МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВІДОКРЕМЛЕНИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ

«ПОЛТАВСЬКИЙ ПОЛІТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ

НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Циклова комісія дисциплін програмної інженерії

**ЗВІТ**

**з технологічної практики**

«Розробка і супроводження програмного продукту»

на тему \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

Телеграм бот «Система заявок на заміну лічильників»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Виконав: здобувач освіти 4 курсу, групи \_\_\_\_\_\_\_  
 спеціальності 121   
 Інженерія програмного забезпечення

Каверін Д.О

44

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (прізвище та ініціали)

Бабич О.В

Керівник \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
 (підпис) (прізвище та ініціали)

Полтава – 2023

**ЗМІСТ**

[ВСТУП 3](#_Toc155348568)

[1. ПОСТАНОВКА ЗАВДАННЯ 4](#_Toc155348569)

[1.1. Основні вимоги до продукту 4](#_Toc155348570)

[1.2. Вимоги до інтерфейсу 5](#_Toc155348571)

[2. ПЛАНУВАННЯ СИСТЕМИ 7](#_Toc155348572)

[2.1. Архітектура системи 8](#_Toc155348573)

[2.2. Тестування 9](#_Toc155348574)

[2.3. Інструкція з використання системи 10](#_Toc155348575)

[ВИСНОВКИ 11](#_Toc155348576)

[СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 12](#_Toc155348577)

[ДОДАТОК А. ПОРІВНЯЛЬНА ХАРАКТЕРИСТИКА 13](#_Toc155348578)

[ДОДАТОК Б. UML ДІАГРАМА ПРЕЦЕДЕНТІВ 14](#_Toc155348579)

[ДОДАТОК В. ПРОТОТИП ІНТЕРФЕЙСУ 15](#_Toc155348580)

[ДОДАТОК Г. UML ДІАГРАМА КЛАСІВ 16](#_Toc155348581)

[ДОДАТОК Д. ВИХІДНІ КОДИ 17](#_Toc155348582)

[ДОДАТОК Е. ЗНІМКИ ЕКРАНУ 20](#_Toc155348583)

# ВСТУП

Місцем моєї технологічної практики було АТ “ПОЛТАВАОБЛЕНЕРГО”.

Компанія працює на ринку послуг з 28 квітня 1995 року. [1]

Основними напрямками її діяльності є виробництво, розподілення та постачання електроенергії.

Послугами є ремонт і технічне обслуговування електророзподільної та контрольної апаратури.

В ході практики я ознайомився з роботою відділу Групи формування корисного відпуску , де переді мною було поставлено завдання ознайомитися з роботою, яку виконують працівники відділу, заносити інформацію до програмного комплексу Utilty ЕnеrguSuite, таку як, заповнення особової картки споживача:

* занесення ІПН ( Ідентифікаційний код) та паспортних даних;
* внесення показників лічильника знятих контролером ГФКВ (Група формування корисного відпуску);
* занесення телефонів споживачів.

Ознайомився з роботою бази даних RobutPobut, за допомогою якої витягується необхідна інформація для подальшого опрацювання та аналізу її в Excel. А також потрібно ознайомитися з роботою сайту по продажу електролічильників та сайту для передачи показів електроенергії.

Основним завданням є розробка і супроводження програмного продукту

Telegram bot «Система заявок на заміну лічильників».

Для виконання поставленого завдання мені стали в нагоді знання, вміння й навички з користування Excel та Word.

Під час практики я працював з базою даних RobutPobut та Utilty ЕnеrguSuite та вивчив мову програмування Python[2], працював в загальному доступі Excel, навчився передавати покази електроенергії через сайт АТ Полтаваобленерго і працював в програмі Utilty.

# 1. ПОСТАНОВКА ЗАВДАННЯ

На практиці в АТ Полтаваобленерго переді мною було поставлено завдання розробити Телеграм бот «Система заявок на заміну лічильників»

Вимогою, яку поставив замовник, має бути зручність використання.

Терміни розробки з 4 по 25 грудня 2023 року.

Функціональність програми: за допомогою бота подавати електронні заявки.

Для створення цієї програми я обрав середовище розробки Visual Studio.[2]

Visual Studio — серія продуктів фірми Майкрософт, які містять інтегроване середовище розробки програмного забезпечення та низку інших інструментальних засобів.

