ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра інженерії програмного забезпечення

Пояснювальна записка

до бакалаврської кваліфікаційної роботи
на ступінь вищої освіти бакалавр

на тему: **«РОЗРОБКА CHAT-BOT ДЛЯ ОПТИМІЗАЦІЇ ГОСПОДАРСЬКОЇ РОБОТИ ГУРТОЖИТКУ МОВОЮ PYTHON»**

Виконала: студентка 4 курсу, групи ПД-44
спеціальності 121 Інженерія програмного забезпечення
(шифр і назва спеціальності)
Тертична Ю.М.
(прізвище та ініціали)
Керівник Негоденко О.В.
(прізвище та ініціали)
Рецензент
(прізвище та ініціали)
Нормоконтроль
(прізвище та ініціали)

Київ – 2021

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ
Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення
Ступінь вищої освіти - «Бакалавр»
Спеціальність -121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____Негоденко О.В.
«____»_____2021 року

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ
ТЕРТИЧНА ЮЛІЯ МИХАЙЛІВНА

1.Тема роботи: «Розробка chat-bot для оптимізації господарської роботи гуртожитку мовою Python.»

Керівник роботи: Негоденко Олена Василівна, доцент кафедри ІПЗ
затверджені наказом вищого навчального закладу від — «12» березня 2021 року № 65.
2. Строк подання студентом роботи 1.06.2021

3. Вхідні дані до роботи:

3.1. Вибір мови програмування

3.2. Алгоритм дії бота

3.3. Aiogram Bot API

3.4. Науково-технічна література, пов’язана з розробкою ботів.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити).

4.1. Аналіз обов’язків розроблюваного бота

4.2. Аналіз та порівняння існуючих прототипів

4.3. Дослідження програмних засобів для розробки бота

4.4. Розробити функціонал створеного бота

5. Перелік графічного матеріалу

5.1.1. Популярність телеграм-ботів

5.1.2. Переваги та недоліки найвідоміших месенджерів

5.1.3. Огляд можливостей розробленого бота

5.1.4. Апробація результатів досліджень

6. Дата видачі завдання 19.04.2021

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Підбір науково-технічної літератури	19.04.21 – 22.04.21	Виконано
2	Аналіз існуючих аналогів	22.04.21 – 23.04.21	Виконано
3	Дослідження програмних засобів	23.04.21 – 01.05.21	Виконано
4	Моделювання об’єкту проектування	01.05.21 – 05.05.21	Виконано
5	Розробка функціоналу застосунку	05.05.21 – 07.05.21	Виконано
6	Вступ, висновки, реферат	07.05.21 – 08.05.21	Виконано
7	Розробка презентації застосунку	08.05.21 – 24.05.21	Виконано
8	Попередній захист роботи	25.05.21	

Студент

Керівник роботи

РЕФЕРАТ

Текстова частина бакалаврської роботи 59с., 10 рис., 58 джерел.

Ключеві слова: PyCharm, Python, Aiogram, Telegram, чат-бот.

Об'єкт дослідження – методи та засоби створення чат-ботів на базі платформи Telegram.

Предмет дослідження – телеграм-бот для оптимізації господарської діяльності в гуртожитку.

Мета роботи – розробка комплексного Telegram-бота для оптимізації господарської діяльності гуртожитку на мові Python.

Методи дослідження – методи теорії інформації, методи структурного аналізу і проектування, методи розробки програмного забезпечення, методи тестування, валідації та верифікації програмного забезпечення.

Наукова новизна даної роботи полягає в наступному:

1. Розроблено алгоритм для подання і обробки господарських заявок гуртожитку.
2. Встановлено, що зв’язка API Aiogram з мовою програмування Python, показує найкращі результати по швидкодії та використанню ресурсів комп’ютера.
3. На основі результатів проведених досліджень, був розроблений авторський бот для оптимізації господарської діяльності гуртожитку на мові Python.

В роботі проведено аналіз предметної області. В результаті аналізу було визначено основні потреби користувачів. Проаналізовано можливості середовища розробки PyCharm, мови програмування Python. Розроблено логіку практичних завдань та загальну концепцію представлення інформації для користувачів.

Галузь використання – бота може використовувати будь-яка людина, яка є мешканцем гуртожитку.

ЗМІСТ

ВСТУП

В сучасному світі популярність різного роду месенджерів, поштових клієнтів, чат – клієнтів і великого різноманіття подібних рішень важко недооціни.

В середньому, у кожної другої людини на комп'ютері або смартфоні встановлений той чи інший месенджер, який може нести в собі не тільки функцію спілкування, але й більш масштабні функції, реалізовані, наприклад, за допомогою ботів різної направленості.

Явище ботів в месенджерах набуває все більшої актуальності за рахунок зручності використання подібних рішень, адже вони не вимагають переходити на зовнішні ресурси, встановлювати додаткове програмне забезпечення, тощо.

З огляду на високу актуальність, кількість різноманітних ботів в месенджерах, зокрема, наприклад, в месенджері телеграм, росте кожного дня і набуває все більшої серйозності в масштабах функціональних можливостей.

Виходячи з усього вищесказаного – важко недооцінити тематику даної роботи.

Також метою роботи є використання методів та засобів створення ботів на базі платформи телеграм для проектування і реалізації телеграм-бота для оптимізації господарської діяльності гуртожитку.

Для досягнення поставленої мети слід виконати наступні завдання:

1. Проаналізувати предметну область
    - провести огляд месенджеру Телеграм;
    - проаналізувати поняття ботів;
    - проаналізувати поняття прикладного інтерфейсу програмування;
2. Обрати засоби реалізації
    - провести огляд мови програмування;
    - провести огляд PostgreSQL;

- провести огляд AWS;

3. Спроектувати і реалізувати телеграм-бота для оптимізації господарської діяльності гуртожитку
    - побудувати діаграму варіантів використання;
    - побудувати структурну діаграму;
    - оглянути основні механізми роботи;
    - провести тестування;
    - написати керівництво користувача.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Платформа телеграм

Telegram – це безкоштовне програмне забезпечення для обміну миттєвими повідомленнями (IM), що працює на різних платформах, і програма для обміну миттєвими повідомленнями. Послуга також надає наскрізні зашифровані відеодзвінки, [15] VoIP, спільний доступ до файлів та кілька інших функцій. Спочатку він був запущений для iOS 14 серпня 2013 року та Android у жовтні 2013 року. Сервери програм Telegram розповсюджуються по всьому світу для зменшення навантаження на дані, в той час як операційний центр в даний час базується в Дубаї. [16] [17] [18] [19]] Різні клієнтські програми Telegram доступні для настільних та мобільних платформ, включаючи офіційні додатки для Android, iOS, Windows, macOS та Linux, а також для Windows Phone, що припинено. Також є офіційний веб-інтерфейс та численні неофіційні клієнти, які використовують протокол Telegram. Усі офіційні компоненти Telegram є відкритими, [20] за винятком сервера, який є закритим та власним. [13]

Telegram забезпечує наскрізні зашифровані голосові та відеодзвінки [21] та додаткові наскрізні зашифровані "секретні" чати. Хмарні чати та групи зашифровані між програмою та сервером, тому Інтернет-провайдери та інші сторонні особи в мережі не можуть отримати доступ до даних, але сервер Telegram має ключ дешифрування. Користувачі можуть надсилати текстові та голосові повідомлення, анімовані наклейки, здійснювати голосові та відеодзвінки, а також ділитися необмеженою кількістю зображень, документів (2 ГБ на файл), місць розташування користувачів, контактів та музики. У січні 2021

року Telegram перевищив 500 мільйонів активних користувачів щомісяця. [22] Це було найбільш завантажуване додаток у світі у січні 2021 року. [23]

Telegram був запущений в 2013 році братами Миколою та Павлом Дуровими. Раніше пара заснувала російську соціальну мережу VK, яку вони залишили в 2014 році після того, як її захопили союзники президента Путіна. Павло Дуров продав свій залишився пакет акцій у VK і залишив Росію після опору урядовому тиску [24]. Микола Дуров створив протокол MTProto, який є основою месенджера, тоді як Павло Дуров надавав фінансову підтримку та інфраструктуру через свій фонд Digital Fortress [25]. Telegram Messenger заявляє, що її кінцевою метою є не отримання прибутку [26] [27], але в даний час вона не структурована як некомерційна організація [28].

Telegram зареєстрований як англійський LLP [29], так і американський LLC. [30] Він не розкриває, де орендує офіси чи які юридичні особи використовує для їх оренди, посилаючись на необхідність "захистити команду від зайвого впливу" та захистити користувачів від державних запитів на дані [31]. Павло Дуров повідомив, що головний офіс служби знаходився в Берліні, Німеччина, з 2014 року [32] до початку 2015 року, але переїхав до різних юрисдикцій після того, як не зміг отримати посвідку на проживання для всіх членів команди. Після того, як Павло Дуров залишив Росію, він, як кажуть, переїжджає з країни в країну з невеликою групою комп'ютерних програмістів, що складається з 15 основних членів [24] [34]. Згідно з повідомленнями преси, Telegram мав співробітників у Санкт-Петербурзі. [33] В даний час команда Telegram знаходиться в Дубаї. [35]

У жовтні 2013 року Telegram оголосив, що має 100 000 активних користувачів щодня [25]. 24 березня 2014 року Telegram оголосив, що досяг щомісяця 35 мільйонів користувачів та 15 мільйонів активних користувачів щодня [36]. У жовтні 2014 року плани урядового нагляду Південної Кореї спонукали багатьох її громадян перейти на Telegram. [32] У грудні 2014 року

Telegram оголосила, що має 50 мільйонів активних користувачів, щоденно генеруючи 1 мільярд повідомлень, і що мільйон нових користувачів щотижня підписуються на її сервіс [37], за 5 місяців трафік подвоївся за 2 мільярди щоденних повідомлень. [38] У вересні 2015 року Telegram оголосив, що додаток налічує 60 мільйонів активних користувачів і щодня передає 12 мільярдів повідомлень. [39]

У лютому 2016 року Telegram оголосила, що має 100 мільйонів активних користувачів щомісяця, причому 350 000 нових користувачів підписуються щодня, щодня передаючи 15 мільярдів повідомлень [40]. У грудні 2017 року Telegram охопив 180 мільйонів активних користувачів щомісяця. [35] У березні 2018 року Telegram охопив 200 мільйонів активних користувачів щомісяця. [41] 14 березня 2019 року Павло Дуров заявив, що «3 мільйони нових користувачів підписалися на Telegram протягом останніх 24 годин» [42]. Дуров не вказав, що спонукало цю повеню нових реєстрацій, але цей період відповідав тривалому технічному відключенню Facebook та його сімейство додатків, зокрема Instagram. [43]

За даними Комісії з цінних паперів та бірж США, кількість щомісячних користувачів Telegram станом на жовтень 2019 року становить 300 мільйонів людей у всьому світі. [44]

24 квітня 2020 року Telegram оголосив, що охопив 400 мільйонів активних користувачів щомісяця. [22] 8 січня 2021 року Дуров повідомив у своєму повідомленні в блозі, що Telegram перебуває на рівні "близько 500 мільйонів" активних користувачів щомісяця [45].

1.2 Телеграм-боти

Інтернет-бот, веб-робот, робот або просто бот – це програмний додаток, який запускає автоматизовані завдання (скрипти) через Інтернет. [1] Зазвичай боти виконують прості та повторювані завдання набагато швидше, ніж це могло б зробити людина. Найбільш широко боти використовуються для веб-сканування, в якому автоматизований скрипт отримує, аналізує та зберігає інформацію з веб-серверів. Більше половини всього веб-трафіку генерується ботами. [2]

Намагання веб-серверів обмежити роботи ботів різняться. Деякі сервери мають файл robots.txt, який містить правила, що регулюють поведінку ботів на цьому сервері. Будь-якому боту, який не дотримується правил, теоретично може бути відмовлено у доступі до веб-сайту, що зазнає впливу, або видалення з нього. Якщо у розміщеному текстовому файлі немає пов'язаної програми / програмного забезпечення / програми, то дотримання правил є абсолютно добровільним. Неможливо застосувати правила або забезпечити, щоб творець або реалізатор бота прочитав або визнав файл robots.txt. Деякі боти "хороші" – напр. павуки в пошукових системах – тоді як інші використовуються для здійснення шкідливих атак, наприклад, на політичні кампанії. [2]

Деякі боти спілкуються з користувачами Інтернет-служб за допомогою миттєвих повідомлень (IM), Інтернет-ретрансляційного чату (IRC) або інших веб-інтерфейсів, таких як Facebook-боти та Twitter-боти. Ці чат-боти можуть дозволити людям задавати запитання простою англійською мовою, а потім формулювати відповідь. Такі боти часто можуть обробляти інформацію про погоду, інформацію про поштовий індекс, спортивні результати, конвертацію валют чи інших одиниць тощо [потрібне цитування] Інші використовуються для розваг, такі як SmarterChild на AOL Instant Messenger та MSN Messenger.

Додатковою роллю бота IRC може бути прослуховування каналу розмови, коментування певних фраз, виголошених учасниками (на основі відповідності шаблону). Це іноді використовується як служба довідки для нових користувачів або для цензури нецензурної лайки.

Боти для соціальних мереж – це набори алгоритмів, які беруть на себе обов'язки повторюваних наборів інструкцій з метою встановлення послуги або зв'язку між користувачами соціальних мереж. Серед різноманітних конструкцій мережевих ботів найпоширенішими є боти чату, алгоритми, розроблені для спілкування з користувачем-людиною, та соціальні боти, алгоритми, що імітують поведінку людини, щоб спілкуватися із шаблонами, подібними до схем користувача. Історію соціального боттингу можна простежити до Алана Тьюрінга в 1950-х роках та його бачення розробки наборів навчальних програм, затверджених тестом Тьюрінга. У 1960-х Джозеф Вайценбаум створив ELIZA, комп'ютерну програму для обробки природних мов. вважається раннім показником алгоритмів штучного інтелекту. Комп’ютерні програмісти ELIZA надихнули розробляти програми, що відповідають завданням, які можуть відповідати шаблонам поведінки їх набору інструкцій. В результаті обробка природної мови стала фактором, що впливає на розвиток штучного інтелекту та соціальних ботів. І оскільки інформація та думки бачать поступове масове поширення на веб-сайтах соціальних мереж [3], інноваційні технологічні досягнення досягаються за тією ж схемою.

