у 1995 році і швидко набула популярності завдяки своєму простому та елегантному синтаксису, який нагадує природну мову та полегшує розробку. У Ruby все є об'єктом, включаючи примітивні типи даних, що сприяє консистентності та зручності у програмуванні. Є можливість змінювати типи змінних під час виконання програми [26].

Ruby підтримує багаторівневий розробку, що дозволяє організовувати код у логічні модулі та класи. Масиви та об'єкти у даному середовищі можуть

динамічно розширюватися та змінюватися, що робить роботу з даними більш гнучкою.

Ruby знаходить своє застосування у різноманітних областях, включаючи веб-розробку (зокрема з фреймворками, такими як Ruby on Rails), автоматизацію.

Python - це високорівнева, інтерпретована, об'єктно-орієнтована мова програмування, яку розробив Гвідо ван Россум. Python став однією з найпопулярніших та найвикористовуваних мов програмування завдяки своїй простоті, читабельності коду та широкому спектру застосувань.

І саме дану мову програмування є резон розглянути більш детально, оскільки вона не просто широко застосовується при написанні програм та сервісів; вона нас цікавить ще й як частина шкільного курсу, в якому учні знайомляться із фундаментальними основами кодування мовою Python.

Python був розроблений Гвідо ван Россумом і вперше випущений у 1991 році. Мова отримала свою назву на честь популярного телевізійного шоу "Monty Python's Flying Circus" [14].

Мова програмування підтримує декомпозицію вирішення задач кількома різними способами:

- Об'єктно-орієнтоване програмування (маніпуляції з колекціями об'єктів. Об'єкти мають внутрішній стан і підтримують методи для запиту або модифікації цього внутрішнього стану певним чином);
- Функціональне програмування (розкладає проблему на набір функцій). В ідеалі, функція тільки приймає вхідні і видає вихідні дані, і немає внутрішнього стану, який впливає на вихідні результати, отримані на заданий вхід.
- Процедурне програмування (засноване на концепції процедурних викликів). Процедури також відомі як підпрограми, методи та функції (вони більше схожі на функції, що використовуються у функціональному програмуванні, ніж на математичні функції). Процедура містить певну послідовність кроків, які потрібно виконати. Під час виконання програми

процедуру можна викликати з будь-якого місця програми, включаючи саму процедуру (рекурсивні виклики).

Python є мультипарадигмальною мовою, яка дозволяє писати процедурні, об'єктно-орієнтовані та функціональні програми і бібліотеки. Наприклад, графічний інтерфейс може бути об'єктно-орієнтованим, тоді як логіка його роботи може бути процедурною або функціональною [15].

У функціональному стилі не рекомендується використовувати функції з побічними ефектами, які змінюють внутрішній стан або вносять інші невидимі зміни у значення, що повертається функцією. Функції без побічних ефектів називаються *чистими* функціональними стилями. Уникнення побічних ефектів означає відмову від використання структур даних, які оновлюються під час виконання, і кожен вихід функції повинен залежати тільки від її вхідних даних.

Функціональне програмування є протилежністю об'єктно-орієнтованого програмування. Об'єкт – це маленька капсула, що містить певний внутрішній стан і набір викликів методів для зміни цього стану, а програма полягає у правильній зміні стану. Функціональне програмування максимально уникає змін стану і має справу з потоком даних між функціями; у Python ці два підходи можна поєднати, написавши функції, які отримують і повертають екземпляри, що представляють об'єктив додатку (повідомлення електронної пошти, транзакції тощо) [27].

Але функціональний стиль має і свої як теоретичні, так і практичні переваги:

- а). Формальна доказовість.
- б). Модульність.
- в). Комбінованість.
- г). Легкість налагодження та тестування.
- д). Формальна доведеність.

Синтаксис Python призначений для зручності, нагадує природну мову, що полегшує розробку та обслуговування коду, і, між іншим, це інтерпретована мова, яка дозволяє виконувати код без необхідності компіляції, що пришвидшує розробку та стимулює до експериментів [28].

Ітератори.

Ітератор – це об'єкт, який представляє потік даних і повертає по одному елементу за раз, а ітератори Python повинні підтримувати метод `__next__()`. Цей метод не приймає аргументів і завжди повертає наступний елемент потоку. Якщо у потоці більше немає елементів, `__next__()` повинна згенерувати виключення `StopIteration`. Однак, ітератори не обов'язково повинні бути скінченними. Має сенс написати ітератор, який створює нескінченний потік даних [29].

Вбудована функція `iter()` приймає випадковий об'єкт і намагається повернути ітератор, який повертає вміст або елементи цього об'єкта; деякі вбудовані типи даних Python підтримують ітерацію, найпоширенішими з яких є списки та словники. Об'єкти, які можуть отримувати ітератор, називаються *ітерабельними*.

Python дає можливість використовувати об'єкти, моделювати класи, та використовувати функціональні конструкції, має велику кількість стандартних бібліотек та сторонніх модулів, що розширюють його можливості. Це робить Python дуже гнучкою та ефективною мовою для різноманітних завдань. Дана мова програмування підтримується на різних операційних системах, включаючи Windows, macOS та Linux, що робить її кросплатформенною і переносимою.

IDE (Integrated Development Environment) – це програмне забезпечення, яке надає розробникам повний набір інструментів для написання, тестування, налагодження та розгортання програм. IDE займає центральне місце в робочому процесі більшості програмістів, оскільки надає все необхідне для ефективного написання коду в одному місці.

Існує декілька факторів, які впливають на важливість IDE для розробки Python:

- **Продуктивність:** IDE включають в себе ряд інструментів, які можуть значно підвищити продуктивність роботи програміста, наприклад, автодоповнення коду, покажчики помилок в реальному часі, вбудовані відладчики та системи контролю версій.
- **Співпраця та підтримка:** інтегровані середовища розробки допомагають спростити процес спільної роботи над проєктами, надаючи стандартизоване середовище розробки. Багато IDE мають вбудовану підтримку систем контролю версій, таких як Git, що дозволяє розробникам легко відстежувати та об'єднувати зміни в коді.
- **Навчання:** Для початківців IDE є чудовим інструментом для вивчення Python. Крім того, IDE можуть надавати корисні поради та полегшувати навчання, вказуючи на типові помилки.

Visual Studio Code (VS Code) – це безкоштовне середовище розробки з відкритим вихідним кодом, що надається Microsoft. Хоча це не спеціалізоване середовище програмування, VS Code можна використовувати для кодування на Python завдяки його розширюваності та гнучкості.

Функції VS Code, які є корисними для розробки на Python:

- **Розширення Python,** розроблене Microsoft для VS Code, надає багато можливостей, включаючи інтелектуальне автозавершення, компонування, форматування коду, налагодження та підтримку.
- **Вбудоване налагодження:** VS Code має вбудовані інструменти налагодження коду, які дозволяють користувачам встановлювати точки зупинки, переглядати змінні та контролювати виконання коду.
- **Вбудована підтримка Git:** VS Code постачається з підтримкою Git, що дозволяє легко створювати коміти, виштовхувати та витягувати оновлення, а також переглядати відмінності між версіями без необхідності використання командного рядка.

- Підтримка розширень, що дозволяє налаштувати середовище розробки відповідно до конкретно заданих потреб. Доступно багато розширень, які підтримують різні мови, бібліотеки, фреймворки, інструменти форматування коду та інші функції.

- Налаштування та гнучкість: VS Code дуже добре налаштовується. Все, від тем і шрифтів до способу роботи редактора коду та його інтеграції з іншими інструментами, можна налаштувати.

Хоча Visual Studio Code не є спеціалізованим інструментом для розробки на Python, його гнучкість та підтримка розширень є дуже корисною для програмування. Це особливо актуально для розробників, які працюють з декількома мовами та технологіями і потребують такої гнучкості.