Для написання програми я обрав мову програмування Python.[2]

Python — високорівнева мова програмування, яку називають [другою за популярністю](https://leftronic.com/blog/python-statistics/) в світі. Її використовують для розробки веб застосунків, програмного забезпечення, машинного навчання. Python застосовують для вирішення робочих завдань у компаніях Google, Instagram, Facebook, IBM, NASA, Dropbox, Netflix та інших. Розробники цінують цю мову програмування за простоту у вивченні, ефективність та мультиплатформність.

Телеграм боту по заміні лічильників ще не має, але є телеграм бот по передачі показів електролічильника.

## 1.1. Основні вимоги до продукту

Функціональні вимоги — це вимоги до програмного забезпечення, які описують внутрішню роботу системи, її поведінку: обчислення даних, маніпулювання даними, обробка даних та інші специфічні функції, які має виконувати система.[3]

Чат-бот повинен бути спроектованим так, щоб мати змогу масштабуватися, розширювати функціональні можливості, підтримувати високу швидкість відповіді та бути відмовостійким до великих навантажень від користувачів, моделювати реальну розмову з клієнтом.[3]

ФУНКЦІОНАЛЬНІ ВИМОГИ:

* створення заявки;
* підтвердження інформації;
* обробка заявок;
* cтатус заявки.

НЕФУНКЦІОНАЛЬНІ ВИМОГИ:

* безпека;
* зручність використання.

ПЕРЕЛІК РЕАЛІЗОВАНИХ ЗА ЧАС ПРАКТИКИ ФУНКЦІОНАЛЬНИХ ВИМОГ[Додаток Б]

* створення заявки;
* підтвердження інформаціїї;
* обробка заявки.
* cтатус заявки.

ПЕРЕЛІК НЕРЕАЛІЗОВАНОЇ ФУНКЦІОНАЛЬНОСТІ

Нереалізованої функціональності не має.

## 1.2. Вимоги до інтерфейсу

Інтерфейс — сукупність засобів і правил, що забезпечують взаємодію комп'ютерів, периферійних пристроїв, пристроїв вводу/виводу та/або комп'ютерних програм. Сукупність описів і узгоджень щодо процедури передачі керування в підпрограму та повернення до вихідної програми.

Види інтерфейсів: графічний, текстовий, SILK.[4]

Вимоги: інтерфейс повинен бути консистентним, тобто, ґрунтуватись на використанні відомих, загальноприйнятих методів і засобів представлення інформації; в ідеалі процес взаємодії користувача з системою не повинен представляти ніяких труднощів.

Для Телеграм боту «Система заявок на заміну лічильників» було поставлено такі вимоги до інтерфейсу:

забезпечити простий та легкий у використанні інтерфейс; користувачі повинні швидко зрозуміти, як подати заявку без додаткових пояснень; забезпечити підтримку різних мов для максимального охоплення аудиторії; надати користувачам повідомлення про будь-які помилки та поради, як їх виправити; можливість звертання за підтримкою або отримання додаткової інформації безпосередньо через Telegram; дозволяти користувачам долучати файли (зображення, документи) до своїх заявок для надання додаткової інформації; користувачі можуть використовувати Telegram на різних платформах (Android, iOS, веб-версія), тому потрібно забезпечити сумісність із всіма ними.

# 

# 2. ПЛАНУВАННЯ СИСТЕМИ

Життє́вий цикл програ́много забезпе́чення — сукупність окремих етапів робіт, що проводяться у заданому порядку протягом періоду часу, який починається з вирішення питання про розроблення програмного забезпечення і закінчується припиненням використання програмного забезпечення.[4]

Серед моделей життєвого циклу програмного забезпечення найбільш відомі такі:

каскадна модель, ітераційні модель, інкрементна модель, спіральна модель.

Для створення системи було обрано каскадний життєвий цикл.

Водоспадна модель життєвого циклу ПЗ — послідовний метод розробки програмного забезпечення, названий так через діаграму, схожу на водоспад

РОЗРОБКА ТА НАЛАШТУВАННЯ

Визначення функціональності: які конкретні завдання буде виконувати ваш бот, які команди буде обробляти, і як буде виглядати процес подачі заявок.