Звіти про політичне втручання в останні вибори, включаючи загальні вибори у США та Великобританію у 2016 році [4], встановили, що поняття ботів є більш розповсюдженим через етику, яка оскаржується між дизайном бота та дизайнером бота. За словами Еміліо Феррари, комп’ютерного науковця з Університету Південної Каліфорнії, який звітує про зв’язок ACM, [5] відсутність ресурсів, доступних для здійснення перевірки фактів та перевірки інформації,

призводить до великих обсягів неправдивих повідомлень та заяв щодо них ботів на платформах соціальних медіа. У випадку з Twitter більшість із цих ботів запрограмовані з можливостями фільтрування пошуку, які націлюються на ключові слова та фрази, що надають перевагу політичним програмам, а потім ретвітують їх. Хоча увага ботів запрограмована на поширення неперевіреної інформації на всіх платформах соціальних медіа [6], це проблема, з якою стикаються програмісти у зв'язку з ворожим політичним кліматом. Ефект бота - це те, що Ferrera повідомляє про соціалізацію ботів та користувачів, що створює уразливість до витоку особистої інформації та поляризуючих впливів поза етикою коду бота. Це підтверджує Гіллорі Крамер у своєму дослідженні, де він спостерігає поведінку емоційно нестабільних користувачів та вплив ботів на користувачів, змінюючи сприйняття реальності.

Було багато суперечок щодо використання ботів у функції автоматизованої торгівлі. Аукціонний веб-сайт eBay розпочав судові дії, намагаючись придушити сторонні компанії використовувати ботів для обходу їх веб-сайту в пошуках вигідних пропозицій; такий підхід дав негативні наслідки на eBay і привернув увагу подальших ботів. Заснована у Великобританії біржа ставок Betfair побачила настільки великий обсяг трафіку, який надходить від ботів, що вона запустила API WebService, націлений на програмістів-ботів, за допомогою якого вона може активно керувати взаємодією ботів.

Відомо, що ферми ботів використовуються в Інтернет-магазинах додатків, таких як Apple App Store та Google Play, для маніпулювання позиціями [7] або для підвищення позитивних оцінок / відгуків. [8]

Чат-ботом є стрімко зростаюча, доброякісна форма інтернет-бота. З 2016 року, коли Facebook Messenger дозволив розробникам розміщувати чат-боти на своїй платформі, спостерігається експоненціальне зростання їх використання лише на цьому форумі. За перші шість місяців для Messenger було створено 30

000 ботів, які до вересня 2017 року зросли до 100 000 [9]. Аві Бен Езра, технічний директор SnatchBot, заявив Forbes, що дані використання їхньої платформи для створення чат-ботів вказують на найближчу перспективу, що заощаджує мільйони годин людської праці, оскільки "живий чат" на веб-сайтах замінюється ботами [10].

Компанії використовують Інтернет-ботів для збільшення зацікавленості в Інтернеті та спрощення спілкування. Компанії часто використовують ботів для скорочення витрат, замість того, щоб наймати людей для спілкування зі споживачами, компанії розробили нові способи бути ефективними. Ці чат-боти використовуються для відповіді на запитання клієнтів. Наприклад, Domino's розробив чат-бот, який може приймати замовлення через Facebook Messenger. Чат-боти дозволяють компаніям розподіляти час своїх співробітників на більш важливі справи. [11]

Одним із прикладів зловмисного використання ботів є координація та функціонування автоматизованої атаки на мережеві комп’ютери, наприклад атаки відмови в обслуговуванні ботнетом. Інтернет-боти або веб-боти також можуть бути використані для здійснення шахрайства із натисканням, і нещодавно вони з'явилися навколо ігор MMORPG як боти для комп'ютерних ігор. У наш час цей тип ботів також використовується у відеоіграх, таких як PUBG. Мобільні боти PUBG також відносяться до сімейства шкідливих ботів. Інша категорія представлена спам-ботами, інтернет-ботами, які намагаються спамувати велику кількість вмісту в Інтернеті, зазвичай додаючи рекламні посилання. Понад 94,2% веб-сайтів зазнали атаки ботів. [2]

1. Існують зловмисні боти (і бот-мережі) таких типів:
    - спам-боти, які збирають адреси електронної пошти зі сторінок контактів або гостьової книги;

- завантажуються програми, які висмоктують пропускну здатність, завантажуючи цілі веб-сайти;
- скребки веб-сайтів, які захоплюють вміст веб-сайтів і повторно використовують його без дозволу на автоматично згенерованих дверних сторінках;
- реєстраційні боти, які підписують певну електронну адресу для численних служб, щоб повідомлення про підтвердження заповнювали поштову скриньку та відволікали увагу від важливих повідомлень, що свідчать про порушення безпеки. [12]
- віруси та глисти;
- DDoS-атаки;
- ботнети, зомбі-комп'ютери тощо;
- спам-боти, які намагаються перенаправити людей на шкідливий веб-сайт, який іноді можна знайти в розділах коментарів або на форумах різних веб-сайтів;
- боти перегляду створюють підроблені подання. [13] [14]

2. Боти також використовуються для викупу більш затребуваних місць для концертів, зокрема, посередниками, що продають квитки [15]. Ці боти проходять процес придбання розважальних сайтів з продажу квитків та отримують кращі місця, відтягуючи стільки місць, скільки можна.
3. Боти часто використовуються в масових багатокористувацьких онлайнових рольових іграх для отримання ресурсів, для отримання яких інакше знадобився б значний час чи зусилля; це турбує більшість онлайн-ігрових економік.
4. Боти також використовуються для збільшення кількості переглядів відео на YouTube.

5. Боти використовуються для збільшення кількості трафіку в аналітичних звітах для вилучення грошей у рекламодавців. Дослідження, проведене компанією Comscore, показало, що більше половини оголошень, що відображаються в тисячах кампаній у період з травня 2012 року по лютий 2013 року, не відображалися для користувачів.
6. у 2012 р. журналіст Персі фон Ліпінський повідомив, що виявив мільйони ботів або переглянув або переглянув погляди на CNN iReport. CNN iReport тихо видалив мільйони переглядів з рахунку так званого суперзірки iReporter Кріса Моррoу. [17] Невідомо, чи доходи від реклами, отримані CNN від підроблених переглядів, колись повертались рекламодавцям.
7. Боти можуть використовуватися на форумах в Інтернеті для автоматичного розміщення запальних чи безглуздих дописів, щоб порушити форум і розсердити користувачів.

Найбільш широко використовувана техніка боротьби з ботами – це використання CAPTCHA, яка є формою тесту Тьюрінга, що використовується для розрізнення між користувачем людини та менш витонченим ботом на основі штучного інтелекту за допомогою графічно закодованого тексту, що читається людиною. Прикладами постачальників є Recaptcha та комерційні компанії, такі як Minteye, Solve Media та NuCaptcha. Однак капчі не є надійними у запобіганні ботам, оскільки їх часто можна обійти за допомогою комп’ютерного розпізнавання символів, дірок у безпеці та навіть передачі розв’язування капчі дешевим робітникам.

Взаємодія між людьми та соціальними роботами дуже поширена у повсякденному житті, наприклад, у школах, на робочих місцях, у відеоіграх та соціальних мережах [18]. Використання ботів має кілька переваг. Боти завжди

доступні, що забезпечує швидке та зручне обслуговування клієнтів. Боти можуть використовуватися для різних цілей, таких як чат / відповіді на запитання та реклама. Боти також зменшують витрати на робочу силу, що, в свою чергу, приносить більший прибуток компанії [5]

Компанії та клієнти можуть скористатися Інтернет-ботами. Інтернет-боти дозволяють клієнтам контактувати з компаніями, не потребуючи спілкування з людиною. KLM Royal Dutch Airlines створив чат-бот, який дозволяє клієнтам отримувати посадкові талони, нагадування про реєстрацію та іншу інформацію, необхідну для польоту. [11] Залучення клієнтів зросло з тих пір, як компанії розробили чат-боти, які можуть принести користь клієнтам.

Чат-боти використовуються щодня. Google Assistant і Siri вважаються формами чатових ботів. Google Assistant і Siri дозволяють людям задавати питання та отримувати відповіді за допомогою системи ШІ. Ці технологічні досягнення позитивно вплинули на повсякденне життя людей. [Цитування]

Більш нішевим варіантом використання інтернет-ботів є автоматичне придбання кросівок через Інтернет. Існує ряд різних ботів для придбання кросівок, які працюють на декількох різних веб-сайтах про кросівки. Часто людина використовує бота, щоб або отримати рідкісне / обмежене взуття, або отримати кілька пар взуття та перепродати їх, щоб отримати досить великий прибуток. [19]

У всьому світі боти викликають дві основні проблеми: ясність та особиста підтримка. Культурне походження людей впливає на спосіб спілкування з соціальними ботами. Багато людей вважають, що боти набагато менш розумні, ніж люди, і тому вони не гідні нашої поваги. [1]

Мін-Сун Кім запропонував п'ять проблем або питань, які можуть виникнути під час спілкування з соціальним роботом, і вони уникають шкоди

почуттів людей, мінімізують нав'язування, спростування з боку інших, проблеми з ясністю та наскільки ефективні їх повідомлення.]

Соціальні роботи також віддаляють від справжніх творінь людських стосунків. [1]

1.3 Telegram Bot API

У обчислювальних програмах інтерфейс прикладного програмування (API) – це інтерфейс, який визначає взаємодію між декількома програмними додатками або змішаними апаратно-програмними посередниками. [1] Він визначає види дзвінків або запитів, які можна зробити, спосіб їх здійснення, формати даних, які слід використовувати, домовленості, яких слід дотримуватися тощо. Він також може забезпечити механізми розширення, щоб користувачі могли розширити існуючі функціональні можливості різними способами та різною мірою. [2] API може бути повністю спеціальним, специфічним для компонента або розробленим на основі галузевого стандарту для забезпечення сумісності. Завдяки приховуванню інформації, API дозволяють модульне програмування, дозволяючи користувачам використовувати інтерфейс незалежно від реалізації.

Наразі посилання на веб-API є найпоширенішим використанням цього терміна. [3] Існують також API для мов програмування, бібліотек програмного забезпечення, комп’ютерних операційних систем та комп’ютерного обладнання. API виникли в 1940-х роках, хоча термін API з’явився лише в 1960-х і 70-х роках.

При побудові додатків API (інтерфейс програмування додатків) спрощує програмування, абстрагуючись від базової реалізації та виставляючи лише об'єкти або дії, необхідні розробнику. Хоча графічний інтерфейс для поштового клієнта може надавати користувачеві кнопку, яка виконує всі дії для отримання та виділення нових електронних листів, API для введення / виведення файлів

може дати розробнику функцію, яка копіює файл з одного місця в інше без вимагаючи, щоб розробник розумів операції з файловою системою, що відбуваються за кадром. [4]

Значення терміна API розширювалось за його історію. Спочатку він описав інтерфейс лише для програм, орієнтованих на кінцевих користувачів, відомих як прикладні програми. Це походження все ще відображається в назві "інтерфейс прикладного програмування". Сьогодні термін API є ширшим, включаючи також програмне забезпечення та навіть апаратні інтерфейси. [6]

Ідея API набагато старша за термін. Британські вчені-комп'ютеристи Уілкс і Вілер працювали над модульними бібліотеками програмного забезпечення в 1940-х роках для комп'ютера EDSAC. Їхня книга «Підготовка програм для електронного цифрового комп’ютера» містить першу опубліковану специфікацію API. Джошуа Блох стверджує, що Уілкс та Уілер "приховано" винайшли "API", оскільки це більше концепція, яка виявляється, ніж винаходить. [6]

Термін "інтерфейс прикладних програм" (без суфікса -ing) вперше зафіксовано в статті "Структури даних та техніки для віддаленої комп'ютерної графіки", представленій на конференції AFIPS у 1968 р. [8] [6] Автори цієї статті використовують термін для опису взаємодії програми - графічної програми в даному випадку - з рештою комп'ютерної системи. Послідовний інтерфейс програми (що складається з викликів підпрограми Fortran) мав на меті звільнити програміста від вирішення особливостей графічного пристрою відображення та забезпечити апаратну незалежність у разі заміни комп'ютера чи дисплея. [7]

Цей термін був введений у поле баз даних К. Дж. Дайтом [9] у роботі 1974 року, що називається "Реляційний та мережевий підходи: порівняння інтерфейсу прикладного програмування" [10]. API став частиною фреймворку ANSI / SPARC для систем управління базами даних. Цей фреймворк трактував інтерфейс

прикладного програмування окремо від інших інтерфейсів, таких як інтерфейс запиту. Професіонали баз даних у 1970-х роках спостерігали, що ці різні інтерфейси можна поєднувати; досить багатий інтерфейс програми може підтримувати й інші інтерфейси. [5]

Це спостереження призвело до API, які підтримували всі типи програмування, а не лише прикладне програмування. До 1990 року API був визначений просто як "набір послуг, доступних програмісту для виконання певних завдань" технологом Карлом Маламудом. [11]

Концепція API була знову розширена з початком веб-API. Дисертація Роя Філдінга "Архітектурні стилі та дизайн мережевих архітектур програмного забезпечення" в UC Irvine у 2000 р. Окреслила передачу представницького стану (REST) та описала ідею "мережевого інтерфейсу програмування програм", який Філдінг протиставив традиційній "бібліотеці" API. [12] Веб-API XML та JSON побачили широке комерційне прийняття, починаючи з 2000 року та продовжуючи з 2021 року.

Зараз веб-API є найпоширенішим значенням терміна API. [3] Вживаний таким чином, термін API має певне перекриття за значенням із термінами протокол зв'язку та виклик віддаленої процедури.

Семантична павутина, запропонована Тімом Бернерсом-Лі в 2001 році, включала "семантичні API", які переробляли API як відкритий, розподілений інтерфейс даних, а не як програмний інтерфейс поведінки. [13] Натомість запатентовані інтерфейси та агенти набули більшого поширення.