Jupyter Notebook – популярний інструмент розробки з відкритим вихідним кодом серед науковців, викладачів, студентів та програмістів, які працюють з даними. Це інструмент для програмування, аналізу та візуалізації даних, важливий інструмент для побудови моделей машинного навчання. Він особливо популярний серед користувачів Python, але також підтримує багато інших мов програмування.

Основні особливості Jupyter Notebook:

- Інтерактивність: Jupyter Notebook дозволяє користувачам запускати код в інтерактивних "блокнотах", які поєднують код, текст, формули та візуалізації. Це дозволяє працювати з кодом і даними у більш наочний та експериментальний спосіб.

- Підтримка Markdown та LaTeX: Jupyter Notebook підтримує Markdown для форматування тексту і LaTeX для математичних виразів, що робить його ідеальним для створення зрозумілих і красивих документів, що містять код.

- Інтеграція з Python та іншими мовами: хоч Jupyter був створений в першу чергу для Python, багато інших мов програмування також підтримують Jupyter Notebook, включаючи Julia, R, Ruby, Haskell і багато інших.

- Розподілені обчислення: за допомогою таких інструментів, як Apache Spark, Jupyter може виконувати розподілені обчислення для обробки великих наборів даних.

- Підтримка бібліотек: Jupyter інтегрується з багатьма популярними бібліотеками Python, такими як NumPy, Pandas, Matplotlib і Scikit-Learn, що дозволяє користувачам маніпулювати даними, виконувати складні обчислення і створювати візуалізації в браузері.

- Співпраця та віддалена робота: Jupyter Notebook можна налаштувати для віддаленої роботи та співпраці, що робить його корисним інструментом для команд.

Це далеко не повний список IDE, які можна використовувати при роботі з Python, але вже стає очевидним, що завдяки таким додатковим інструментам, програмування стає значно зручнішим, доступнішим навіть для початківців.

Саме тому Python досить широко застосовується у різних областях, таких як веб-розробка (зокрема з Django та Flask), наукові дослідження, штучний інтелект, обробка даних, автоматизація завдань та багато інших [16]. Python легко взаємодіє з кодом, написаним на інших мовах програмування, таких як C і C++, що дозволяє використовувати їхні функціональні можливості.

Що ж стосується питання створення чат-ботів у Telegram, різні мови програмування можуть використовувати дуже широкий спектр додаткових інструментів, таких як набори бібліотек, фреймворків тощо.

Наприклад:

1. Node.js (JavaScript/TypeScript).

Для Node.js існують фреймворки, які спрощують роботу з Telegram API. Наприклад, варто скористатися *telegraf*, який є потужним фреймворком для створення чат-ботів.

Приклад використання telegraf:

```
const { Telegraf } = require('telegraf');

const bot = new Telegraf('YOUR_TELEGRAM_BOT_TOKEN');

bot.start((ctx) => ctx.reply('Привіт!'));
bot.hears('Привіт', (ctx) => ctx.reply('Привіт! Я бот в Телеграмі.'));
bot.on('text', (ctx) => ctx.reply(ctx.message.text));

bot.launch();
```

2. Java:

Бібліотека: Для Java можна використовувати бібліотеку *TelegramBots* для роботи з Telegram API.

Приклад використання TelegramBots:

```
import org.telegram.telegrambots.bots.TelegramLongPollingBot;
import org.telegram.telegrambots.meta.api.objects.Update;
import org.telegram.telegrambots.meta.exceptions.TelegramApiException;
import org.telegram.telegrambots.meta.generics.LongPollingBot;

public class MyBot extends TelegramLongPollingBot {

    @Override
    public void onUpdateReceived(Update update) {
        // Обробка вхідного повідомлення
        // ...
    }

    @Override
    public String getBotUsername() {
        return "YOUR_BOT_USERNAME";
    }

    @Override
    public String getBotToken() {
        return "YOUR_TELEGRAM_BOT_TOKEN";
    }

    public static void main(String[] args) {
        MyBot bot = new MyBot();
        bot.run();
    }
}
```

3. Python:

Python є однією з найпопулярніших мов програмування для створення чат-ботів. Декілька відомих фреймворків для роботи з Telegram API включають *python-telegram-bot* та *Telepot*.

Приклад використання бібліотеки *python-telegram-bot*:

```
from telegram import Update
from telegram.ext import Updater, CommandHandler, MessageHandler, Filters, CallbackContext

def start(update: Update, context: CallbackContext) -> None:
    update.message.reply_text('Привіт!')

def echo(update: Update, context: CallbackContext) -> None:
    update.message.reply_text(update.message.text)

def main() -> None:
    updater = Updater("YOUR_TELEGRAM_BOT_TOKEN")

    dp = updater.dispatcher
    dp.add_handler(CommandHandler("start", start))
    dp.add_handler(MessageHandler(Filters.text & ~Filters.command, echo))

    updater.start_polling()

    updater.idle()

if __name__ == '__main__':
    main()
```

І останнє, що варто розглянути перед початком безпосереднього створення освітнього додатку, це сам принцип тестових систем.

2.3. *Форми тестових завдань у середовищі освіти*

Основними формами ТЗ є:

- а). Відкриті форми – без зазначення відповіді;
- б). Закриті форми, які в свою чергу діляться на:
 1. Вибір однієї правильної відповіді;
 2. Множинний вибір;
 3. Альтернативна відповідь;

4. Встановлення правильної відповідності [17].

Відкрите тестове завдання – це один із форм тестування, що включає:

1. Створення повної відповіді учасників тестування (від окремого слова, словосполучення, речення до цілого тексту);
2. Учасник тестування відповідає правилам, визначеним в критеріях розподілу за обсягом, змістом і т. д.;
3. Запис відповіді у вказаному місці (наприклад, у формі) для учасника тестування.

Кількість відповідей на відкриті тестові завдання варіюється від окремої цифри, слова, словосполучення до цілісного тексту у вигляді творів, статей, документів і т. д.

Перший тип ТЗ закритого типу – це з *вибором однієї правильної відповіді*. Даний метод оцінювання полягає в тому, щоб учасники вибирали лише одну правильну відповідь із заданого переліку альтернатив. Цей тип широко використовується в освітніх і оцінювальних програмах, оскільки дозволяє швидко та ефективно оцінювати знання і розуміння учнів чи учасників.

Тести з вибором однієї правильної відповіді можуть бути проведені швидше порівняно з іншими формами тестування, оскільки перевірка відповідей є стандартизованою та автоматизованою. При цьому, оцінка тесту залежить від правильності вибору відповідей, що дозволяє уникнути суб'єктивності при оцінці.

Цей тип тестування легко координується за допомогою комп'ютерних платформ, і результати можуть бути швидко оброблені та представлені [18].

Проте важливо враховувати, що тести з вибором однієї правильної відповіді мають свої обмеження, зокрема, вони можуть бути менш ефективними в оцінці творчих та критичного мислення учасників. Також, деякі учні можуть "вгадувати" правильні відповіді, що може вплинути на точність оцінки їхніх знань.

Тестові завдання типу *множинного вибору* є одними із найпоширеніших видів питань в освітній та оціночній практиці [19]. Вони використовуються для оцінки різних рівнів знань та навичок учнів. Структура даного типу заключається у тому, що кожне питання має одну або кілька правильних відповідей серед варіантів, які надаються. До кожного питання надається набір варіантів відповідей, включаючи один чи кілька правильних та декілька неправильних. Відповіді можуть бути сформульовані як текст, числа, графіки чи інші форми, при цьому кількість правильних відповідей може бути різною, від одного до декількох, а загальна кількість становитиме – від двох до декількох десятків варіантів, але загалом обмежується так, щоб вибір був відносно простим, але не занадто очевидним. Важливо, щоб формулювання та варіанти були такими, щоб учень знав, яка відповідь є правильною, а інші варіанти виглядали логічно.