Створення бота: використовуючи Visual Studio для розробки.

Встановлення середовища: необхідні бібліотеки, такі як python-telegram-bot.

РЕАЛІЗАЦІЯ ФУНКЦІОНАЛЬНОСТІ БОТА

Створення команд та обробників: реалізувати обробників для команд, які бот буде обробляти, зокрема для команди /submit або іншої, яка відповідає за подачу заявок.

ЗАПУСК ТА ТЕСТУВАННЯ

Запуск бота: перевірити, чи він взаємодіє з Telegram коректно.

Тестування функціональності: перевірити роботу кожної команди, включаючи подачу заявок. Впевніться, що обробники працюють вірно.

ОПТИМІЗАЦІЯ ТА РОЗШИРЕННЯ

Оптимізація коду: переглянути код на предмет можливостей оптимізації та покращень.

Додавання функцій: додати нові функції, такі як збереження заявок у базі даних, відправлення повідомлень адміністраторам тощо.

Підтримка користувачів: забезпечити засоби для зворотного зв'язку та підтримки користувачів, які мають питання або стикаються з проблемами.

ОНОВЛЕННЯ ТА СКЛАДАННЯ

Оновлення бота: вносити оновлення для вдосконалення функціональності та безпеки.

Складання в готовий продукт: завершити розробку та впровадження бота в роботу.

Створення звітів: створювати регулярні звіти для власного аналізу та для інформування адміністраторів чи клієнтів. Для керування проектом було використано GitHub.

## 2.1. Архітектура системи

В розробці програмного продукту була використана монолітна архітектура.

Монолітна архітектура (Monolithic Architecture) – це традиційний підхід до розробки програмного забезпечення, при якому весь додаток розробляється як одна єдина технологічна система.[5]

Telegram Bot Application:

центральна частина системи, яка взаємодіє з Telegram Bot API та обробляє вхідні повідомлення та команди користувачів.

Функції: обробка команд /start, /submit та інших команд бота;

отримання текстових повідомлень від користувачів та обробка їх як заявок;

валідація та обробка заявок від користувачів;

збереження даних про заявки в базі даних.

Database Storage: база даних для збереження інформації про користувачів, заявки та інші необхідні дані.

Таблиці:

Users: зберігає інформацію про користувачів бота.

Applications: зберігає дані про подані заявки.

User Interface (Telegram Client):

Logging: система логування для реєстрації дій та помилок в системі.

Запис логів для відстеження подій та виявлення помилок. [Додаток Г]

## 

## 2.2. Тестування

Є 4 види тестування: функціональне тестування, нефункціональне тестування, тестування змін, структурне тестування.[6]

Мета функціонального тестування — виявлення невідповідностей між реальною поведінкою реалізованих функцій і очікуваною поведінкою відповідно до специфікації і вимог. Функціональні тести повинні охоплювати всі реалізовані функції з урахуванням найбільш ймовірних типів помилок.

Було проведено функціональне тестування: перевірка основних функцій бота, таких як обробка команд, відправлення повідомлень, обробка вхідних даних від користувачів, перевірка всіх функціональних вимог до бота.

Тестування взаємодії з Telegram API: cимуляція вхідних повідомлень та команд через Telegram API для перевірки відповідей бота.

Тестування валідації та обробки даних: перевірка валідації введених даних користувачів, тестування обробки помилок та виведення коректних повідомлень про помилки.

Тестування збереження даних: перевірка збереження даних про користувачів та їх заявки в базі даних. Тестування валідності та точності збережених даних.

Тестування висновків: перевірка виведення коректних висновків та повідомлень користувачам.

## 2.3. Інструкція з використання системи

Для того щоб почати працювати з телеграм ботом потрібно зайти в телеграм та ввести в пошуку ЛічильникBot. Зайшовши до боту, потрібно натиснути кнопку розпочати і буде виведено привітання, потім заходимо в меню і натискаємо submit подати заявку. Після того, як натискаємо подати заявку у Вас запросить Ім’я, електронну пошту, номер телефону та адресу.

Відповівши на всі запитання заявка буде сформована і збережена, а протягом доби вам буде надіслано відповідь на електронну пошту.

Також з’явиться вікно з посиланням на тех-підтримку.[Додаток Е]

# ВИСНОВКИ

Місцем моєї технологічної практики було АТ “ПОЛТАВАОБЛЕНЕРГО”.