Інтерфейс бібліотеки програмного забезпечення є одним із типів API. API описує та прописує "очікувану поведінку" (специфікацію), тоді як бібліотека є "фактичною реалізацією" цього набору правил.

Один API може мати кілька реалізацій (або жодну, будучи абстрактною) у вигляді різних бібліотек, що мають один і той же інтерфейс програмування.

Відділення API від його реалізації може дозволити програмам, написаним однією мовою, використовувати бібліотеку, написану іншою. Наприклад, оскільки Scala та Java компілюються до сумісного байт-коду, розробники Scala можуть скористатися будь-яким Java API. [14]

Використання API може змінюватися залежно від типу мови програмування. API для процедурної мови, такої як Lua, може складатися в основному з базових підпрограм для виконання коду, маніпулювання даними або обробки помилок, тоді як API для об'єктно-орієнтованої мови, такої як Java, забезпечуватиме специфікацію класів та методів її класів. [15] [16]

Мовні прив’язки – це також API. Завдяки відображенню особливостей і можливостей однієї мови з інтерфейсом, реалізованим іншою мовою, прив’язка мови дозволяє використовувати бібліотеку чи послугу, написану однією мовою, при розробці на іншій мові. [17] Такі інструменти, як SWIG та F2PY, генератор інтерфейсу Fortran-to-Python, полегшують створення таких інтерфейсів. [18]

API також може бути пов'язаний із програмною структурою: структура може базуватися на декількох бібліотеках, що реалізують декілька API, але на відміну від звичайного використання API, доступ до поведінки, вбудованої в фреймворк, опосередковується шляхом розширення його вмісту новими класами підключений до самого фреймворку.

Більше того, загальний потік керування програмою може бути поза контролем абонента та в руках фреймворку шляхом інверсії управління або подібного механізму. [19] [20]

2 ВИБІР ЗАСОБІВ РЕАЛІЗАЦІЇ

2.1 Вибір мови програмування

Python – інтерпретована мова програмування високого рівня та загального призначення. Філософія дизайну Python наголошує на читабельності коду завдяки помітному використанню значних відступів. Його мовні конструкції та об’єктно-орієнтований підхід мають на меті допомогти програмістам написати чіткий логічний код для малих та великих проектів.

Python динамічно набирається та збирається сміття. Він підтримує декілька парадигм програмування, включаючи структуроване (зокрема, процедурне), об'єктно-орієнтоване та функціональне програмування. Python часто описують як мову, що включає батареї, завдяки своїй повній стандартній бібліотеці.

Гідо ван Россум почав працювати над Python наприкінці 1980-х років, як наступник мови програмування ABC, і вперше випустив її в 1991 році під назвою Python 0.9.0. [32] Python 2.0 був випущений в 2000 році і представив нові функції, такі як розуміння списків та система збору сміття з використанням підрахунку посилань, і був припинений з версією 2.7.18 у 2020 році [33]. Python 3.0 був випущений в 2008 році і був серйозним переглядом мови, яка не є повністю сумісною із зворотною суттю, і більша частина коду Python 2 не працює незміненою на Python 3.

Python стабільно входить до числа найпопулярніших мов програмування.

Python був задуманий наприкінці 1980-х Гвідо ван Россумом з Centrum Wiskunde & Informatica (CWI) в Нідерландах як спадкоємець мови програмування ABC, натхненної SETL, здатною обробляти винятки та взаємодіяти з Операційна система Amoeba. Його реалізація розпочалась у грудні 1989 р. Ван Россум взяв на себе виключну відповідальність за проект, як

провідний розробник, до 12 липня 2018 року, коли він оголосив про свої "постійні канікули", виконуючи обов'язки як "Доброзичливий диктатор за життя" Пітона, титул, який йому надала спільнота Пітона, щоб відобразити його тривалий час. строкове зобов'язання як головний керівник проекту. Зараз він ділиться своїм керівництвом як член наглядової ради з п'яти осіб. У січні 2019 року активні розробники ядра Python обрали Бретта Кеннона, Ніка Коглана, Барі Варшаву, Керол Віллінг та Ван Россума членами керівної ради з п’яти членів. З тих пір Гвідо ван Россум відкликав свою кандидатуру на посаду Керівної ради 2020 року.

Python 2.0 був випущений 16 жовтня 2000 року з багатьма основними новими можливостями, включаючи збирач сміття, що виявляє цикл, та підтримку Unicode.

Python 3.0 був випущений 3 грудня 2008 року. Це був серйозний перегляд мови, який не є повністю сумісним із зворотною стороною. Багато його основних функцій були передані до версій Python 2.6.x та 2.7.x. Випуски Python 3 включають утиліту 2to3, яка автоматизує (принаймні частково) переклад коду Python 2 на Python 3.

Дата закінчення терміну служби Python 2.7 спочатку була встановлена на 2015 рік, а потім перенесена на 2020 рік через занепокоєння тим, що велика кількість існуючого коду не може бути легко перенесена на Python 3. Більше виправлень безпеки та інших удосконалень для нього випускати не буде. У кінці життя Python 2 підтримується лише Python 3.6.x і пізніші версії.

Python 3.9.2 та 3.8.8 були прискорені, оскільки всі версії Python мали проблеми із безпекою, що призвело до можливого віддаленого виконання коду та отруєння веб-кешу.

Python – мова програмування з багатьма парадигмами. Об'єктно-орієнтоване програмування та структуроване програмування повністю підтримуються, і багато його функцій підтримують функціональне

програмування та аспектно-орієнтоване програмування (в тому числі метапрограмуванням та метаоб'єктами (магічні методи)). Багато інших парадигм підтримуються за допомогою розширень, включаючи дизайн за контрактом та логічне програмування.

Python використовує динамічне введення тексту та комбінацію підрахунку посилань та збирач сміття, що визначає цикл, для управління пам’яттю. Він також має динамічну роздільну здатність імен (пізнє прив'язування), яка зв'язує імена методів та змінних під час виконання програми.

Дизайн Python пропонує певну підтримку функціонального програмування за традицією Lisp. Він має функції фільтрування, картографування та зменшення; перелічити розуміння, словники, набори та вирази генераторів. Стандартна бібліотека має два модулі (itertools та functools), що реалізують функціональні інструменти, запозичені у Haskell та Standard ML.

Основна філософія мови узагальнена в документі Zen of Python (PEP 20), який включає такі афоризми, як:

- красиве – краще, ніж потворне;
- явне краще, ніж неявне;
- просте – краще, ніж складне;
- складний – це краще, ніж складний;
- підрахунок читабельності.

Замість того, щоб усі його функціональні можливості були вбудовані в його ядро, Python був розроблений таким чином, щоб бути дуже розширюваним (з модулями). Ця компактна модульність зробила його особливо популярним як засіб додавання програмованих інтерфейсів до існуючих додатків. Бачення Ван Россума щодо малої основної мови з великою стандартною бібліотекою та легко розширюваним перекладачем випливало з його розчарувань у ABC, який підтримував протилежний підхід. Python прагне до більш простого, менш

захараценого синтаксису та граматики, одночасно надаючи розробникам можливість вибору методології кодування. На відміну від девізу Perl "існує більше ніж один спосіб зробити це", Python охоплює "повинен бути один – і бажано лише один – очевидний спосіб зробити це" філософія дизайну. [69] Алекс Мартеллі, співробітник Фонду програмного забезпечення Python і автор книги про Python, пише, що "Описувати щось як" розумне "не вважається компліментом у культурі Python" .

Розробники Python прагнуть уникнути передчасної оптимізації та відкидають виправлення для некритичних частин еталонної реалізації CPython, які пропонують незначне збільшення швидкості ціною ясності. Коли швидкість важлива, програміст Python може переміщати критично важливі для часу функції до модулів розширення, написаних мовами, такими як C, або використовувати PyPy, своєчасний компілятор. Також доступний Cython, який перекладає скрипт Python на C і здійснює прямі виклики API рівня C в інтерпретатор Python.

Важливою метою розробників Python є підтримка його задоволення від використання. Це відображається в назві мови – данина поваги британській гумористичній групі Монті Пайтон – і в іноді грайливих підходах до навчальних посібників та довідкових матеріалів, таких як приклади, що стосуються спаму та яєць (із відомого етюду Монті Пайтона) стандартних foo and bar.

Поширеним неологізмом у спільноті Python є пітонічний, який може мати широкий діапазон значень, пов’язаних зі стилем програми. Сказати, що код пітонічний, означає сказати, що він добре використовує ідіоми Python, що він природний або демонструє вільну мову, що відповідає мінімалістичній філософії Python та наголосу на читабельності. На відміну від цього, код, який важко зрозуміти або читається як груба транскрипція з іншої мови програмування, називається непітонічним.

Користувачів та шанувальників Python, особливо тих, хто вважається обізнаним чи досвідченим, часто називають Pythonistas. Python має бути легкочитабельною мовою. Його форматування візуально не захаращене, і в ньому часто використовуються англійські ключові слова, де інші мови використовують розділові знаки. На відміну від багатьох інших мов, він не використовує фігурні дужки для розмежування блоків, а крапка з комою після операторів дозволена, але рідко, якщо взагалі використовується. Він має менше синтаксичних винятків та особливих випадків, ніж C або Pascal.

Python використовує пробіли, а не фігурні дужки або ключові слова, для відмежування блоків. Збільшення відступу відбувається після певних тверджень; зменшення відступу означає кінець поточного блоку. Таким чином, візуальна структура програми точно відображає семантичну структуру програми. Цю функцію іноді називають правилом "поза стороною", яке мають деякі інші мови, але в більшості мов відступ не має семантичного значення. Рекомендований розмір відступу – чотири пробіли.

Оператори Python включають (серед іншого):

- оператор присвоєння, використовуючи один знак рівності =;
- оператор if, який умовно виконує блок коду, разом з else та elif (скорочення else-if);
- оператор for, який переглядає ітерабельний об’єкт, захоплюючи кожен елемент до локальної змінної для використання вкладеним блоком;
- оператор while, який виконує блок коду, доки його умова відповідає істині;
- оператор try, що дозволяє вилученням, що містяться у вкладеному блоці коду, ловити та обробляти за винятком речень; він також

гарантує, що код очищення в блоці нарешті завжди буде виконуватися незалежно від того, як блок виходить;

- оператор raise, що використовується для отримання вказаного винятку або повторного підняття спійманого винятку;
- оператор class, який виконує блок коду та приєднує його локальний простір імен до класу для використання в об'єктно-орієнтованому програмуванні;
- оператор def, що визначає функцію або метод;
- оператор with з Python 2.5, випущений у вересні 2006 р. [82], який охоплює блок коду в контекстному менеджері (наприклад, отримання блокування перед запуском блоку коду та звільнення блокування після цього, або відкриття файлу, а потім закриваючи його), дозволяючи поведінку, схожу на отримання ресурсів – це ініціалізація (RAII), і замінює загальну ідіому try / нарешті;
- оператор break, виходить із циклу;
- оператор continue, пропускає цю ітерацію і продовжує наступний пункт;
- оператор del видаляє змінну, що означає, що посилання з імені на значення видаляється, і спроба використання цієї змінної спричинить помилку. Видалену змінну можна перепризначити;
- оператор pass, яка виконує функцію NOP. Це синтаксично потрібно для створення порожнього блоку коду;
- оператор assert, який використовується під час налагодження для перевірки умов, які мають застосовуватися;
- оператор yield, який повертає значення з функції генератора. З Python 2.5, yield також є оператором. Ця форма використовується для реалізації спільних програм;

- оператор return, який використовується для повернення значення з функції;
- оператор import, який використовується для імпорту модулів, функції чи змінні яких можна використовувати в поточній програмі. Є три способи використання імпорту: імпорт <ім'я модуля> [як <псевдонім>] або з <ім'я модуля> імпорт * або з <ім'я модуля> імпорт <визначення 1> [як <псевдонім 1>], <визначення 2> [як <псевдонім 2>], ...

Оператор присвоєння (=) діє шляхом прив'язки імені як посилання на окремий динамічно розподілений об'єкт. Потім змінні можуть бути відскочені в будь-який час до будь-якого об’єкта. У Python ім'я змінної є загальним власником посилання і не має фіксованого типу даних, пов’язаного з ним. Однак у даний момент часу змінна буде посилатися на якийсь об'єкт, який матиме тип. Це називається динамічним набором тексту і протиставляється статично набраним мовам програмування, де кожна змінна може містити лише значення певного типу.

Python не підтримує оптимізацію хвостових викликів або першокласні продовження, і, за словами Гвідо ван Россума, він ніколи не буде. Однак краща підтримка подібних до корутинів функціональних можливостей надається в 2.5, розширюючи генератори Python. До 2.5 генератори були ледачими ітераторами; інформація передавалась в односпрямованому напрямку з генератора. З Python 2.5 можна передавати інформацію назад у функцію генератора, а з Python 3.3 інформацію можна передавати через кілька рівнів стека.