Множинний вибір може використовуватися для тестування різних рівнів навичок, від знань фактів до вміння розв'язувати складні завдання та аналізувати інформацію.

Формулювання коротких відповідей є ще одним типом тестових завдань, який використовується для оцінювання різних рівнів знань та навичок учнів. Вони можуть бути ефективним інструментом для вимірювання розуміння концепцій та здатностей до висловлення інформації. При такій формі тестування учням подається коротке питання або вказівка, і їм слід надати відповідь у короткій формі, зазвичай, у вигляді короткого речення чи фрази. Дані питання формулюються таким чином, щоб вони вимагали від учнів конкретних, чітких відповідей, але при цьому не були занадто вузькими чи невизначеними. Основна мета такого типу - дозволити учням висловлювати свої думки конкретно та лаконічно.

Але є певний недолік такого способу проведення контролю: оцінка відповідей може бути більш суб'єктивною, оскільки вона зазвичай вимагає від оцінювача аналізу відповідей та їх порівняння з критеріями.

Тести з короткими відповідями можуть бути частиною різних типів оцінювання, включаючи письмові тести, онлайн-тести та інші форми оцінювання знань учнів.

Альтернативний вибір відповіді є ще одним типом тестування, де учасники повинні вибрати правильну відповідь з числа альтернатив. Основна відмінність від тестування з вибором однієї правильної відповіді полягає в тому, що тут може бути кілька правильних відповідей, і учасник повинен вибрати одну чи кілька з них. Завдяки наявності кількох правильних відповідей, цей тип тестування може сприяти глибшому розумінню теми та підвищенню рівня когнітивного аналізу. Також у завданнях можуть бути враховані різні аспекти теми чи предмету, що дозволяє отримати різноманітну інформацію про знання учасників. Учні можуть вибирати нестандартні чи творчі відповіді, що сприяє розвитку їх творчого мислення.

Оскільки кількість варіантів відповідей обмежена, це може зменшити ймовірність вгадування правильної відповіді. Інакше кажучи, тестування з альтернативним вибором відповіді може бути більш розширеним та комплексним, але водночас воно може бути більш складним у підготовці та оцінці результатів.

І це далеко не всі види форм, які можуть бути застосовані на практиці, але навіть такий огляд дає змогу обрати найбільш коректний для створення майбутнього освітнього додатку.

Висновки до розділу 2

Створення чат-ботів у телеграмі може вимагати використання різних мов програмування. Проте саме Python є найбільш доцільною у застосуванні для розробки чат-боту, особливо при умові створення тестування множинного вибору за ітераційною моделлю програмування. Особливо важливим фактором при виборі мови програмування став фактор швидкості та простоти розробки чат-боту. У Python існує велика кількість готових бібліотек для обробки тексту, роботи з API телеграма та іншими завданнями, пов'язаними з чат-ботами, що ще більше полегшує виконання даного завдання.

Окрім того, Python підтримує ітераційний підхід до програмування, що дозволяє поступово додавати новий функціонал та тестувати його, вдосконалювати та впроваджувати нові можливості.

В цілому, використання Python для розробки чат-бота в Telegram разом з тестуванням множинного вибору за ітераційною моделлю програмування може сприяти ефективній та швидкій роботі, а також полегшити підтримку та вдосконалення функціоналу в майбутньому.

3.СТВОРЕННЯ ОСВІТНЬОГО ДОДАТКУ ДЛЯ ТЕСТУВАННЯ

3.1. Загальні поняття тестових ботів

Суть полягає у перевірці знань за допомогою тестів. За допомогою такого інструменту викладачам набагато простіше проводити контрольні, тематичні іспити, а також відстежувати результати успішності своїх учнів. Тепер немає необхідності реєструвати результати, вираховувати кількість правильних/не правильних відповідей та обчислювати середнє значення оцінки. Сучасна система онлайн-тестування допомагає відстежувати прогрес кожного учня, щоб дозволяє уникнути складних розрахунків. Сам інструмент такого тестування стає зрозумілішим і зручнішим для всіх, як для вчителів, так і для здобувачів освіти.

Боти для проведення тестування здобувачів освіти використовуються для автоматизації процесу тестування та оцінювання. Характерною особливістю такого інструменту є наявність так званого банку даних: колекція запитань, з якої обираються завдання для створення конкретного тесту. Певна система класифікації цих даних дозволяє управляти та знаходити завдання з конкретних тем. Збір та аналіз даних про результати тестів для отримання інформації про продуктивність та ефективність відбувається автоматично і швидко.

Ці поняття є важливими для розуміння та реалізації ботів для тестування учнів, які можуть полегшити та покращити процес проведення тестів та оцінювання знань.

Враховуючи все вище зазначене, створення додатку передбачало ще кілька функціональних особливостей. По-перше, бот повинен мати зрозумілий і простий інтерфейс. По-друге, навігація має являти собою меню з кнопок. І, найголовніше, кількість спроб проходження тестування має бути необмеженою, оскільки саме такий підхід дозволяє врахувати свої помилки і не допускати їх повторно. Також досить важливим моментом є те, що чат-бот

може працювати на будь-якому пристрої з встановленим Telegram і підключеним до інтернету.

3.2. *Проектування чат-боту*

Із загалу доступних бібліотек взаємодії з Telegram було обрано саме PyTelegramBotAPI, оскільки вона є лише дуже тонкою «обгорткою» відповідних сервісів. Використання вказаної бібліотеки надає можливість ознайомитись з механізмами роботи сервісів Telegram як це було реалізовано їх розробниками, а не розробниками самої бібліотеки.

Згідно документації PyTelegramBotAPI надає можливість створення ботів двох режимів роботи: polling та web-hooks. В першому випадку (polling) запусканий скрипт телеграм бота регулярно відправляє запити до сервісів Telegram для отримання нових повідомлень, що надходять користувачу. В останньому ж (web-hooks), навпаки, при надходженні нових повідомлень сервер Telegram буде намагатись зв'язатись з запускеним чат-ботом та повідомити про присутність нових повідомлень. Для реального використання в шкільних умовах набагато зручнішим є перший режим функціонування, оскільки не вимагає присутності сервера з доступною (виділеною) IP адресою.

Суттєвою особливістю чат-ботів Telegram (як і програмування графічних інтерфейсів взагалі) є їх подійна-орієнтованість (event-driven). На відміну від «послідовних» програм, які виконуються від початку та завершуються в кінці, «подійна» програма має бути постійно запускеною та реагувати (обробляти) події, що стаються в програмному середовищі. В випадку Telegram ботів найпоширенішими подіями є отримання команд або повідомлень від користувачів. Бібліотека pyTelegramBotAPI дозволяє зареєструвати будь-яку функцію Python в якості обробника команд або повідомлень за допомогою спеціальних декораторів.

Важливим наслідком «подійної» орієнтовності та багатокористувацького режиму є необхідність підтримки «сесій» користувачів. Кожному відкритому сеансу роботи з ботом Telegram призначає

унікальний ідентифікатор, який може використовуватись для збереження даних користувача в деякому сховищі. Прикладами таких даних користувача є його ім'я, вік, статистика взаємодії з ботом тощо. Найпростішим сховищем «сесій» може бути простий словник Python, який використовує унікальний ідентифікатор сеансу в якості ключа.