Під час технологічної практики було розроблено Телеграм бот «Система заявок на заміну лічильників». Метою системи було спростити спосіб подачи заяв на заміну лічильників.

Для створення було використано середовище програмування Visual Studio та мову програмування Python.

Для зберігання даних про заявки та керування користувачами був використаний сайт Sendpulse та бібліотеку python-telegram-bot.

В ході практики я ознайомився з роботою відділу Групи формування корисного відпуску, де переді мною було поставлено завдання ознайомитися з роботою, яку виконують працівники відділу, заносити інформацію до програмного комплексу Utilty ЕnеrguSuite, таку як, заповнення особової картки споживача.

Ознайомився з роботою бази даних RobutPobut, за допомогою якої витягується необхідна інформація для подальшого опрацювання та аналізу її в Excel. А також ознайомився з роботою сайту по продажу електролічильників та сайту для передачи показів електроенергії.

Під час практики я працював з базою даних RobutPobut та Utilty ЕnеrguSuite та вивчив мову програмування Python, працював в загальному доступі Excel, навчився передавати покази електроенергії через сайт АТ Полтаваобленерго і працював в програмі Utilty.

# СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1.ПОЛТАВАОБЛЕНЕРГО: [Веб-сайт]. URL: <https://www.poe.pl.ua/peredacha-pokaziv/> (дата звернення: 01.12.2023).

2.FreeCodeCamp.org:[Веб-сайт].URL: <https://www.freecodecamp.org/learn/python-for-everybody/#python-for-everybody> (дата звернення: 06.12.2023).

3.Visure: [Веб-сайт]. URL: <https://visuresolutions.com/uk/blog/functional-requirements/> (дата звернення: 13.01.2024).

4.Evergreen: [Веб-сайт]. URL: <https://evergreens.com.ua/ua/articles/software-development-metodologies.html> (дата звернення: 12.01.2024).

5.Qalight: [Веб-сайт]. URL: <https://qalight.ua/baza-znaniy/shho-take-monolitna-arhitektura/> (дата звернення: 12.01.2024).

6.Cases.media: [Веб-сайт]. URL: <https://cases.media/article/sho-take-interfeis-riznovidi-interfeisiv> (дата звернення: 14.12.2023).

7.Методичка:[Інтернет-портал].URL: <https://classroom.google.com/u/0/c/NjM5OTgwODA1MTAx/m/NjQwNjQ2NTI4NzI1/details?hl=ru> (дата звернення: 10.12.2023).