Деякі вирази Python подібні до таких, що зустрічаються в таких мовах, як C та Java, а деякі – ні:

- додавання, віднімання та множення однакові, але поведінка ділення відрізняється. У Python існує два типи поділів. Вони є розподілом

підлоги (або цілочисельним поділом) // та плаваючою комою / поділом. Python також використовує оператор ** для піднесення до степені;

- з Python 3.5 був представлений новий оператор @ infix. Він призначений для використання бібліотеками, такими як NumPy, для множення матриць;
- з Python 3.8 був введений синтаксис: =, який називається «моржовим оператором». Він присвоює значення змінним як частину більшого виразу; [91]
- у Python == порівнює за значенням проти Java, яка порівнює числові значення за значенням, а об'єкти за посиланням. (Порівняння значень в Java на об'єктах можна виконувати методом equals ().) Оператор Python is is може використовуватися для порівняння ідентифікацій об'єктів (порівняння за посиланням). У Python порівняння можуть бути ланцюговими, наприклад a <= b <= c;
- python використовує слова та, або, не для своїх булевих операторів, а не для символічного &&, ||,! використовується в Java та C;
- python має тип виразу, що називається розумінням списку, а також більш загальний вираз, який називається генераторським виразом;
- анонімні функції реалізовані за допомогою лямбда-виразів; однак вони обмежені тим, що тіло може бути лише одним виразом;
- умовні вирази в Python пишуться як x, якщо c else y (відрізняється за порядком операндів від оператора c? X: y, загального для багатьох інших мов);
- python розрізняє списки та кортежі. Списки записуються як [1, 2, 3], змінюються та не можуть використовуватися як ключі словників (ключі словника повинні бути незмінними в Python). Кортежі

записуються як (1, 2, 3), є незмінними і, отже, можуть використовуватися як ключі словників, за умови, що всі елементи кортежу є незмінними. Оператор + може бути використаний для об'єднання двох кортежів, що безпосередньо не змінює їх вміст, а створює новий кортеж, що містить елементи обох передбачених кортежів. Таким чином, враховуючи змінну t, спочатку рівну (1, 2, 3), при виконанні t = t + (4, 5) спочатку обчислюється t + (4, 5), що дає (1, 2, 3, 4, 5) , який потім призначається назад до t, тим самим ефективно "модифікуючи вміст" t, одночасно відповідаючи незмінній природі об'єктів кортежу. Дужки не є обов’язковими для кортежів у однозначному контексті;

- python має розпаковування послідовностей, при якому кілька виразів, кожен з яких обчислює будь-що, до чого можна присвоїти (змінну, властивість для запису тощо), пов’язані ідентично тому, що формує кортежні літерали, і, як ціле, розміщуються на ліва сторона знака рівності у твердженні про присвоєння. Оператор очікує ітерабельного об'єкта праворуч від знака рівності, який видає таку ж кількість значень, що і надані вираз для запису, коли його перебирають і буде перебирати його, присвоюючи кожне із створених значень відповідному виразу зліва;
- python має оператор "формат рядка"%. Це функціонує аналогічно рядкам формату printf на C, наприклад "спам =% s яєць =% d"% ("бла", 2) оцінюється як "спам = бла-яйця = 2". У Python 3 та 2.6+ це було доповнено методом format () класу str, наприклад "спам = {0} яйця = {1}". формат ("бла", 2). Python 3.6 додав "f-рядки": blah = "blah"; яйця = 2; f'spam = {blah} яйця = {яйця} ';

- рядки в Python можна об'єднати, "додавши" їх (той самий оператор, що і для додавання цілих чи значень з плаваючою точкою). Наприклад "спам" + "яйця" повертає "спамегги". Навіть якщо ваші рядки містять числа, вони все одно додаються як рядки, а не цілі числа. Наприклад "2" + "2" повертає "22";
- python має різні типи рядкових літералів;
- рядки, розділені одинарними або подвійними лапками. На відміну від оболонок Unix, мови Perl та Perl, що впливають на Perl, одинарні лапки та подвійні лапки функціонують однаково. Обидва типи рядків використовують зворотну скісну риску (\) як символ виходу. Інтерполяція рядків стала доступною в Python 3.6 як "відформатовані рядкові літерали";
- рядки з подвійними лапками, які починаються та закінчуються серією з трьох одинарних або подвійних лапок. Вони можуть охоплювати кілька рядків і функціонувати, як тут документи в оболонках, Perl і Ruby;
- сирі різновиди рядків, що позначаються префіксом рядкового літералу перед r. Послідовності втечі не інтерпретуються; отже, необроблені рядки корисні там, де буквенні зворотні слеші є загальними, такі як регулярні вирази та шляхи у стилі Windows. Порівняйте "@ -citation" у C #;
- python має індекси масивів та вирази нарізування масивів у списках, що позначаються як [ключ], [старт: зупинка] або [старт: зупинка: крок]. Індекси базуються на нулі, а негативні індекси відносно кінця. Зрізи беруть елементи від початкового індексу до, але не включаючи, індексу зупинки. Третій параметр зрізу, який називається кроком або кроком, дозволяє пропускати та обертати елементи. Індекси зрізів

можуть бути опущені, наприклад, [:] повертає копію всього списку. Кожен елемент зрізу є неглибокою копією.

У Python чітко дотримується різниці між виразами та висловлюваннями, на відміну від таких мов, як Common Lisp, Scheme або Ruby. Це призводить до дублювання деяких функціональних можливостей.

Методи на об'єктах – це функції, приєднані до класу об'єкта. Синтаксис instance.method (аргумент) - це для звичайних методів та функцій синтаксичний цукор для Class.method (екземпляр, аргумент). Методи Python мають явний параметр self для доступу до даних екземпляра, на відміну від неявного self (або цього) в деяких інших об’єктно-орієнтованих мовах програмування (наприклад, C ++, Java, Objective-C або Ruby).

Python використовує набір качок і має набрані об'єкти, але нетипізовані імена змінних. Обмеження типу не перевіряються під час компіляції; швидше, операції з об'єктом можуть зазнати невдачі, що означає, що даний об'єкт не є відповідним типом. Не дивлячись на те, що Python набирається динамічно, забороняє операції, які не є чітко визначеними (наприклад, додавання числа до рядка), а не намагається їх тихо зрозуміти.

Python дозволяє програмістам визначати власні типи за допомогою класів, які найчастіше використовуються для об'єктно-орієнтованого програмування. Нові екземпляри класів будуються за допомогою виклику класу (наприклад, SpamClass () або EggsClass ()), а класи є екземплярами типу метакласу (сам екземпляр самого себе), що дозволяє метапрограмувати та відображати.

До версії 3.0 Python мав два типи класів: старий і новий. Синтаксис обох стилів однаковий, різниця полягає в тому, чи об’єкт класу успадковується, прямо чи опосередковано (усі класи нового стилю успадковуються від об’єкта і є екземплярами типу). У версіях Python 2, починаючи з Python 2.2, можна

використовувати обидва типи класів. Класи старого стилю були ліквідовані в Python 3.0.

Довгостроковий план полягає в підтримці поступового набору тексту, а з Python 3.5 синтаксис мови дозволяє вказувати статичні типи, але вони не перевіряються в реалізації за замовчуванням, CPython. Експериментальна додаткова перевірка статичного типу з ім'ям муру підтримує перевірку типу під час компіляції.

CPython – це посилальна реалізація Python. Він написаний на мові C, відповідаючи стандарту C89 з декількома вибраними функціями C99 (з випуском пізніших версій C він вважається застарілим; CPython включає власні розширення C, але розширення сторонніх розробників не обмежуються старими C версії, наприклад, можуть бути реалізовані з C11 або C ++). Він компілює програми Python у проміжний байт-код, який потім виконується його віртуальною машиною. CPython поширюється з великою стандартною бібліотекою, написаною на суміші C та рідного Python. Він доступний для багатьох платформ, включаючи Windows (починаючи з Python 3.9, інсталятор Python навмисно не вдається встановити в Windows 7 і 8; Windows XP підтримувався до Python 3.5) та більшості сучасних Unix-подібних систем, включаючи macOS (та Apple M1 Macs, починаючи з Python 3.9.1, з експериментальним інсталятором) та неофіційну підтримку, наприклад VMS. Переносимість платформи була одним із її перших пріоритетів, протягом періодів Python 1 та 2 підтримувались навіть OS / 2 та Solaris; підтримка відтоді відмовилася для багатьох платформ.

Розробка Python здійснюється в основному за допомогою процесу Програми вдосконалення Python (PEP), основного механізму пропонування основних нових можливостей, збору вкладу спільноти з питань та документування рішень щодо проектування Python. Стиль кодування Python

викладений у PEP 8. Видатні PEP переглядаються та коментуються спільнотою Python та керівною радою.

Покращення мови відповідає розробці посилальної реалізації CPython. Список розсилки python-dev є основним форумом для розвитку мови. Конкретні проблеми обговорюються у програмі відстеження помилок Roundup, розміщеній на bugs.python.org. Спочатку розробка здійснювалась у власному сховищі вихідного коду під управлінням Mercurial, поки Python не перейшов на GitHub у січні 2017 року.

Публічні випуски CPython бувають трьох типів, що відрізняються тим, яка частина номера версії збільшується:

Несумісні із зворотною мовою версії, де, як очікується, зламається код і його потрібно перенести вручну. Перша частина номера версії збільшується. Ці випуски трапляються рідко – версія 3.0 вийшла через 8 років після 2.0.

Основні або "функціональні" випуски відбувалися приблизно кожні 18 місяців, але з прийняттям щорічної каденції випуску, починаючи з Python 3.9, очікується, що це відбуватиметься раз на рік. Вони значною мірою сумісні, але вводять нові функції. Друга частина номера версії збільшується. Кожна основна версія підтримується виправленнями протягом декількох років після її випуску.

Випуски виправлень, які не вводять нових функцій, трапляються приблизно кожні 3 місяці і робляться, коли з останнього випуску виправлена достатня кількість помилок. Уразливості безпеки також виправлені у цих випусках. Третя і остання частина номера версії збільшується.

Багато альфа-, бета-версій та кандидатів на випуск також випускаються у вигляді попереднього перегляду та тестування перед остаточними випусками. Хоча існує приблизний графік кожного випуску, вони часто затримуються, якщо код не готовий. Команда розробників Python контролює стан коду, запускаючи великий пакет модульних тестів під час розробки.

Основною академічною конференцією з Python є PyCon. Існують також спеціальні програми наставництва Python, такі як Pyladies.

Pythons 3.10 припиняє wstr (буде видалено в Python 3.12; це означає, що розширення Python потрібно буде змінити до того часу), а також планує додати відповідність шаблону до мови.

Використання

З 2003 р. Python стабільно входить до десятки найпопулярніших мов програмування в Індексі програмного забезпечення TIOBE, де, станом на лютий 2021 р., Це третя за популярністю мова (після Java та C). Він був обраний мовою програмування року (за "найвищий ріст рейтингу за рік") у 2007, 2010, 2018 та 2020 роках (єдина мова, яка зробила це чотири рази).

Емпіричне дослідження показало, що мови сценаріїв, такі як Python, є більш продуктивними, ніж звичайні мови, такі як C та Java, для проблем програмування, що включають маніпулювання рядками та пошук у словнику, і встановило, що споживання пам'яті часто "краще, ніж Java, а не набагато гірше, ніж C або C ++ ".

До великих організацій, які використовують Python, належать Wikipedia, Google, Yahoo!, CERN, NASA, Facebook, Amazon, Instagram, Spotify та деякі менші організації, такі як ILM та ITA. Сайт соціальних новин Reddit був написаний переважно на Python.

Python може служити мовою сценаріїв для веб-додатків, наприклад, через mod_wsgi для веб-сервера Apache. Завдяки інтерфейсу шлюзу веб-сервера для полегшення роботи цих програм розроблено стандартний API. Веб-фреймворки, такі як Django, Pylons, Pyramid, TurboGears, web2py, Tornado, Flask, Bottle і Zope, підтримують розробників у розробці та обслуговуванні складних додатків. Pyjs та IronPython можуть бути використані для розробки клієнтської частини програм на базі Ajax. SQLAlchemy може використовуватися як перетворювач

даних у реляційну базу даних. Twisted – це фреймворк для програмування зв’язку між комп’ютерами, який використовується (наприклад) Dropbox.

Такі бібліотеки, як NumPy, SciPy та Matplotlib, дозволяють ефективно використовувати Python у наукових обчисленнях, а спеціалізовані бібліотеки, такі як Biopython та Astropy, забезпечують функціональність, що залежить від домену. SageMath – математичне програмне забезпечення з інтерфейсом ноутбука, програмоване на Python: його бібліотека охоплює багато аспектів математики, включаючи алгебру, комбінаторику, числову математику, теорію чисел і числення. OpenCV має палітурні прив'язки з багатим набором функцій для комп'ютерного зору та обробки зображень.

Python зазвичай використовується в проектах штучного інтелекту та машинному навчанні за допомогою таких бібліотек, як TensorFlow, Keras, Pytorch та Scikit-learn. Як мова сценаріїв з модульною архітектурою, простим синтаксисом та інструментами обробки багатого тексту, Python часто використовується для обробки природної мови. Python успішно вбудований у багато програмних продуктів як мову сценаріїв, включаючи програмне забезпечення методом скінченних елементів, таке як Abaqus, 3D-параметричний моделер, такий як FreeCAD, пакети 3D-анімації, такі як 3ds Max, Blender, Cinema 4D, Lightwave, Houdini, Maya, modo , MotionBuilder, Softimage, композитор візуальних ефектів Nuke, програми для 2D-зображень, такі як GIMP, Inkscape, Scribus і Paint Shop Pro, та музичні нотатні програми, такі як автори та капели. Налагоджувач GNU використовує Python як гарний принтер для показу складних структур, таких як контейнери C ++. Esri пропагує Python як найкращий вибір для написання сценаріїв в ArcGIS. Він також використовувався в декількох відеоіграх і був прийнятий як перша з трьох доступних мов програмування в Google App Engine, інші дві – Java і Go.

Багато операційних систем включають Python як стандартний компонент. Він постачається з більшістю дистрибутивів Linux, AmigaOS 4 (з використанням Python 2.7), FreeBSD (як пакет), NetBSD, OpenBSD (як пакет) та macOS і може використовуватися з командного рядка (терміналу). Багато дистрибутивів Linux використовують інсталятори, написані на Python: Ubuntu використовує інсталятор Ubiquity, тоді як Red Hat Linux і Fedora використовують інсталятор Anaconda. Gentoo Linux використовує Python у своїй системі управління пакунками Portage.

Python широко використовується в галузі інформаційної безпеки, в тому числі в розробці експлойтів.

Більшість програм Sugar для одного ноутбука на дитину XO, які зараз розробляються в Sugar Labs, написані на Python. Проект одноплатної комп'ютерної програми Raspberry Pi прийняв Python як основну мову програмування користувачів.

LibreOffice включає Python і має намір замінити Java на Python. Його постачальник сценаріїв Python є основною функцією з версії 4.0 від 7 лютого 2013 року.

2.2 Система управління базами даних PostgreSQL

PostgreSQL, [12] також відомий як Postgres, – це безкоштовна система управління реляційними базами даних із відкритим кодом (RDBMS), що підкреслює розширюваність та відповідність SQL. Спочатку він називався POSTGRES, посилаючись на своє походження як спадкоємець бази даних Ingres, розробленої в Каліфорнійському університеті в Берклі. [13] [14] У 1996 році проект був перейменований на PostgreSQL, щоб відобразити підтримку SQL.