Вже на етапі початкового проєктування системи необхідно закласти деяку «гнучкість» в її подальше функціонування. Так, в процесі створення прототипів, необхідні дані можуть бути інтегровані у вихідний код. Для прикладу, питання та варіанти відповідей тестів можуть бути присутні в вихідному коді в вигляді списків словників. Очевидно, що в процесі використання системи це не буде зручним, оскільки додавання, видалення та зміна питань, відповідей до них вимагатиме безпосередніх змін вихідного коду. Загальноприйнятим вважається винесення таких «гнучких» даних в конфігурацію системи, або зовнішню базу даних [20]. Тоді для проведення маніпуляцій не потрібно буде змінювати вихідний код системи.

Представлений тест-бот використовує набір файлів в форматі JSON для збереження тем, питань та відповідей до них. JSON файли є дуже зручними текстовими представленнями списків та словників даних та можуть бути використані з усіма сучасними мовами програмування. Окрім того, формат є також «людино»-орієнтовним, тобто може безпосередньо редагуватись в будь-якому зручному текстовому редакторі. Цей факт було використано в представленому проєкті та дозволило уникнути складнощів створення окремих адміністративних застосунків для конфігурування тест-боту.

Хотілося б дещо детальніше зупинитися на темі даного формату. JSON (JavaScript Object Notation) - це текстовий формат, відповідальний за збереження структурованих даних. Створений Дугласом Крокфордом (американський програміст) на основі JavaScript, але незалежний від нього, JSON легко поєднується із сучасними середовищами програмування, зокрема PHP, Python, Java і Ruby.

Файли JSON мають однойменне розширення .json, формат також може бути представлений в інших типах файлів (наприклад, .html) і відображатися у вигляді рядка або об'єкта JSON. Важливою особливістю цього стандарту є те, що рядки JSON виглядають як звичайний текст і можуть бути легко прочитані людиною так само, як і інші текстові формати. JSON часто порівнюють з XML, що являється досить популярним форматом даних, але перший має цілий ряд переваг, які привертаються увагу:

- JSON простіший у використанні і містить менше даних, ніж XML;
- JSON – це єдиний спосіб обміну даними між різними веб-сайтами;
- У поєднанні з AJAX він дозволяє асинхронне завантаження даних у фоновому режимі, що робить ресурси швидшими і кориснішими для користувачів;
- Можна використовувати масиви даних, які неможливі, наприклад, у тому ж самому XML.
- Його можна розбирати стандартними інструментами, тоді як інші формати можна розбирати тільки за допомогою спеціальних парсерів.
- Все більше онлайн-сервісів підтримують цей формат у своїх API.
- Всі сучасні мови програмування здатні читати і редагувати JSON-файли, а сам формат легко адаптується до різних програмних середовищ. Зберігання даних у текстовому форматі ізначно спрощує їх передачу існуючими мережевими каналами.

Формат JSON був створений для зручності зберігання даних в процесі обміну інформації між веб-браузерами і веб-сайтами або між різними веб-сайтами. Це текстовий формат даних, основною одиницею якого є пара "ключ-значення", і який може оброблятися як JS, так і іншими популярними мовами програмування.

Окрім того, JSON, разом з AJAX, дозволяє вносити зміни на веб-сайтах і веб-додатках без необхідності оновлення сторінки.

Для створення «швидкого» прототипу представленого тест-боту було вирішено реалізувати його в вигляді процедурного скрипта мови Python. Для представлення даних було використано вбудовані типи мови Python, списки та словники. Як показала подальша розробка, переведення проекту до об'єктно-орієнтованого підходу не було необхідним, оскільки отриманий код був вже достатньо лаконічним та не передбачав суттєвого ускладнення в подальшому.

Розробка

На зараз виділяють два основні підходи до розробки програмного забезпечення: каскадна («водоспадна») та ітераційна. Перша передбачає чітке розділення всього робочого періоду на три стадії: проектування, реалізації та впровадження. Друга пропонує проводити розробку додаючи програмний функціонал невеликими ітераціями кожна з яких містить ті самі три стадії, що й в каскадній моделі. Кожна з моделей має свої переваги та недоліки, але каскадна модель є дуже чутливою до використання нових технологій, оскільки важко проводити планування та проектування для незнайомих бібліотек та платформ [21]. Через це для реалізації даного проекту було обрано ітеративну модель та вирішено почати розробку зі створення «швидкого» коректно працюючого прототипу з мінімальним функціоналом.

Для реєстрації нового боту з використанням вбудованого BotFather було достатньо виконати команду `/new` та надати необхідну інформацію за запитами. Результатом реєстрації нового бота є його ім'я та спеціальний секретний токен, який необхідно надати конструктору класу `TeleBot` для створення об'єкта. Вже перший «швидкий» прототип було реалізовано в вигляді повноцінного телеграм-боту, а взагалом для розробки було пророблено 4 наступні ітерації (табл. 1):

Таблиця 1

№ ітерації	Опис
0	«швидкий» прототип – повноцінний Telegram бот з чотирма питаннями та відповідями присутніми безпосередньо у вихідному коді
1	Зчитування теми з питаннями та відповідями з одного зовнішнього json файлу (підтримка лише однієї теми тестування)
2	Зчитування усіх json файлів з директорії groups (підтримка декількох тем). Ведення статистики відповідей поточної теми.
3	Підтримка імені користувача, збереження та відображення результатів тестувань

Механізм роботи

Як вже зазначалось, всі подійно-орієнтовані програми мають бути постійно запущені для обробки подій, що настають. PyTelegramBotAPI дозволяє досягти такої поведінки (polling режим) шляхом створення об'єкту бота (`bot = telebot.TeleBot(«TOKEN»)`) та виклику його функції (`infinity_polling`). Ця функція не повертає керування коду, що її визиває, а постійно виконує так званий цикл обробки повідомлень.

Telegram – обробники

Бібліотека PyTelegramBotAPI дозволяє будь-яку функцію Python призначити обробником подій через застосування декоратора `@bot.message_handler` [30]. При цьому існує два основні типи подій Telegram – команди та повідомлення. Перші призначені для виконання окремих операцій ботів (запустити ті чи інші процеси, відобразити ту чи іншу інформацію тощо), а другі для передачі їм довільних значень. PyTelegramBotAPI дозволяє призначити окремі обробники команд та

повідомлень через передачу різних параметрів декоратору `@bot.message_handler`.

Обробники команд Telegram

Через `@bot.message_handler(commands=[{команда}])` позначено наступні функції-обробники команд:

Таблиця 2.

Функція	Команда	Опис
start(message)	/start	Повний (пере)запуск тест-бота. Тестування буде перервано.
select_user(message)	/user	Зміна інформації користувача. Може бути виконана в будь-який момент.
select_group(message)	/test	Вибір теми для тестування. Поточне тестування буде перервано.
show_results(message)	/results	Відображення результатів пройдених тестів. Може бути виконано в будь-який момент.

Обробники повідомлень Telegram

Декоратор `@bot.message_handler(content_types='text')` дозволяє позначити функцію, яку буде викликано для обробки будь-яких текстових повідомлень від користувача (окрім вже оброблених команд). В коді тест-бота є тільки одна така функція:

Таблиця 3

Функція	Опис
process_message(message)	Проводить загальну обробку повідомлень-відповідей. Повторює запитання в разі неочікуваної відповіді або переходить до

	наступного. Після відповіді на останнє запитання
--	--

Додаткові функції

Наступні функції створено для зручності та виключення повторень в коді проекту (табл. 4). Всі ці функції отримують об'єкт сесії в якості другого параметра. Використовуються при реалізації функцій-обробників Telegram подій.