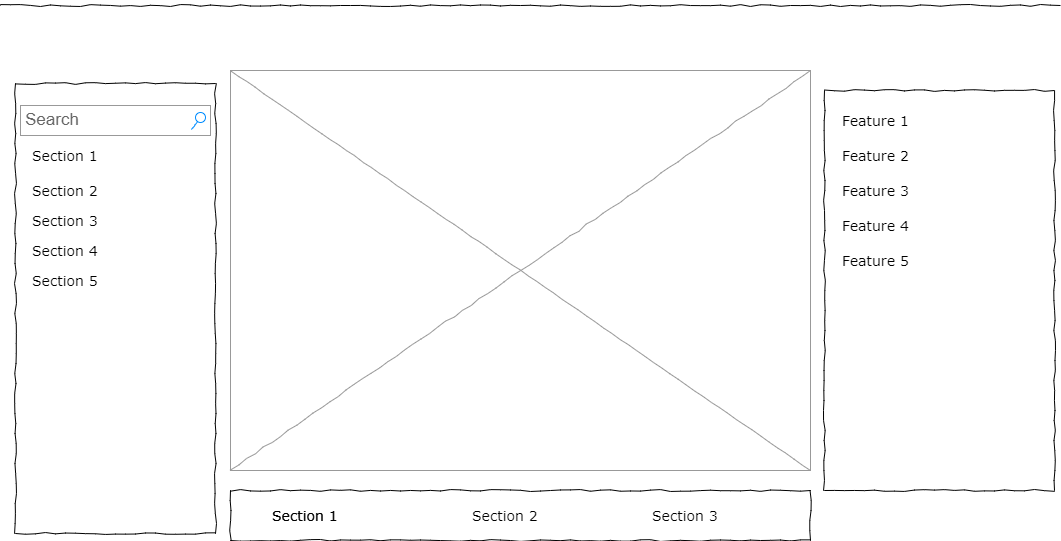
# ДОДАТОК А. ПОРІВНЯЛЬНА ХАРАКТЕРИСТИКА

|  |  |  |  |
| --- | --- | --- | --- |
|  | Програмний продукт 1 | Програмний продукт 2 | Створений програмний продукт |
| Адаптивність | + | + | + |
| Багатокористувацький режим | + | + | + |
| Інтуїтивно-зрозумілий інтерфейс користувача | + | + | + |
| Наявність української мови | + | - | + |
| Можливість імпорту/експорту даних | + | - | + |
| Крос-платформність (наявність версій для Linux/MacOS) | + | - | - |
| Автентифікація з використанням соціальних облікових записів (Google/Facebook/Twitter тощо) | - | + | - |
| Наявність довідкових матеріалів та документації | - | + | - |

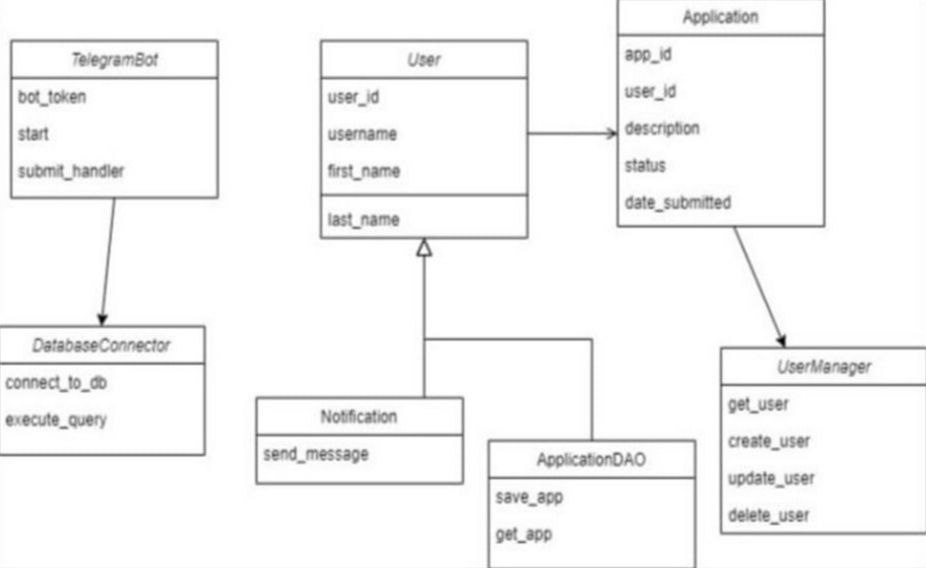
# ДОДАТОК Б. UML ДІАГРАМА ПРЕЦЕДЕНТІВ



# ДОДАТОК В. ПРОТОТИП ІНТЕРФЕЙСУ



# ДОДАТОК Г. UML ДІАГРАМА КЛАСІВ



# 

# ДОДАТОК Д. ВИХІДНІ КОДИ

Важливі частини коду.

import re

def get\_user\_data():

# Запитати ім'я користувача

name = input("Введіть ваше ім'я: ")

# Запитати email

while True:

email = input("Введіть ваш email: ")

if re.match(r"[^@]+@[^@]+\.[^@]+", email):

break

else:

print("Введіть коректний email.")

# Запитати номер телефону

while True:

phone\_number = input("Введіть ваш номер телефону: ")

if re.match(r"\d{10}", phone\_number):

break

else:

print("Введіть коректний номер телефону (10 цифр без пробілів та інших символів).")