Після огляду в 2007 році команда розробників вирішила зберегти назву PostgreSQL та псевдонім Postgres. [15]

PostgreSQL містить транзакції зі властивостями Atomicity, Consistentity, Isolation, Durability (ACID), автоматично оновлювані подання, матеріалізовані подання, тригери, зовнішні ключі та збережені процедури. Він призначений для обробки ряду робочих навантажень, від окремих машин до сховищ даних або веб-сервісів із багатьма одночасними користувачами. Це база даних за замовчуванням для macOS Server [17] [18] [19], а також доступна для Windows, Linux, FreeBSD та OpenBSD.

PostgreSQL з’явився в результаті проекту Ingres в Університеті Каліфорнії, Берклі. У 1982 році лідер команди в Інгресі Майкл Стоунбракер залишив Берклі, щоб створити власну версію Ingres. [13] Він повернувся в Берклі в 1985 році і розпочав проект після Інгресу для вирішення проблем із сучасними системами баз даних, які стали все більш зрозумілими на початку 1980-х. У 2014 році він виграв премію Тьюрінга за ці та інші проекти [20], а також новаторські технології в них.

Новий проект, POSTGRES, мав на меті додати найменшу кількість функцій, необхідних для повної підтримки типів даних. [21] Ці функції включали можливість визначати типи та повністю описувати взаємозв'язки – те, що широко використовується, але повністю підтримується користувачем. У POSTGRES база даних розуміла взаємозв'язки і могла отримувати інформацію у відповідних таблицях природним чином, використовуючи правила. ПОСТГРЕС використовував багато ідей Енгра, але не його код. [22]

Починаючи з 1986 року, опубліковані статті описували основу системи, а прототип версії був показаний на конференції ACM SIGMOD 1988 року. У червні 1989 року команда випустила версію 1 для невеликої кількості користувачів, а за нею – версію 2 із переписаною системою правил у червні 1990 року. Версія 3,

випущена в 1991 році, знову переписала систему правил і додала підтримку кількох менеджери сховищ [23] та вдосконалений механізм запитів. До 1993 року кількість користувачів почала засипати проект запитами на підтримку та функції. Після випуску версії 4.2 [24] 30 червня 1994 р. – насамперед очищення – проект закінчився. Берклі випустив POSTGRES під варіантом ліцензії MIT, що дозволило іншим розробникам використовувати код для будь-якого використання. Тоді POSTGRES використовував інтерпретатор мови запитів POSTQUEL під впливом Ingres, який можна було інтерактивно використовувати з консольним додатком з іменем monitor.

У 1994 р. аспіранти Берклі Ендрю Ю та Джолі Чен замінили інтерпретатор мови запитів POSTQUEL на мову мови запитів SQL, створивши Postgres95. Консоль монітора також була замінена на psql. Ю і Чень оголосили першу версію (0,01) для бета-тестерів 5 травня 1995 р. Версія 1.0 Postgres95 була оголошена 5 вересня 1995 р. З більш ліберальною ліцензією, що дозволила програмне забезпечення вільно модифікуватися.

8 липня 1996 року Марк Фурньє з Hub.org Networking Services надав перший сервер розвитку, який не є університетом, для роботи з відкритим кодом. [3] За участю Брюса Момджяна та Вадима Б. Міхеєва розпочалася робота по стабілізації коду, успадкованого від Берклі.

У 1996 році проект був перейменований на PostgreSQL, щоб відобразити підтримку SQL. Присутність в Інтернеті на веб-сайті PostgreSQL.org розпочалася 22 жовтня 1996 р. [25] Перший випуск PostgreSQL сформував версію 6.0 29 січня 1997 р. З тих пір розробники та волонтери у всьому світі підтримують програмне забезпечення як The PostgreSQL Global Development Group. [2]

Проект продовжує надавати випуски під своїм безкоштовним програмним забезпеченням з відкритим кодом PostgreSQL. Код походить від внесків власних постачальників, компаній підтримки та програмістів з відкритим кодом.

PostgreSQL управляє паралельністю за допомогою мультиверсійного паралельного контролю (MVCC), який надає кожній транзакції "знімок" бази даних, дозволяючи вносити зміни, не впливаючи на інші транзакції. Це значною мірою усуває необхідність блокування читання та гарантує, що база даних підтримує принципи ACID. PostgreSQL пропонує три рівні ізоляції транзакцій: зчитування, повторне читання та серіалізація. Оскільки PostgreSQL захищений від брудних читань, запит рівня ізоляції транзакції Read Uncommit забезпечує замість цього здійснене читання. PostgreSQL підтримує повну серіалізацію за допомогою серіалізованого методу ізоляції знімків (SSI). [26]

PostgreSQL включає вбудовану двійкову реплікацію, засновану на доставці змін (журнали попереднього запису (WAL)) до реплікаційних вузлів асинхронно, з можливістю запускати запити лише для читання щодо цих реплікаційних вузлів. Це дозволяє ефективно розподіляти трафік читання між кількома вузлами. Раніше програмне забезпечення для реплікації, яке дозволяло подібне масштабування читання, зазвичай покладалося на додавання тригерів реплікації до головного, збільшуючи навантаження.

PostgreSQL включає вбудовану синхронну реплікацію [27], яка гарантує, що для кожної транзакції запису ведучий чекає, поки принаймні один вузол репліки не запише дані до свого журналу транзакцій. На відміну від інших систем баз даних, довговічність транзакції (незалежно від того, є вона асинхронною чи синхронною) може бути вказана для кожної бази даних, для кожного користувача, за сеанс або навіть для кожної транзакції. Це може бути корисним для робочих навантажень, які не вимагають таких гарантій, і, можливо, не потрібні для всіх даних, оскільки це уповільнює роботу через вимогу підтвердження транзакції, що доходить до синхронного режиму очікування.

Резервні сервери можуть бути синхронними або асинхронними. Синхронні резервні сервери можна вказати в конфігурації, яка визначає, які сервери є

кандидатами на синхронну реплікацію. Перший у списку, який активно транслюється, буде використовуватися як поточний синхронний сервер. Коли це не вдається, система переходить до наступного по черзі.

Синхронна багаторівнева реплікація не включена в ядро PostgreSQL. Postgres-XC, що базується на PostgreSQL, забезпечує масштабовану синхронну мульти-головну реплікацію. [28] Він ліцензований за тією ж ліцензією, що і PostgreSQL. Пов’язаний проект називається Postgres-XL. Postgres-R – ще одна форк. [29] Двонаправлена реплікація (BDR) – це асинхронна багатопрофільна система реплікації для PostgreSQL. [30]

PostgreSQL включає вбудовану підтримку звичайних індексів B-дерева та хеш-таблиць та чотири методи доступу до індексу: узагальнені дерева пошуку (GiST), узагальнені інверсовані індекси (GIN), розділений пробілами GiST (SP-GiST) [32] та блок Індекси діапазону (BRIN). Крім того, можна створювати призначені користувачем методи індексу, хоча це досить залучений процес. Індекси в PostgreSQL також підтримують такі функції:

- індекси виразів можна створювати за допомогою індексу результату виразу або функції, а не просто значенням стовпця;
- часткові індекси, які індексують лише частину таблиці, можна створити, додавши пропозицію WHERE у кінець оператора CREATE INDEX. Це дозволяє створити менший індекс;
- планувальник може використовувати декілька індексів разом, щоб задовольнити складні запити, використовуючи тимчасові операції з індексом растрових зображень в пам'яті (корисно для програм зберігання даних для приєднання великої таблиці фактів до менших таблиць розмірів, таких як розміщені в зірковій схемі);
- індексація k-найближчих сусідів (k-NN) (також звана KNN-GiST [33]) забезпечує ефективний пошук "найближчих значень" до вказаних,

корисних для пошуку подібних слів, або закриття об'єктів або місць із геопросторовими даними. Це досягається без вичерпного збігу значень;

- індексне сканування часто дозволяє системі отримувати дані з індексів, не маючи доступу до головної таблиці;
- postgreSQL 9.5 представив індекси блокових діапазонів (BRIN);

У PostgreSQL схема містить усі об'єкти, крім ролей і табличних просторів. Схеми ефективно діють як простори імен, дозволяючи об’єктам з однаковим ім’ям співіснувати в одній базі даних. За замовчуванням у новостворених базах даних є схема, яка називається загальнодоступною, але будь-які подальші схеми можуть бути додані, і загальнодоступна схема не є обов’язковою.

Параметр search_path визначає порядок, у якому PostgreSQL перевіряє схеми на предмет некваліфікованих об'єктів (тих, що не мають попередньо встановленої схеми). За замовчуванням для нього встановлено $ user, public ($ user відноситься до поточно підключеного користувача бази даних). Це значення за замовчуванням можна встановити на рівні бази даних або ролі, але оскільки це параметр сеансу, його можна вільно змінювати (навіть кілька разів) під час сеансу клієнта, впливаючи лише на цей сеанс.

Неіснуючі схеми, перелічені в search_path, мовчки пропускаються під час пошуку об'єктів.

Нові об'єкти створюються в залежності від того, яка діюча схема (та, яка існує на даний момент) з'являється першою у шляху_пошуку.

Таблиці можна встановити для успадкування своїх характеристик від батьківської таблиці. Дані в дочірніх таблицях, здається, існують у батьківських таблицях, якщо тільки дані не вибрані з батьківської таблиці за допомогою ключового слова ONLY, тобто SELECT * FROM ONL parent_table; Додавання стовпця в батьківську таблицю призведе до появи цього стовпця в дочірній таблиці.

Наслідування може бути використана для реалізації розділення таблиці, використовуючи тригери або правила, щоб направити вставки до батьківської таблиці у відповідні дочірні таблиці.

Станом на 2010 р. Ця функція ще не повністю підтримується – зокрема, обмеження таблиці наразі не успадковуються. Усі обмеження перевірки та ненульові обмеження батьківської таблиці автоматично успадковуються її нащадками. Інші типи обмежень (обмеження унікального, первинного ключа та зовнішнього ключа) не успадковуються.

Наслідування забезпечує спосіб відображення особливостей ієрархій узагальнення, зображених на діаграмах взаємозв'язків сутності (ERD), безпосередньо в базі даних PostgreSQL.

PostgreSQL забезпечує асинхронну систему обміну повідомленнями, доступ до якої здійснюється за допомогою команд NOTIFY, LISTEN і UNLISTEN. Сеанс може подати команду NOTIFY разом із вказаним користувачем каналом та необов'язковим корисним навантаженням для позначення певної події, що відбувається. Інші сесії можуть виявити ці події, видавши команду LISTEN, яка може прослуховувати певний канал. Цю функціональність можна використовувати для найрізноманітніших цілей, наприклад, повідомляти інші сеанси про оновлення таблиці або для окремих додатків для виявлення, коли було виконано певну дію. Така система запобігає необхідності постійного опитування додатків, щоб дізнатись, чи щось ще не змінилося, і зменшуючи непотрібні накладні витрати. Повідомлення є повністю транзакційними, оскільки повідомлення не надсилаються, доки не буде здійснено транзакцію, з якої вони були надіслані. Це усуває проблему надсилання повідомлень для виконуваної дії, яка потім відкочується назад.

Багато роз'ємів для PostgreSQL забезпечують підтримку цієї системи сповіщень (включаючи libpq, JDBC, Npgsql, psycopg та node.js), тому вона може використовуватися зовнішніми програмами.

PostgreSQL може виступати ефективним, постійним сервером "pub / sub" або сервером завдань, поєднуючи LISTEN з FOR UPDATE SKIP LOCKED, [49] комбінацією, яка існує з версії PostgreSQL 9.5 [50] [51]

2.3 Хмарна платформа Amazon AWS

Amazon Web Services (AWS) є дочірньою компанією Amazon, яка надає платформи та API для хмарних обчислень на вимогу приватним особам, компаніям та урядам на основі дозованої оплати. Ці веб-служби хмарних обчислень забезпечують різноманітну базову абстрактну технічну інфраструктуру та розподілені обчислювальні блоки та інструменти. Однією з таких служб є Amazon Elastic Compute Cloud (EC2), яка дозволяє користувачам мати у своєму розпорядженні віртуальний кластер комп’ютерів, постійно доступний через Інтернет. Версія віртуальних комп’ютерів AWS емулює більшість атрибутів реального комп’ютера, включаючи апаратні центральні процесори (ЦП) та графічні процесори (ГП) для обробки; локальна / оперативна пам’ять; сховище на жорсткому диску / SSD; вибір операційних систем; створення мереж; та попередньо завантажене прикладне програмне забезпечення, таке як веб-сервери, бази даних та управління взаємовідносинами з клієнтами (CRM).

Технологія AWS впроваджена на серверах ферм у всьому світі та підтримується дочірньою компанією Amazon. Комісія базується на поєднанні використання (відомого як модель "плати за дорогу"), апаратного забезпечення, операційної системи, програмного забезпечення або функцій мережі, обраних

абонентом, необхідної доступності, надмірності, безпеки та параметрів обслуговування. Абоненти можуть платити за один віртуальний комп'ютер AWS, виділений фізичний комп'ютер або кластери будь-якого з них. В рамках угоди про передплату [10] Amazon забезпечує безпеку систем абонентів. AWS працює з багатьох глобальних географічних регіонів, включаючи 6 у Північній Америці. [11]

Amazon продає AWS абонентам як спосіб отримання великих обчислювальних потужностей швидше та дешевше, ніж побудова фактичної ферми фізичних серверів. [12] За всі послуги стягується плата залежно від використання, але кожна послуга вимірює використання різними способами. Станом на 2017 рік, AWS володіє домінуючим 33% всієї хмарності (IaaS, PaaS), тоді як у наступних двох конкурентів, Microsoft Azure та Google Cloud, відповідно 18% та 9%, згідно з даними Synergy Group. [13] [14]

Станом на 2021 рік AWS включає понад 200 [15] продуктів та послуг, включаючи обчислення, зберігання, мережу, базу даних, аналітику, прикладні послуги, розгортання, управління, машинне навчання, [16] мобільні, інструменти для розробників та інструменти для Інтернету речей . Найбільш популярними є Amazon Elastic Compute Cloud (EC2), Amazon Simple Storage Service (Amazon S3), Amazon Connect і AWS Lambda (безсерверна функція, що дозволяє безсерверний ETL, наприклад, між екземплярами EC2 і S3). [17]

Більшість служб не піддаються безпосередньому користуванню кінцевими користувачами, але натомість пропонують функціональність через API для розробників для використання у своїх додатках. До пропозицій Amazon Web Services можна отримати доступ через HTTP, використовуючи архітектурний стиль REST та протокол SOAP для старих API та виключно JSON для нових.