Таблиця 4

Функція	Опис
process_user_message(message, session)	Обробка інформації користувача. Введена строка буде збережена в поточній сесії.
process_group_message(message, session)	Обробка обраної теми для тестування. Назва теми буде збережена в поточній сесії.
process_test_message(message, session, group)	Обробка відповіді на запитання тесту поточної теми. В залежності від результату питання буде або задано повторно, або буде здійснено перехід до наступного питання теми.
send_question(message, session, test)	Відправка користувачеві тестового запитання
finish_group(message, session)	Завершення теми з відображенням статистики
validate_message(message, session)	Перевірка коректності відповіді користувача (чи є значення варіантом відповіді). В разі

неочікуваної відповіді питання буде задано повторно.

Завантаження даних

Мова Python має вбудовану бібліотеку для роботи з файлами формату Json. Її функції використовуються для завантаження всіх Json файлів з директорії groups, які повинні містити інформацію про питання та відповіді до тестів конкретної теми [31]. Кожен Json файл має містити лише один словник наступної структури:

```
{
  "name": "Інформаційні процеси",
  "tests": [
    {
      "question": "1. Інформацію можна:",
      "answers": ["Отримувати", "Зберігати", "Опрацьовувати", "Їсти"],
      "correct": ["Отримувати"]
    },
    {
      "question": "2. Набір спеціальних програм, які керують роботою всіх пристроїв комп'ютера та",
      "answers": ["Комп'ютерна система", "Операційна система", "Інформаційна система"],
      "correct": ["Операційна система"]
    },
    {
      "question": "3. Який інтерфейс використовують у сучасних операційних системах?",
      "answers": ["Графічний", "Командний", "Звуковий"],
      "correct": ["Графічний"]
    },
    {
      "question": "4. Головне вікно комп'ютера – це:",
      "answers": ["Мій комп'ютер", "Пуск", "Робочий стіл"],
      "correct": ["Робочий стіл"]
    },
    {
      "question": "5. Головне меню комп'ютера – це:",
      "answers": ["Мій комп'ютер", "Пуск", "Робочий стіл"],
      "correct": ["Пуск"]
    }
  ]
}
```

Таблиця 5

Функція	Опис
load_groups()	Завантажує всі json файли з піддиректорії groups робочої директорії бота.

Для полегшення доступу та кешування переліку тем та їх запитань, результат роботи функції load_groups() зберігається в змінній groups одразу після запуску бота.

Підтримка сесій користувачів

Для забезпечення роботи тест-боту необхідно зберігати деяку додаткову інформацію для кожного сеансу Telegram користувача. Сукупність такої додаткової інформації для певного сеансу називають «сесією». Найпростішою реалізацією сесії є звичайний python-словник, де додаткові дані зберігаються під певними ключами:

Таблиця 6

Ключ словника сесії	Опис
curr_user	Ім'я користувача сеансу
curr_group_name	Обрана тема тестування
curr_test_idx	Індекс поточного питання, яке опрацьовує користувач
started_at	Час початку тестування по обраній темі
finished_at	Час закінчення тестування по обраній темі
correct_answers_count	Кількість правильних відповідей по обраній темі
incorrect_answers_count	Кількість хибних відповідей по обраній темі
last_used_keyboard	Об'єкт telegram-клавіатури для поточного питання
last_used_buttons	Варіанти відповідей на поточне питання

Сесії користувачів також зручно розмістити в словнику, де ключами виступають певні ідентифікатори сеансів, а значеннями – словники-сесії. Бібліотека PyTelegramBotAPI надає доступ до таких ідентифікаторів, через будь-який об'єкт повідомлення: `message.chat.id`. Для зручності та уникнення повторень у проєкті використовуються наступні функції для роботи з сесіями користувачів:

Таблиця 7

Функція	Опис
init_session(session_id)	Створює та зберігає сесію користувача з наданим ідентифікатором

find_session(session_id)	Шукає та повертає сесію користувача з наданим ідентифікатором
clear_session(session_id)	Очищує дані сесії з наданим ідентифікатором. Окрім інформації про користувача.
find_group(session)	Повертає словник даних поточної теми (інформація про поточну тему береться з наданої сесії)
find_test(session)	Повертає словник даних поточного тесту (інформація про поточний тест береться з наданої сесії)

Самі сесії розміщені в словнику `sessions`, який ініціалізується в процесі запуску бота. Оскільки сесії тест-бота не зберігаються в жодне довгострокове сховище, всі дані користувачів автоматично втрачаються при перезапуску програми-боту. Але це не є проблемою для короткотривалих сценаріїв використання (протягом шкільного уроку).

Клавіатура Telegram

Відображення клавіатури-підказки з дозволеними відповідями пов'язане (в бібліотеці `PyTelegramBotAPI`) зі створенням спеціального об'єкта класу `telebot.types.ReplyKeyboardMarkup`. Оскільки при наданні неочікуваної відповіді бот намагається підказати користувачеві дозволені варіанти відповідей та повторити запитання ще раз, видається доцільним зберігати (кешувати) останній створений клавіатурний об'єкт в сесії користувача. В разі необхідності повтору, не буде необхідності створювати нові клавіатурні об'єкти – достатньо використати закешований. Наступні функції реалізують згадане кешування:

Таблиця 8

Функція	Опис
init_keyboard(session, *buttons)	Створює об'єкт Telegram клавіатури та зберігає (кешує) його в сесії. Назви варіантів передаються параметром *buttons.
find_keyboard(session)	Відшукує та повертає закешований об'єкт Telegram клавіатури
find_buttons(session)	Відшукує та повертає закешовані назви варіантів відповідей.
clear_keyboard(session)	Видаляє з наданої сесії закешований об'єкт Telegram клавіатури

3.3. Опис роботи чат-боту для тестування

Особливість розробленого тесту полягає в тому, що учень відразу бачить свої помилки і це стає мотиваційним поштовхом для знаходження прогалин у знаннях і дає можливість доопрацювати ці пропуски. Окрім того, є можливість перегляду усіх результатів після проходження тестування кожного окремого учня або класу в цілому.

На даному етапі розробки, чат-бот пропонує провести тестування учнів з предмету «Інформатика. 5 клас НУШ».

Учень відкриває чат-бот і бачить одну кнопку «Start», яку необхідно натиснути для початку проходження тестування (рис. 3.1).

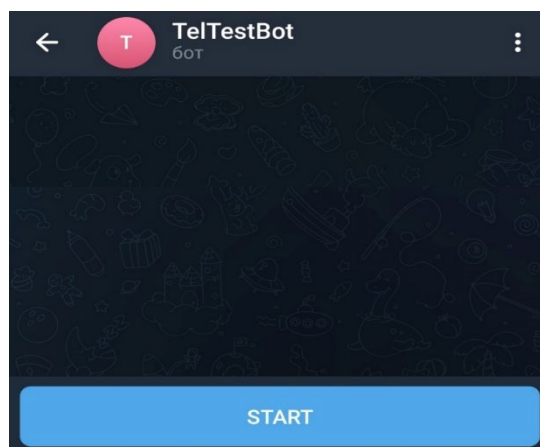


рис. 3.1

Після запуску, чат-бот пропонує ввести свої дані (прізвище, ім'я) (рис.3.2), а далі обрати одну із тем, які вивчаються протягом усього курсу «Інформатика. 5 клас» (рис. 3.3), а саме:

1. Інформаційні процеси;
2. Комп'ютерні мережі;
3. Текстові документи та презентації;
4. Алгоритми та програми.