# Запитати адресу

address = input("Введіть вашу адресу: ")

# Вивести отримані дані

print("Ваші дані:")

print("Ім'я:", name)

print("Email:", email)

print("Номер телефону:", phone\_number)

print("Адреса:", address)

# Викликати функцію для отримання даних від користувача

get\_user\_data()

перейти за посиланням для тех підтримки

import webbrowser

url = "https://t.me/Den4kav"

# Відкрити посилання в браузері

webbrowser.open(url)

class ZayavaNaZaminuLichylnyka:

def \_\_init\_\_(self):

self.name = None

self.email = None

self.phone\_number = None

self.address = None

def get\_user\_input(self):

self.name = input("Введіть ваше ім'я: ")

self.email = input("Введіть ваш email: ")

self.phone\_number = input("Введіть ваш номер телефону: ")

self.address = input("Введіть вашу адресу: ")

def validate\_input(self):

# Додайте ваші власні перевірки для коректності введення (наприклад, використання регулярних виразів)

def submit\_request(self):

# Логіка для відправки заявки, наприклад, вивід повідомлення про успішну заявку

print("Ваша заявка на заміну лічильників успішно відправлена.")

print("Ім'я:", self.name)

print("Email:", self.email)

print("Номер телефону:", self.phone\_number)

print("Адреса:", self.address)

# Створити екземпляр класу та виконати процес заявки

zayava = ZayavaNaZaminuLichylnyka()

zayava.get\_user\_input()

zayava.validate\_input()

zayava.submit\_request()

import tkinter as tk

import webbrowser

from telegram import Update

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

def handle\_text(update: Update, context: CallbackContext) -> None:

user\_text = update.message.text

Базова відповідь

response\_text = "Дякуємо за ваше повідомлення: {}".format(user\_text)

Надсилання відповіді користувачеві

update.message.reply\_text(response\_text)

Отримання диспетчера для реєстрації обробників

dp = updater.dispatcher

Реєстрація обробника текстових повідомлень

dp.add\_handler(MessageHandler(Filters.text & ~Filters.command, handle\_text))

Запуск бота

updater.start\_polling()

#Зупинка бота при натисканні Ctrl+C

updater.idle()

if \_\_name\_\_ == '\_\_main\_\_':

main()

class TechSupportWindow:

def \_\_init\_\_(self, master):

self.master = master

master.title("Технічна підтримка")

# Додати кнопку для відкриття веб-браузера

self.support\_button = tk.Button(master, text="Звернутися до технічної підтримки", command=self.open\_support\_link)

self.support\_button.pack(pady=20)

Релізи вихідних кодів розміщені в публічному репозиторії.

Для перегляду відскануйте QR посилання.