Станом на січень 2021 року AWS здійснює окремі операції в 25 географічних "регіонах": [11] 7 у Північній Америці, 1 у Південній Америці, 6 у

Європі, 1 на Близькому Сході, 1 у Африці та 8 у Азіатсько-Тихоокеанському регіоні.

AWS збирається запустити ще 15 своїх зон доступності і знову окремо ще п'ять зон в Австралії, Індії, Індонезії, Іспанії та Швейцарії [11].

Кожен регіон повністю міститься в межах однієї країни, і всі його дані та послуги залишаються у визначеному регіоні. [10] Кожен регіон має безліч "Зон доступності" [103], які складаються з одного або декількох дискретних центрів обробки даних, кожен із надлишковим живленням, мережею та підключенням, розміщений в окремих приміщеннях. Зони доступності не забезпечують автоматично додаткову масштабованість або надмірність в межах регіону, оскільки вони навмисно ізольовані один від одного, щоб запобігти поширенню відключень між зонами. Кілька служб можуть працювати в зонах доступності (наприклад, S3, DynamoDB), тоді як інші можуть бути налаштовані на реплікацію в зонах для розповсюдження попиту та уникнення простою внаслідок збоїв.

За станом на грудень 2014 року Amazon Web Services експлуатував приблизно 1,4 мільйона серверів у 28 зонах доступності. [104] Глобальна мережа локацій AWS Edge складається з 54 пунктів присутності у всьому світі, включаючи локації в США, Європі, Азії, Австралії та Південній Америці [105].

У 2014 році AWS заявила, що її метою є досягнення 100% використання відновлюваної енергії в майбутньому. [106] У Сполучених Штатах партнерські відносини AWS з постачальниками відновлюваних джерел енергії включають Community Energy of Virginia для підтримки східного регіону США; [107] Pattern Development, в січні 2015 року, для побудови та експлуатації хребта Фаулер Вінд Фарм Амазонки; [108] Iberdrola Renewables, LLC, у липні 2015 року, з метою побудови та експлуатації Amazon Wind Farm US East; EDP Renewables North America, у листопаді 2015 року, для побудови та експлуатації Amazon Wind Farm US Central; [109] та Tesla Motors, щоб застосувати технологію зберігання

акумуляторів для задоволення потреб в енергії в регіоні Західної (Північної Каліфорнії) США. [107]

AWS також має "спливаючі горища" в різних місцях по всьому світу. [110] Вони пропонують AWS для підприємців та стартапів у різних галузях техніки у фізичному місці. Відвідувачі можуть попрацювати або відпочити всередині горища або дізнатись більше про те, що вони можуть робити з AWS. У червні 2014 року AWS відкрив свій перший тимчасовий спливаючий горище в Сан-Франциско. [111] У травні 2015 року вони розширилися до Нью-Йорка [112] [113], а у вересні 2015 року – до Берліна [114]. AWS відкрили свою четверту філію в Тель-Авіві з 1 березня 2016 року по 22 березня 2016 року [115]. Спливний лофт був відкритий у Лондоні з 10 вересня по 29 жовтня 2015 р. [116] Спливні лофти в Нью-Йорку [117] та Сан-Франциско [118] закриті на невизначений час через пандемію COVID-19, тоді як Токіо залишається відкритим обмеженою кількістю. [119]

AWS також має "спливаючі горища" в різних місцях по всьому світу. [110] Вони пропонують AWS для підприємців та стартапів у різних галузях техніки у фізичному місці. Відвідувачі можуть попрацювати або відпочити всередині горища або дізнатись більше про те, що вони можуть робити з AWS. У червні 2014 року AWS відкрив свій перший тимчасовий спливаючий горище в Сан-Франциско. [111] У травні 2015 року вони розширилися до Нью-Йорка [112] [113], а у вересні 2015 року - до Берліна [114]. AWS відкрили свою четверту філію в Тель-Авіві з 1 березня 2016 року по 22 березня 2016 року [115]. Спливний лофт був відкритий у Лондоні з 10 вересня по 29 жовтня 2015 р. [116] Спливні лофти в Нью-Йорку [117] та Сан-Франциско [118] закриті на невизначений час через пандемію COVID-19, тоді як Токіо залишається відкритим обмеженою кількістю. [119]

3 РЕАЛІЗАЦІЯ ЧАТ-БОТУ ДЛЯ ОПТИМІЗАЦІЇ ГОСПОДАРСЬКОЇ РОБОТИ В ГУРТОЖИТКУ

3.1 Діаграма варіантів використання

Діаграма використання найпростіша – це представлення взаємодії користувача із системою, яка показує взаємозв'язок між користувачем та різними випадками використання, в яких користувач бере участь. Діаграма випадків використання може ідентифікувати різні типи користувачів системи та різні випадки використання, і часто вона супроводжується також іншими типами діаграм. Варіанти використання представлені кругами або еліпсами.

Незважаючи на те, що сам випадок використання може детально вивчити кожну можливість, діаграма прикладів використання може допомогти забезпечити огляд системи на більш високому рівні. Раніше вже було сказано, що "схеми використання – це принципи вашої системи".

Через їх спрощений характер, схеми використання можуть бути хорошим інструментом комунікації для зацікавлених сторін. Креслення намагаються імітувати реальний світ і дають зацікавленій стороні уявлення про те, як буде розроблена система. Сіау та Лі провели дослідження, щоб визначити, чи взагалі існувала дійсна ситуація для схем використання або вони були непотрібними. Було виявлено, що діаграми випадків використання передають намір системи більш спрощеним чином зацікавленим сторонам і що вони "інтерпретуються більш повно, ніж діаграми класів".

Метою діаграми використання є відображення динамічного аспекту системи. Додаткові схеми та документація можуть бути використані для забезпечення повного функціонального та технічного уявлення про систему.

Вони забезпечують спрощене та графічне представлення того, що система насправді повинна робити.

Елементи діаграми використання:

- рамки системи (англ. system border) – прямоугольник із назвою у верхніх частинах та еліпсами (прецедентами) всередині. Часто може бути опущено без корисної інформації про полезну інформацію;
- актор (англ. актор) – стилізований людський персонаж, обозначаючий набір ролей користувача (розуміється в широкому змісті: людина, внешняя сутність, клас, інша система), взаємодіючого з деякою сутністю (системною, підсистемою, класом). Актори не можуть бути пов'язані між собою з іншим (за вимкнення відносин щодо обробки / дослідження);
- прецедент – еліпс із надписом, що обозначає виконувану систематичну дію (може включати можливі варіанти), що призводить до спостерігаємих акторами результатів. Надпис може бути ім'ям або описом (з точки зору актора) того, "що" робить система (а не "як"). Ім прецедента зв’язано з неперервним (атомарним) сценарієм – конкретною послідовністю дій, ілюстріруючою повіддю. Під час сценарію актори обмінюються із систематичними повідомленнями. Сценарій може бути приведений на діаграмі прецедентів у відео UML-коментарі. З одним прецедентом може бути пов'язано кілька різних сценаріїв

На рисунках 3.1 – 3.2 зображено діаграму варіантів використання, яка описує можливі дії користувача в системі.

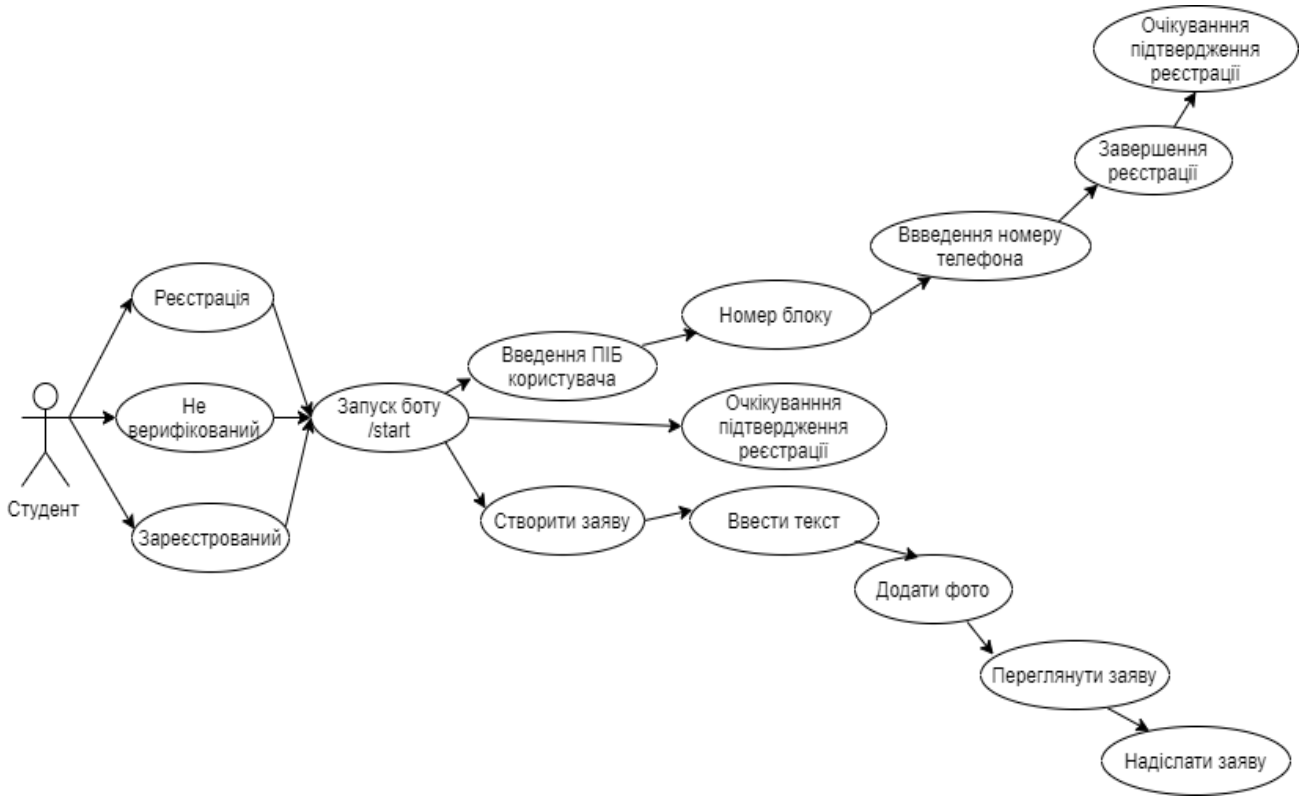


Рис. 3.1 — Діаграма варіантів використання (студент)

Рис. 3.2 — Діаграма варіантів використання (адмін)

3.2 Структурна діаграма

На рисунку 3.3 зображена структурна діаграма розроблюваного програмного продукту, яка описує взаємодію між внутрішніми компонентами системи та ілюструє внутрішню будову розроблюваного програмного продукту.

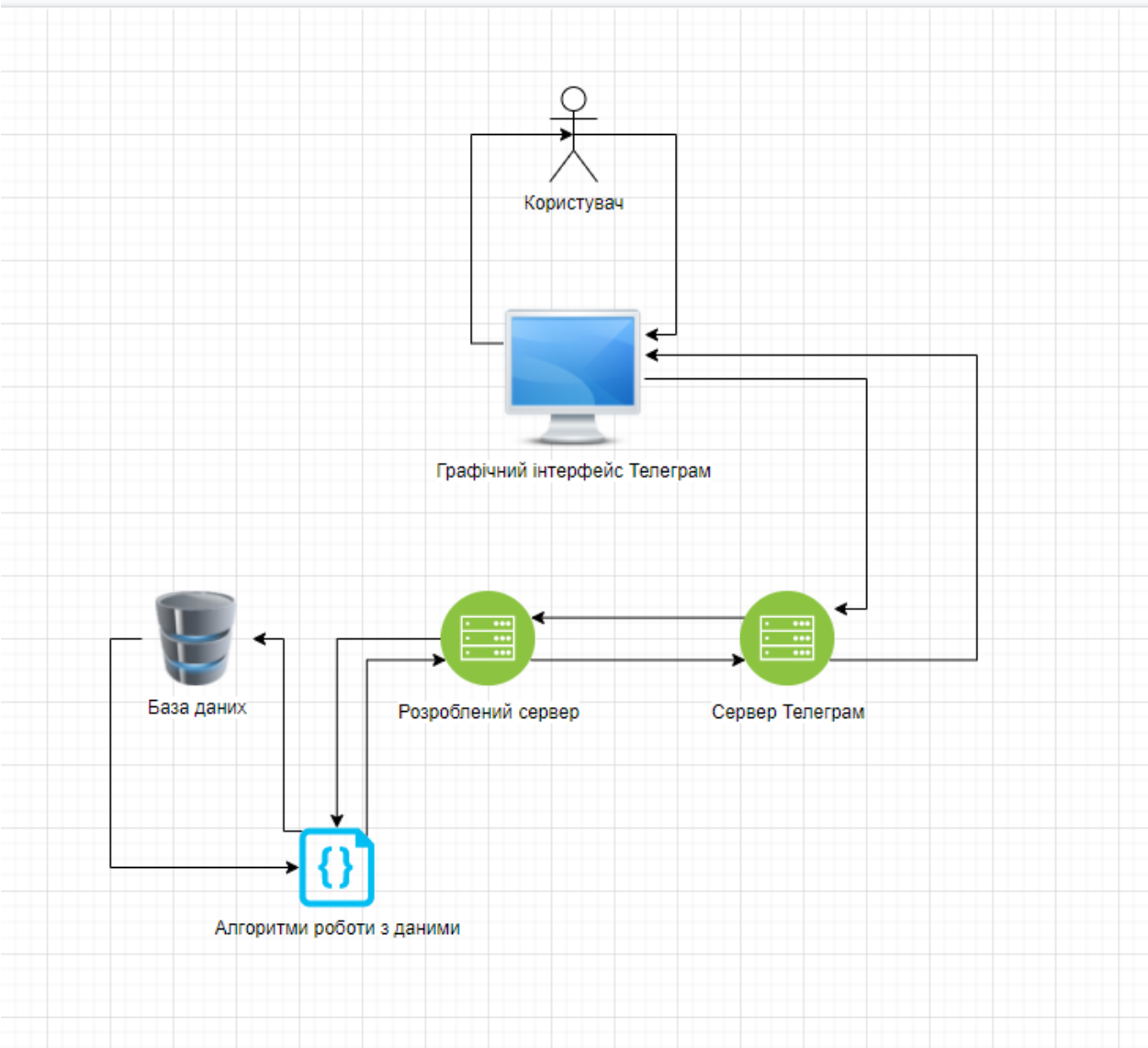


Рис. 3.3 — Структурна діаграма ІС

3.3 Основні механізми роботи

Робота проекту починається з запуску боту на рівні серверної частини, за допомогою фреймворку aiogram (лістинг 3.1)