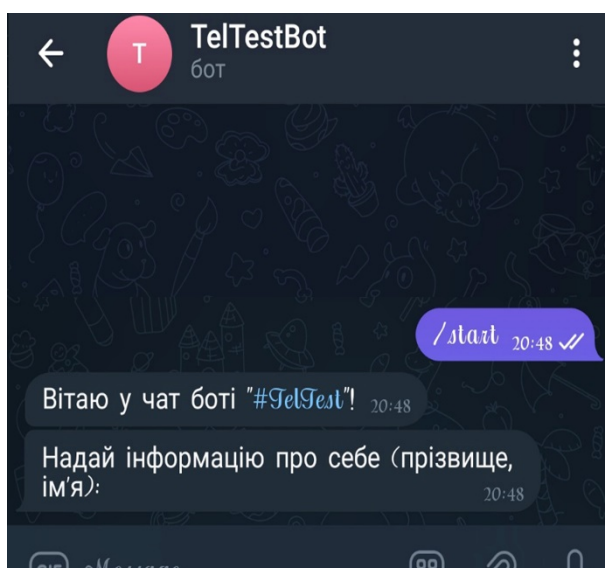


рис. 3.2

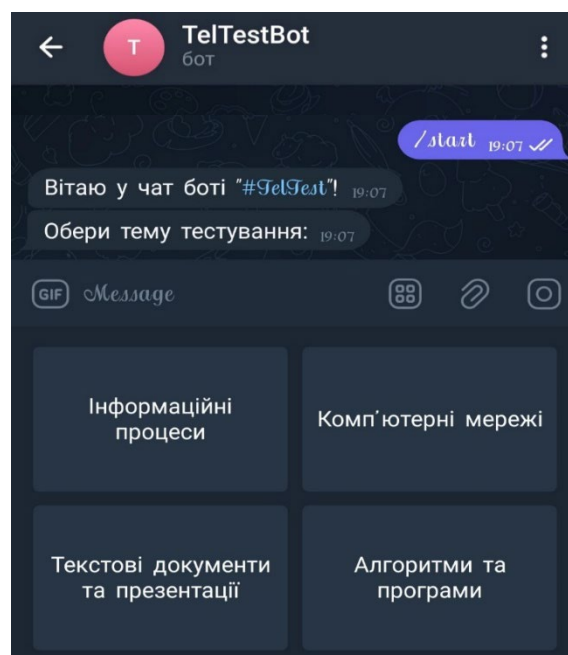


рис. 3.3

Таким чином можна провести підсумкове оцінювання знань здобувача освіти чи екстернату, або ж перевірити вивчення окремих конкретних тем курсу.

Тож після обрання однієї з тем починається безпосереднє тестування, коли після питання, з'являються варіанти відповіді. При чому, дана система налаштована за принципом закритої форми із однією правильною відповіддю. Якщо учень відповідає правильно, чат-бот підмічає вірну відповідь і пропонує наступне питання (рис. 3.4). Та навіть при неправильному виборі варіанту, з'являється наступне запитання, але програма фіксує цей варіант як неправильний (рис. 3.5) і в кінці тестування підсумовує кількість правильних та хибних результатів, виводячи це співвідношення на екран (рис. 3.6).

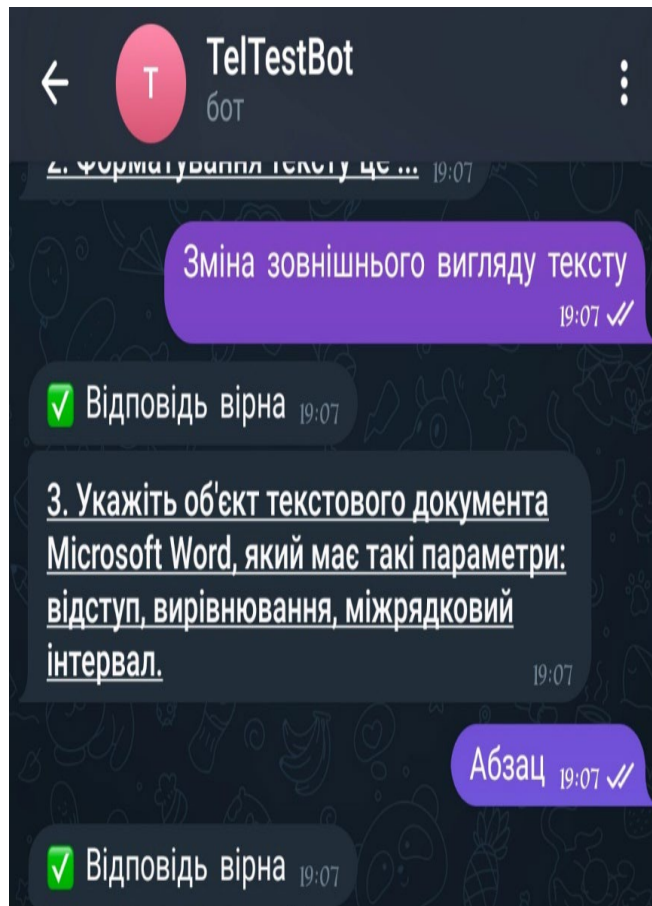


рис. 3.4

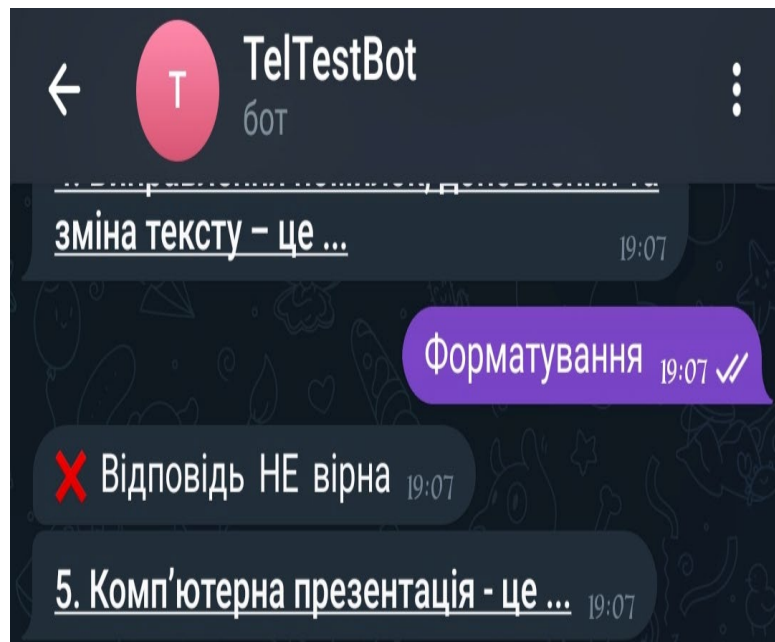


рис. 3.5

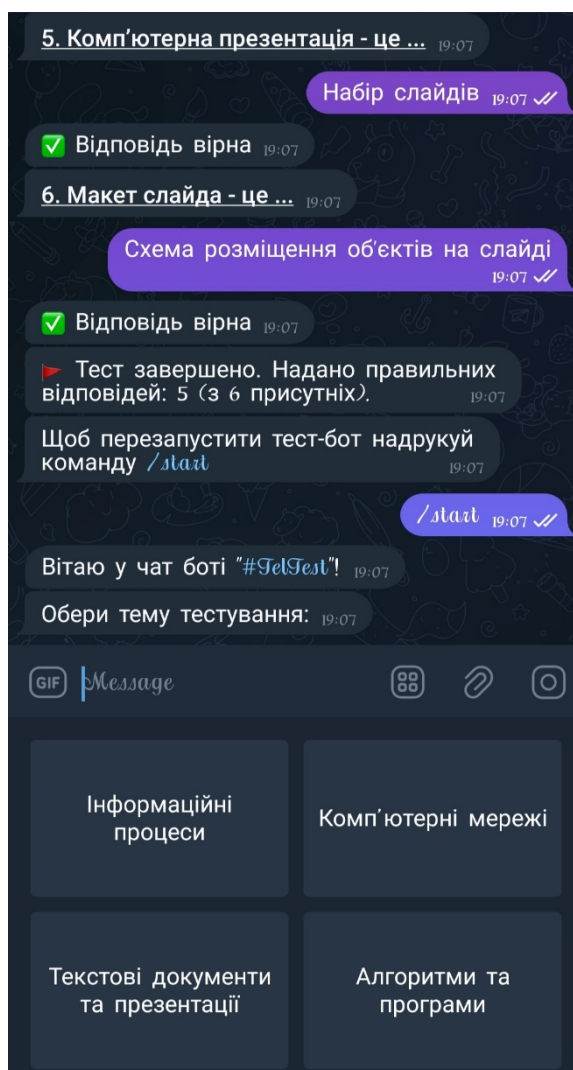


рис. 3.6

3.4. Апробація чат-боту

Для апробації роботи даного ресурсу, було запропоновано скласти тематичне оцінювання для учнів 5-А класу з вивченої теми «Текстовий редактор. Редактор презентацій». Цей клас було обрано не випадково, оскільки у ньому є учні які навчаються за кордоном і обрали екстернатну форму навчання, але, заради експерименту, також взяли участь у тестуванні.