# ДОДАТОК Е. ЗНІМКИ ЕКРАНУ

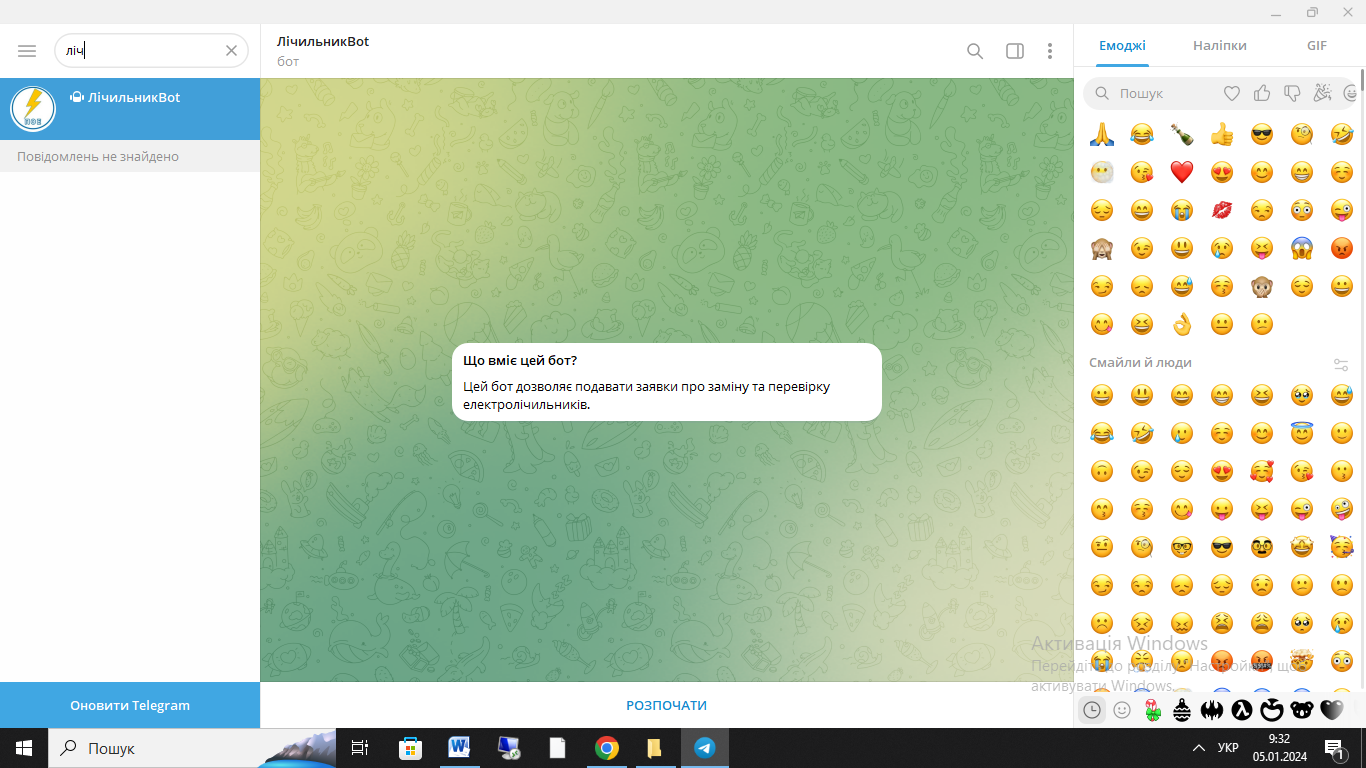


Рисунок Е.1 — пошук бота в телеграм

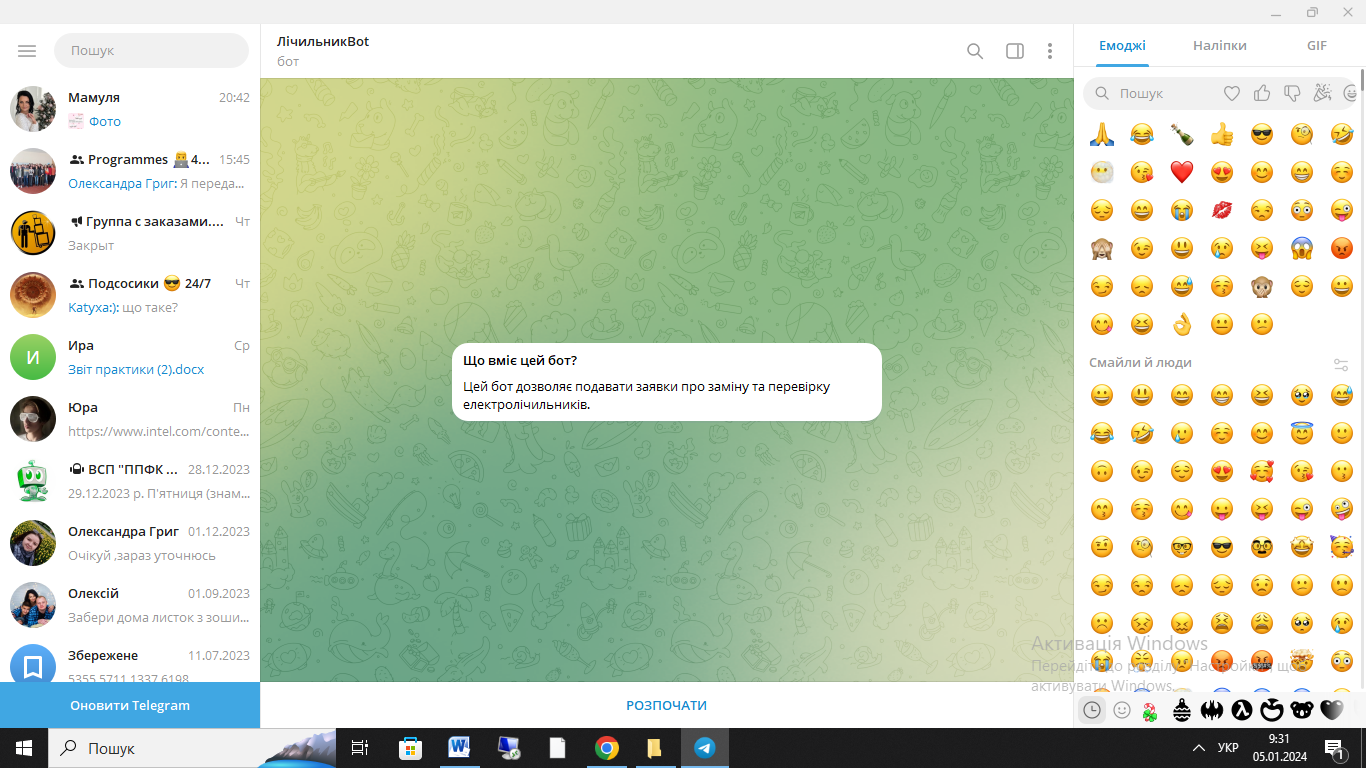