Лістинг 3.1

Запуск бота

```
from aiogram import Bot, Dispatcher, types
from aiogram.contrib.fsm_storage.memory import MemoryStorage
from data import config
from utils.db_api.postgresql import Database
bot = Bot(token=config.BOT_TOKEN, parse_mode=types.ParseMode.HTML)
storage = MemoryStorage()
dp = Dispatcher(bot, storage=storage)
db = Database()
```

Важливим механізмом роботи проекту є отримання даних стосовно токену боту, списку адміністраторів, адреси хоста та таблиць бази даних. Даний процес проілюстровано в лістингу 3.2

Лістинг 3.2

Кофігурування проекту

```
import os
from environs import Env
env = Env()
env.read_env()
BOT_TOKEN = env.str("BOT_TOKEN")
ADMINS = env.list("ADMINS")
IP = env.str("ip")
```

```
DB_USER = str(os.getenv("DB_USER"))
DB_PASS = str(os.getenv("DB_PASS"))
DB_HOST = str(os.getenv("DB_HOST"))
DB_NAME = str(os.getenv("DB_NAME"))
```

Одним із ключових моментів є отримання адміністратором заявок на роботи в гуртожитку. Дана механіка показана у лістингу 3.3.

Лістинг 3.3

Отримання заявок адміністратором

```
if telegram_id in ADMINS:
    reports = await db.select_reports_by_date()
    if reports:
        for report in reports:
            report_id = report[0]
            user_id = report[1]
            text = report[2]
            image = report[3]
            date = report[6].strftime("%d/%m/%Y")
            user = await db.select_user(telegram_id=user_id)
            users_phone = user[2]
            users_name = user[3]
            markup = await form_report_keyboard(report_id)
            await message.answer(f"Нова заява № {report_id}:\n\n"
                                 f"{users_name}\n"
                                 f"Тел: {users_phone}\n\n"
                                 f"{text}\n\n"
                                 f"Дата: {date}", reply_markup=markup)
```

```
            await message.answer_photo(image)
        else:
            await message.reply("Нових заяв наразі немає!")
    else:
        await message.reply("Ви не маєте на це прав!", reply_markup=ReplyKeyboardRemove())
```

3.4 Тестування програмного забезпечення

Для тестування необхідно повністю перевірити функціонал програмного забезпечення.

Почати слід із функціоналу адміністратора, тобто, перегляду нових користувачів і перегляду нових заяв.

Перегляд нових заяв виконується через кнопку «Нові заяви» на головному екрані (рис. 3.4). Вивід заяв зображено на рисунку 3.5.

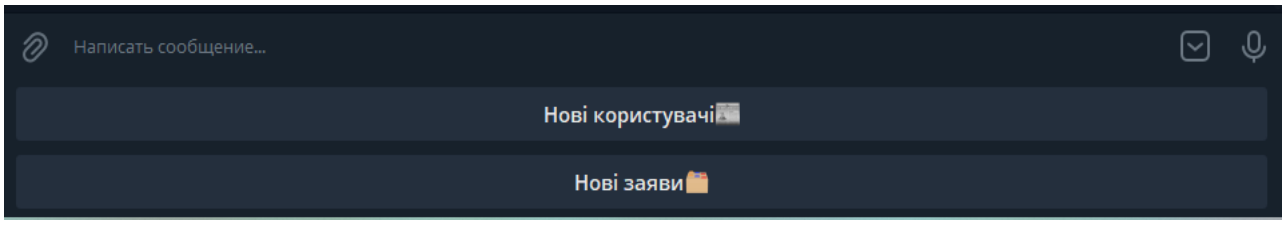


Рис. 3.4 — Меню адміністратора

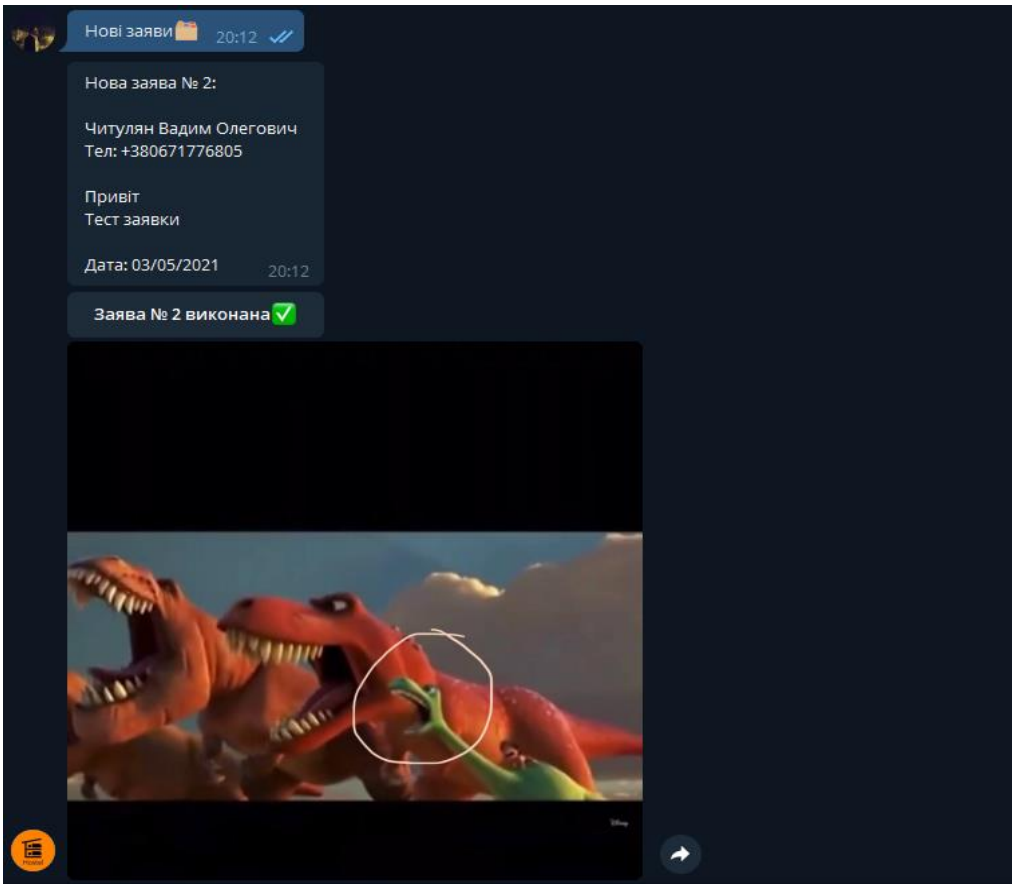


Рис. 3.5 — Вивід нових заяв

Вивід нових заяв працює корректно. Далі слід перевірити перегляд нових користувачів.

Перелік нових користувачів виводиться корректно (рис. 3.6).

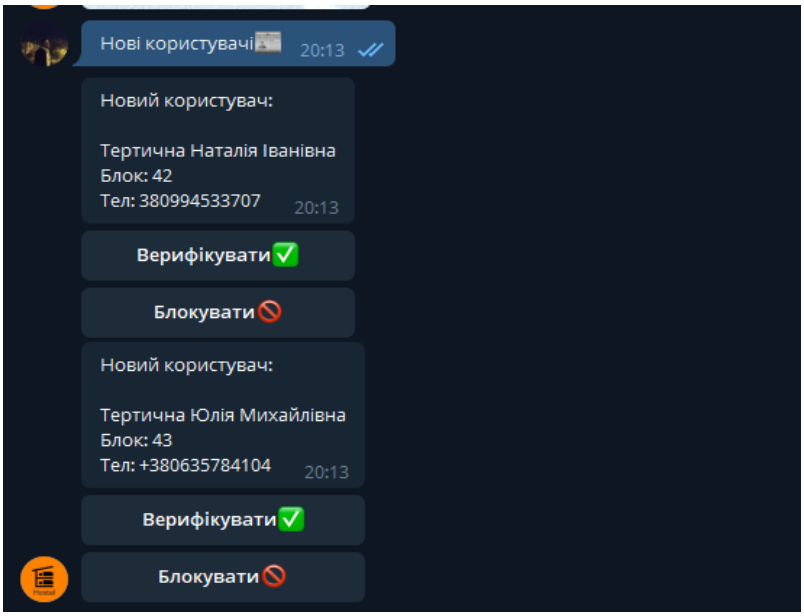


Рис. 3.6 — Вивід нових користувачів

Далі слід перевірити реєстрацію нового користувача. Для цього вводимо всі необхідні данні про користувача. (рис. 3.7 — 3.9) На рисунках можна побачити, що бот оброблює всі виключні ситуації, які може спровокувати користувач.

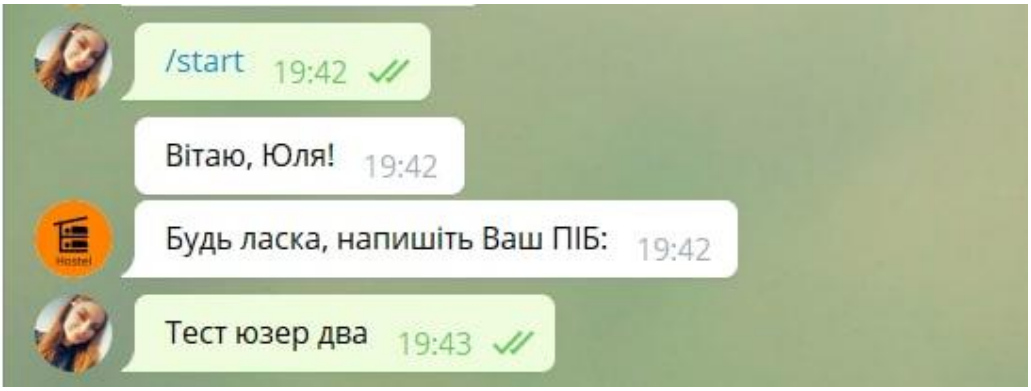


Рис. 3.7 — Введення імені некоректного ПІБ

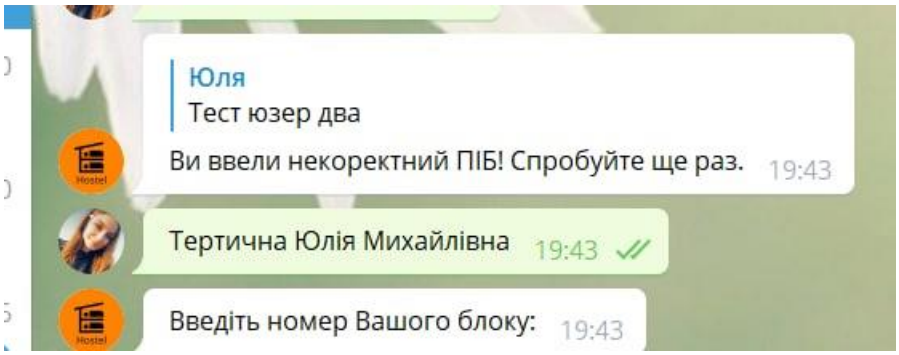


Рис. 3.8 — Введення коректного ПІБ

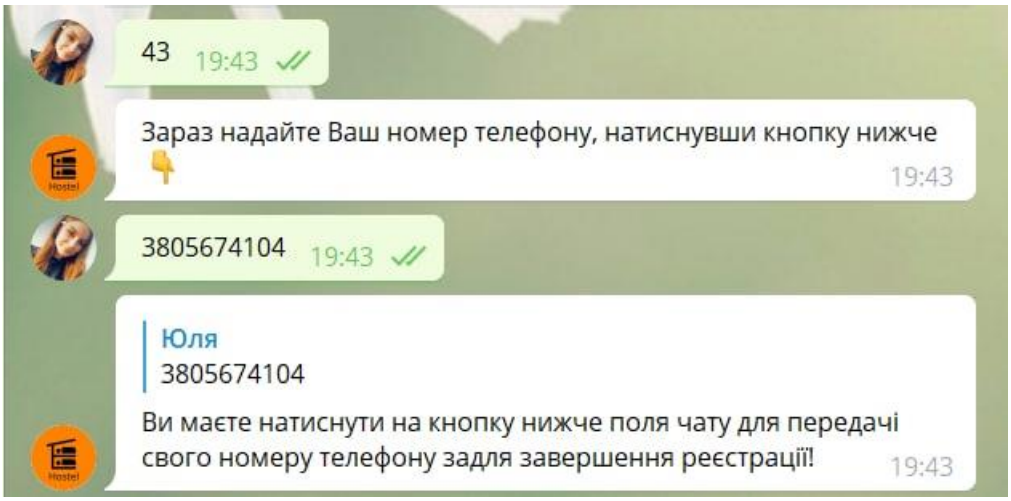


Рис. 3.9 — Надання номера в некоректній формі

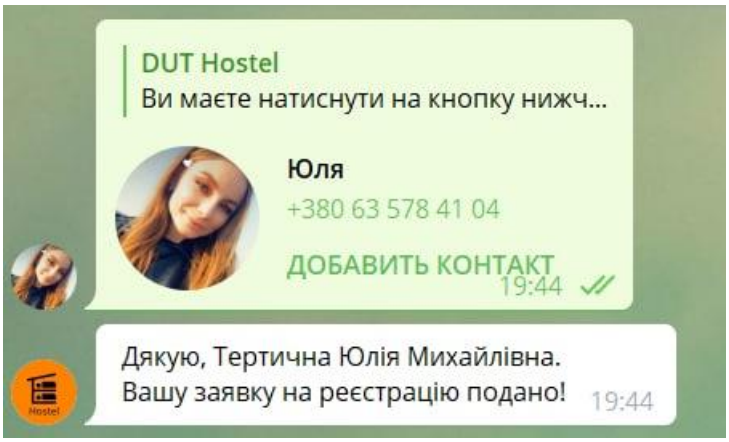


Рис. 3.10 — Введення номера телефону в коректній формі

3.5 Інструкція користувача

Для того, щоб зареєструватися в боті слід виконати наступні дії:

- Ввести команду «старт»
- Ввести ПІБ
- Надати номер телефону
- Ввести номер блоку
- Очікувати підтвердження

Для того, щоб подати заяву слід виконати наступні дії:

- Натиснути кнопку «Створити заяву»
- Натиснути кнопку «Ввести текст»
- Ввести опис проблеми
- Натиснути кнопку «Додати фото»
- Додати фото проблеми
- Натиснути «Відправити заяв

ВИСНОВКИ

В рамках випускної кваліфікаційної роботи був розроблений та впроваджений чат-бот. Ідея дипломного проекту полягала в створенні Telegram-боту для студентського гуртожитку Державного університету телекомунікацій.