Тож учні, після отримання посилання на чат-бот, доєдналися з різних пристроїв, на яких був завантажений та встановлений Telegram: телефони, планшети, ноутбуки, комп'ютери. Система запустила на кожному із гаджетів без перешкод.

Зайшовши в чат-бот і ввівши свої дані, діти почали проходити тест. Оскільки запитань було не багато, тестування пройшло досить швидко.



Після проходження тестування усіма учнями класу, відразу чи у будь-який зручний момент за необхідності, можна переглянути результати кожного окремого учня (рис.3.7, 3.8), що дозволяє оцінити їхню роботу навіть після уроку, не витрачаючи на це час протягом уроку.

Тож стає очевидним, що даний метод оцінювання є досить простим у користуванні, але не менш ефективним при оцінці, навіть для тих учнів, які не навчаються у стінах школи. І це стає надійним помічником для викладачів, особливо враховуючи факт збереження великої кількості часу не лише при проведенні самого тематичного оцінювання, а і при обробці результатів тестування.

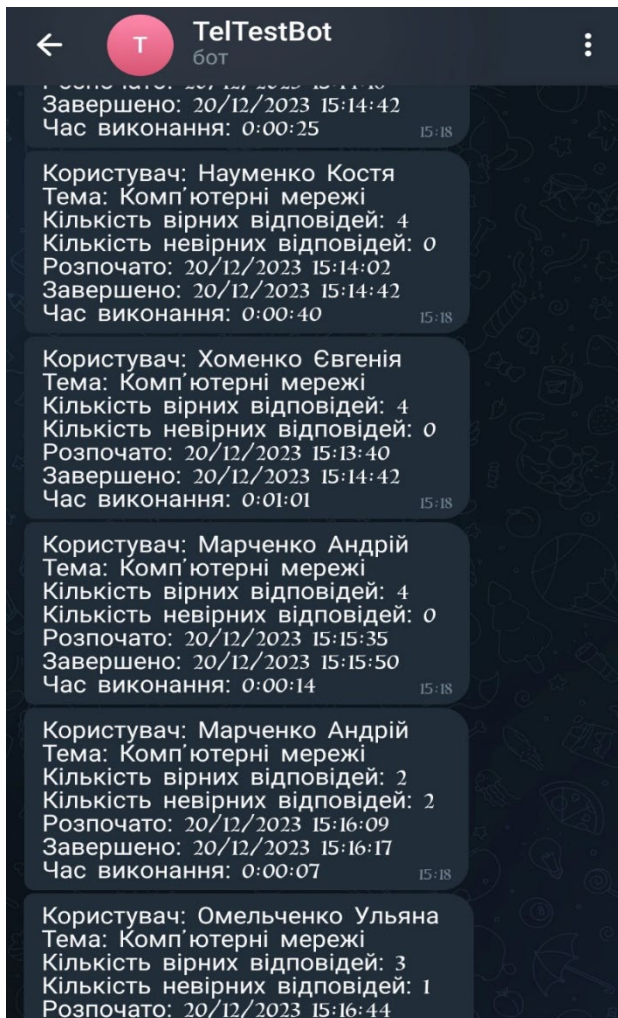


рис. 3.7

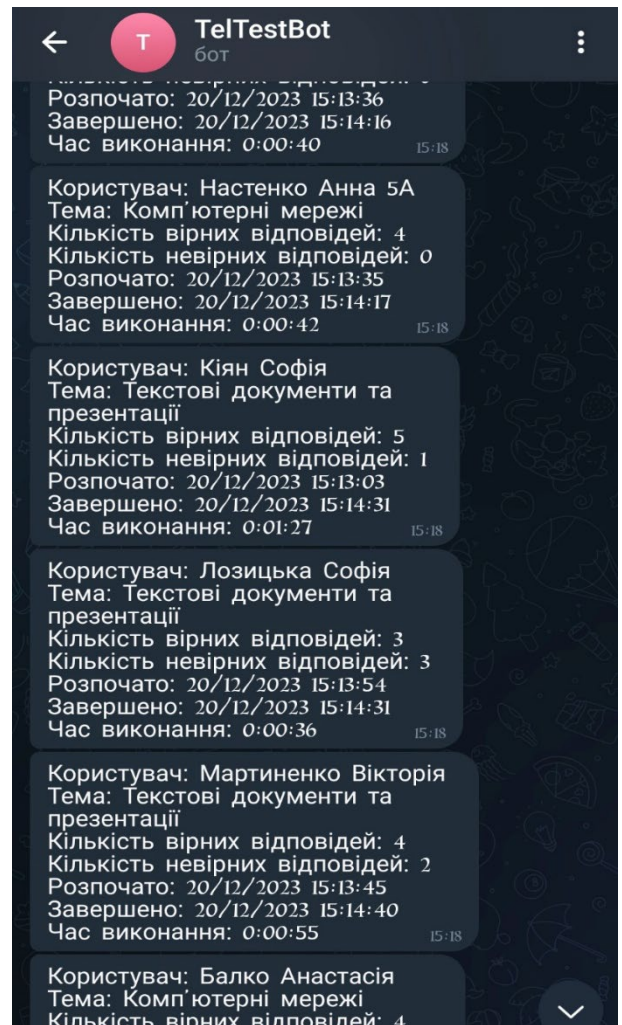


рис. 3.8

Висновок до розділу 3

Для створення чат-боту для тестування було обрано мову програмування Python за ітеративною моделлю, що дозволяє розробити коректно працюючий прототип з мінімальним функціоналом, але тим не менш достатнім, для запуск у роботу на уроках Інформатики.

Під час роботи було створено сценарії, які бот повинен виконувати у відповідь на вибір користувачем команди. Інтерфейс додатку абсолютно простий і зрозумілий для кожного.

Застосування функціоналу введення особистих даних дає можливість відслідкувати результативність кожного окремого здобувача освіти, навіть через певний проміжок часу, що забезпечує збереження великої кількості часу та ресурсів вчителя, уникаючи затрат на перевірку і оцінювання тестування.

ВИСНОВОК

Сучасна освіта потребує не лише передачі знань, але і використання новітніх технологій для поліпшення навчального процесу. Однією з таких інновацій є використання додатку для проведення тестів на уроках Інформатики. Такі тестування можуть бути не лише гарним інструментом перевірки знань учнів, але й незамінним помічником в умовах дистанційного, змішаного чи екстернатного навчання, оскільки дозволяють досить швидко оцінити результати навчання і внести корективи у викладку матеріалу з урахування цих результатів. Не витрачаючи великої кількості на розробку, проведення, а потім ще й аналіз контрольних запитань з відповідних тем програми, викладач отримує ледь не миттєвий висновок про проведenu роботу, що дає змогу використати зекономлений час на доопрацювання прогалин у навчальному процесі, а це, в свою чергу, призводить до підвищення рівня засвоєння матеріалу в цілому та здобуття нових додаткових навичок в окремих випадках. Тому розробка такого додатку є актуальним питанням на сьогоднішній день.