На основі аналізу були сформовані наступні вимоги а саме: проаналізувати предметну область, обрати засоби реалізації, спроектувати і реалізувати телеграм-бота для оптимізації процесів господарської діяльності гуртожитку.

Під час аналізу предметної області було обрано та описано програмні засоби для розробки чат-бота: асинхронний фреймворк Aiogram та середовище для розробки PyCharm Community Edition. Для системи управління базами даних використано PostgreSQL та хмарна платформа Amazon AWS.

Додаток є актуальним для адміністрації гуртожитку та студентів, що проживають у ньому. Чат-бот націлений на спрощення та полегшення введення журналу заяв та інцидентів шляхом перенесення його в електронний вигляд за допомогою месенджера Telegram.

Завдяки чіткому виконанню завдань, які були поставлені на початку роботи, було отримано вичерпні знання щодо методів і засобів розробки телеграм-ботів, за допомогою яких було спроектовано і розроблено повноцінного телеграм-бота для оптимізації господарської роботи гуртожитку, який повністю задовольняє вимоги, готовий до впровадження і використання в реальних умовах.

Робота пройшла апробацію на конференції : Застосування програмного забезпечення в інфокумунікаційних технологіях // Розробка програмного забезпечення для інфокумунікацій. Збірник тез. – К.: ДУТ, 2021. — С. 28 – 29, м. Київ, 12.02.2020.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. "Telegram Messenger". Google Play. Retrieved 18 March 2021.
2. "Telegram X". Google Play. Retrieved 25 February 2021.
3. "Telegram Messenger". App Store. Retrieved 26 March 2021.
4. "Telegram Desktop latest release". GitHub. Retrieved 20 March 2021.
5. "Telegram Desktop". Microsoft Store. Retrieved 20 March 2021.
6. "Telegram". Mac App Store. Retrieved 18 March 2021.
7. "Telegram Beta 2 – HockeyApp". rink.hockeyapp.net.
8. "Join the Telegram Messenger beta". testflight.apple.com.
9. "Version history". Telegram.
10. "Telegram Swift – HockeyApp". rink.hockeyapp.net.
11. "Telegram Messenger". Telegram Messenger LLP. Retrieved 25 February 2021.
12. "List of Telegram applications". 6 February 2014.
13. "Telegram FAQ". Telegram. Retrieved 29 March 2021.
14. "Telegram Company profile". 12 December 2019.
15. "Telegram introduces end-to-end encrypted video calls". The Next Web. Retrieved 29 March 2021.
16. EWDN, Editor (30 August 2013). "Russia's Zuckerberg launches Telegram, a new instant messenger service". Reuters. Retrieved 8 November 2020.
17. "Meet Telegram, A Secure Messaging App From The Founders Of VK, Russia's Largest Social Network". TechCrunch. Retrieved 8 November 2020.
18. "'Nobody can block it': how the Telegram app fuels global protest". The Guardian. Retrieved 7 November 2020.
19. "The Evolution of Telegram". Telegram. Retrieved 4 January 2021.
20. "Telegram Applications". Telegram.

21. "FAQ for the Technically Inclined". core.telegram.org. Retrieved 1 October 2017.
22. "Telegram hits 500M monthly active users". Telegram. Retrieved 12 January 2021.
23. "Durov Telegram". Telegram. 8 February 2021. Retrieved 10 February 2021.
24. Hakim, Danny (2 December 2014). "Once Celebrated in Russia, the Programmer Pavel Durov Chooses Exile". The New York Times. Retrieved 19 November 2015.
25. Shu, Catherine (27 October 2013). "Meet Telegram, A Secure Messaging App From The Founders Of VK, Russia's Largest Social Network". TechCrunch. Retrieved 18 March 2016.
26. Telegram F.A.Q, "...making profits will never be an end-goal for Telegram."
27. Why Telegram has become the hottest messaging app in the world, The Verge. Retrieved 25 February 2014. "Telegram operates as a non-profit organization, and doesn't plan to charge for its services."
28. Dewey, Caitlin (23 November 2015). "The secret American origins of Telegram, the encrypted messaging app favored by the Islamic State". The Washington Post. Retrieved 31 March 2018.
29. "Telegram - Android Apps on Google Play". play.google.com. Retrieved 19 November 2015.
30. "Telegram Messenger on the App Store". App Store. Retrieved 19 November 2015.
31. Thornhill, John (3 July 2015). "Lunch with the FT: Pavel Durov". Financial Times. Retrieved 19 November 2015.
32. Brandom, Russell (6 October 2014). "Surveillance drives South Koreans to encrypted messaging apps". The Verge. Retrieved 19 November 2015.

33. Turton, William (29 September 2017). "What isn't Telegram saying about its connections to the Kremlin?". The Outline. Retrieved 11 October 2017.
34. "Telegram app free-speech advocate no stranger to Apple-FBI woes". 23 February 2016 – via www.reuters.com.
35. "This $5 Billion Encrypted App Isn't for Sale at Any Price". Bloomberg. 12 December 2017. Retrieved 22 December 2017.
36. Telegram Hits 35M Monthly Users, 15M Daily With 8B Messages Received Over 30 Days, TechCrunch, 24 March 2014
37. Telegram Reaches 1 Billion Daily Messages, Telegram, 8 December 2014
38. Telegram Hits 2 Billion Messages Sent Daily, Telegram, 13 May 2015
39. Lomas, Natasha (21 September 2015). "Telegram Now Seeing 12BN Daily Messages, up From 1BN in February". Techcrunch. Retrieved 19 November 2015.
40. Dunham, Ken; Melnick, Jim (2009). Malicious Bots: An outside look of the Internet. CRC Press. ISBN 9781420069068.
41. Zeifman, Igal. "Bot Traffic Report 2016". Incapsula. Retrieved 1 February 2017.
42. "Twitter Followers Guide". 20 November 2019
43. Howard, Philip N (18 October 2018). "How Political Campaigns Weaponize Social Media Bots". IEEE Spectrum.
44. Ferrara, Emilio; Varol, Onur; Davis, Clayton; Menczer, Filippo; Flammini, Alessandro (2016). "The Rise of Social Bots". Communications of the ACM. 59 (7): 96–104. arXiv:1407.5225. doi:10.1145/2818717. S2CID 1914124.
45. Alessandro, Bessi; Emilio, Ferrara (2016-11-07). "Social Bots Distort the 2016 US Presidential Election Online Discussion". SSRN 2982233.
46. "Touch Arcade Forum Discussion on fraud in the Top 25 Free Ranking".

47. "App Store fake reviews: Here's how they encourage your favourite developers to cheat". Electricpig. Archived from the original on 2017-10-18. Retrieved 2014-06-11.
48. "Facebook Messenger Hits 100,000 bots". 2017-04-18. Retrieved 2017-09-22.
49. Murray Newlands. "These Chatbot Usage Metrics Will Change Your Customer Service Strategy". Retrieved 2018-03-08.
50. "How companies are using chatbots for marketing: Use cases and inspiration - MarTech Today". MarTech Today. 2018-01-22. Retrieved 2018-04-10.
51. Dima Bekerman: How Registration Bots Concealed the Hacking of My Amazon Account, Application Security, Industry Perspective, December 1st 2016, In: www.Imperva.com/blog
52. Carr, Sam (July 15, 2019). "What Is Viewbotting: How Twitch Are Taking On The Ad Fraudsters". PPC Protect. Retrieved 19 September 2020.
53. Lewis, Richard (March 17, 2015). "Leading StarCraft streamer embroiled in viewbot controversy". Dot Esports. Retrieved 19 September 2020.
54. Safruti, Ido (June 19, 2017). "Why Detecting Bot Attacks Is Becoming More Difficult". DARKReading.
55. Holiday, Ryan (January 16, 2014). "Fake Traffic Means Real Paydays". BetaBeat. Archived from the original on 2015-01-03. Retrieved 2014-04-28.
56. von Lipinski, Percy (28 May 2013). "CNN's iReport hit hard by pay-per-view scandal". PulsePoint. Archived from the original on 18 August 2016. Retrieved 21 July 2016.
57. Heo, Hyun-Hee; Kim, Min-Sun (2013). "The effects of multiculturalism and mechanistic disdain for robots in human-to-robot communication scenarios". Interaction Studies. 14 (1): 81–106. doi:10.1075/is.14.1.06heo. ISSN 1572-0373.
58. "Bots to Buy Shoes With". GeoSurf. 2018-02-15. Retrieved 2020-04-26.

ДОДАТКИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Розробка chat-bot для оптимізації господарської роботи гуртожитку мовою Python.

Виконала студентка 4 курсу
групи ПД-44
Тертична Юлія Михайлівна
Керівник роботи
Негоденко Олена Василівна

Київ – 2021

МЕТА, ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

- **Мета роботи** - розробка комплексного Telegram-бота для оптимізації господарської діяльності гуртожитку на мові Python.
- **Об'єкт дослідження -** методи та засоби створення чат-ботів на базі платформи Telegram.
- **Предмет дослідження -** телеграм-бот для оптимізації господарської діяльності в гуртожитку.

АНАЛОГИ

DUT Hostel

- Telegram Messenger;
- Швидкість обробки інформації;
- Особливості отримання даних;
- Створення фотографій;

Закарпаттяобленерго

- Viber, Telegram та Facebook;
- повторні підтвердження особових рахунків;

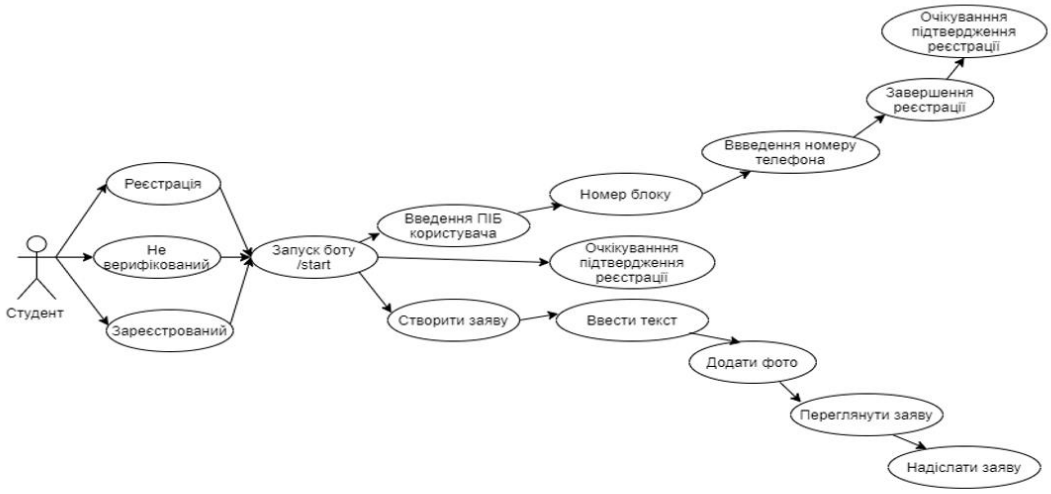
ТЕХНІЧНІ ЗАВДАННЯ

Чат-бот має виконувати наступні функції :

- Можливість реєстрації та авторизації в системі;
- Отримання даних від користувача та їх зберігання;
- Надсилання фото та опис несправності об’єкту;
- Надсилання даних про користувача адміністратору;
- Можливість адміна блокувати або верифікувати користувача;

ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

Діаграма користувача



Діаграма користувача (admin)

АПРОБАЦІЇ

Результати дослідження бакалавроської роботи апробовані:

1. Застосування програмного забезпечення в інфокумунікаційних технологіях // Розробка програмного забезпечення для інфокумунікацій. Збірник тез. 12.02.2020, ДУТ, м. Київ — К.: ДУТ, 2021. — С. 28 – 29.
2. XII міжнародна науково-технічна конференція студентства та молоді «світ інформації та телекомунікацій» // Сучасні інформаційні технології. Збірник тез. 21.05.2021, ДУТ, м. Київ — К.: ДУТ, 2021.

ВИСНОВКИ

В рамках випускної кваліфікаційної роботи був розроблений та впроваджений чат-бот. В процесі роботи над даним проектом було виконано наступні завдання:

- Проаналізовано предметну область;
- Обрати засоби реалізації;
- Спроектовано та реалізовано телеграм-бота;
- Побудовано діаграму варіантів використання;
- Оглянуто основні механізми роботи;
- Проведено тестування;

ДЯКУЮ ЗА УВАГУ!