Розробка системи тестування у месенджері Telegram була не випадково обрана, тому як такий підхід забезпечує доступ як викладачів, так і учнів із будь-якого наявного гаджету, на якому встановлений дана програма і є доступ до інтернету. Враховуючи момент, що не в усіх дітей вдома є ноутбук чи комп'ютер, це стає основною перевагою при обранні платформи розробки.

Із загалу доступних бібліотек взаємодії з Telegram було обрано саме PyTelegramBotAPI, оскільки вона є лише дуже тонкою «обгорткою» відповідних сервісів. Згідно документації PyTelegramBotAPI надає можливість створення ботів двох режимів роботи: polling та web-hooks. В першому випадку (polling) запущений скрипт телеграм бота регулярно відправляє запити до сервісів Telegram для отримання нових повідомлень, що надходять користувачу. Для реального використання в шкільних умовах набагато

зручнішим є перший режим функціонування, оскільки не вимагає присутності сервера з доступною (виділеною) IP адресою.

Через це для реалізації даного проєкту було обрано ітеративну модель та вирішено почати розробку зі створення «швидкого» коректно працюючого прототипу з мінімальним функціоналом.

Основною перевагою додатку стає миттєвий результат учнів, що дозволяє їм зрозуміти свої сильні та слабкі сторони. Викладач також має можливість швидко аналізувати результати та адаптувати подальше навчання, вводячи корегувальні елементи для покращення засвоєння матеріалу.

Застосування додатків для проведення тестів на уроках інформатики – це ефективний спосіб підвищити якість навчання та зробити процес більш інтерактивним та індивідуалізованим. Це також дозволяє вчителям ефективно використовувати час та аналізувати успішність учнів, щоб надати їм оптимальну підтримку та допомогу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Асоянц, П.Г., Чекаль, Г.С., Сердюков, П.І. та ін. Основи методики створення і застосування комп'ютерних програм у навчанні іноземних мов. Київ: КДППМ. 2013.
2. Полат Є. С., Бухаркіна М. Ю., Моїсєєва В. М., Петров О. Є. Сучасні педагогічні та інформаційні технології у системі освіти : навч. посіб. / під ред. Є. С. Полат. 4-е вид. М.: Видавничий центр "Академія". 2009. 272 с.
3. Жук, Ю.О. Деякі психолого-педагогічні проблеми використання засобів нових інформаційних технологій у навчальному процесі середнього закладу освіти. *Комп'ютер у школі та сім'ї*. 2018. с. 7–9.
4. Brown, H.D. *Teaching by principles: an interactive approach to language pedagogy*. USA: Prentice Hall Regents. 2014.
5. Brown, K. *Using new technology in the classroom*. Sydney: Macquarie University. 2018.
6. Hardisty, D. & Windeatt, S. *CALL*. Hong Kong: Oxford University Press.. 2017.
7. Littlewood, W. *Foreign and second language learning*. Cambridge: Cambridge University Press..2011.
8. Рахімов Б. К. Оптимальний алгоритм взаємодії інформаційного ресурсу з мобільними додатками. *Молодий вчений*, 2016. С. 158–162.
9. Боднарчук І. О. Експертна система проектування архітектури програмного забезпечення. 2013. 210 с.
10. Петренко О. О. Порівняння типів архітектури систем сервісів / О. О. Петренко // *Системні дослідження та інформаційні технології*, 2015. С. 48-62.
11. Брайн Харді, Білл Філіпс Android. Програмування для професіоналів. 2016. 640 с.
12. Ерік Фрімен. Елізабет Робсон. Head First. Програмування на JavaScript. Харків: Фабула, 2022. С. 672.

13. Сухов К. NODE.JS. Путівник по технології. 2017. С. 416
14. Тлумачний словник з інформатики / Г.Г. Півняк, Б.С. Бусигін, М.М. Дівізінюк та ін. Дніпропетровськ: Нац. гірнич. ун-т. 2010. С. 600.
15. Ерік Маттес. Пришвидшений курс Python. Львів: Видавництво Старого Лева. 2021. С. 600.
16. Fabrizio Romano, Gaston C. Hillar, Arun Ravindran. Learn Web Development with Python Packt Publishing, 2018. P. 796.
17. Л.О. Кухар, В.П. Сергієнко. Конструювання тестів. Курс лекцій: навч. посіб. Луцьк. 2010. С. 182.
18. Моніторинг рівня навчальних досягнень з використанням Інтернет-технологій : монографія / за ред. В.Ю. Бикова, Ю.О. Жука. Київ : Педагогічна думка, 2008. С. 128.
19. Тестові технології оцінювання ключових і предметних компетентностей учнів основної і старшої школи : монографія / за ред. О.І. Ляшенко. Київ: Педагогічна думка, 2014. С. 168.
20. Азарян А. А., Карабут Н. О. Основи алгоритмізації та програмування: Навчальний посібник. Кривий Ріг: ОктанПринт, 2014. С. 308.
21. Алгоритми та структури даних / уклад. О. В. Щербаков, Ю. Е. Парфьонов, В. М. Федорченко. Харків: ХНЕУ ім. С. Кузнеця, 2017. С. 58.

Інтернет джерела

22. Moodle – Advantages and Disadvantages. URL: <https://www.beamstacks.com/blog/moodle-advantages-and-disadvantages-learning-system/> (дата звернення: 15.11.2023).
23. Marcin Frąckiewicz. Посібник для початківців зі створення власного чат-бота, 2023. URL: <https://ts2.space/uk/%D0%BF%D0%BE%D1%81%D1%96%D0%B1%D0%BD%D0%B8%D0%BA-%D0%B4%D0%BB%D1%8F-%D0%BF%D0%BE%D1%87%D0%B0%D1%82%D0%BA%D1%96%D0%B2%D1%86%D1%96%D0%B2-%D0%B7%D1%96->

[%D1%81%D1%82%D0%B2%D0%BE%D1%80%D0%B5%D0%BD%D0%BD/#gsc.tab=0](#) (дата звернення: 20.11.2023).

24. Wezom. Стаття «Методологія розробки програмного забезпечення: дивимось на роботу команди зсередини», 2021 URL: <https://wezom.com.ua/ua/blog/metodologija-razrabotki-programmnogo-obespechenija> (дата звернення: 20.11.2023).

25. Кемерон Маккензі, Java URL: <https://www.theserverside.com/definition/Java> (дата звернення: 05.12.2023).

26. Ruby URL: <https://www.geeksforgeeks.org/ruby-introduction-to-multithreading/?ref=rp>. (дата звернення: 05.10.2023).

27. Functional programming. URL: https://en.wikipedia.org/wiki/Functional_programming (дата звернення: 27.09.2023).

28. Яковенко А.В. Основи програмування Python. Частина 1. / Уклад.: А. В.Яковенко.–К. КПІ м.. Сікорського, 2018, 195 с. URL: <https://ela.kpi.ua/bitstream/123456789/25111/1/Python.pdf> (дата звернення: 30.11.2023).

29. Refactoring.Guru. Ітератор. URL: <https://refactoring.guru/uk/design-patterns/iterator> (дата звернення: 29.11.2023).

30. Офіційна документація pyTelegramBotAPI. // eternnoir/pyTelegramBotAPI. URL: <https://github.com/eternnoir/pyTelegramBotAPI> (дата звернення: 05.12.2023).

31. Список редакторів для мови Python. URL: <http://wiki.python.org/moin/PythonEditors> (дата звернення: 12.11.2023).