Рисунок Е.2 — ­­початок роботи з ботом

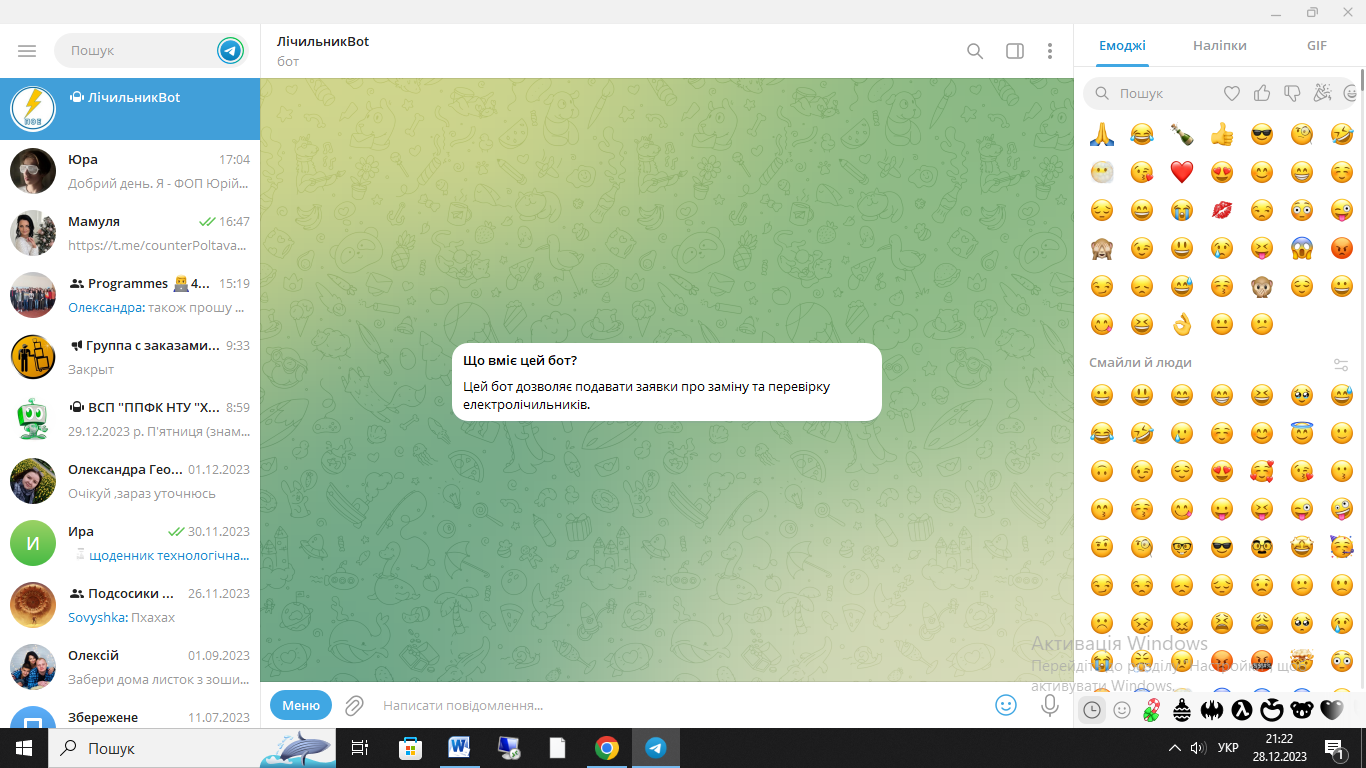


Рисунок Е.3 — Головний екран

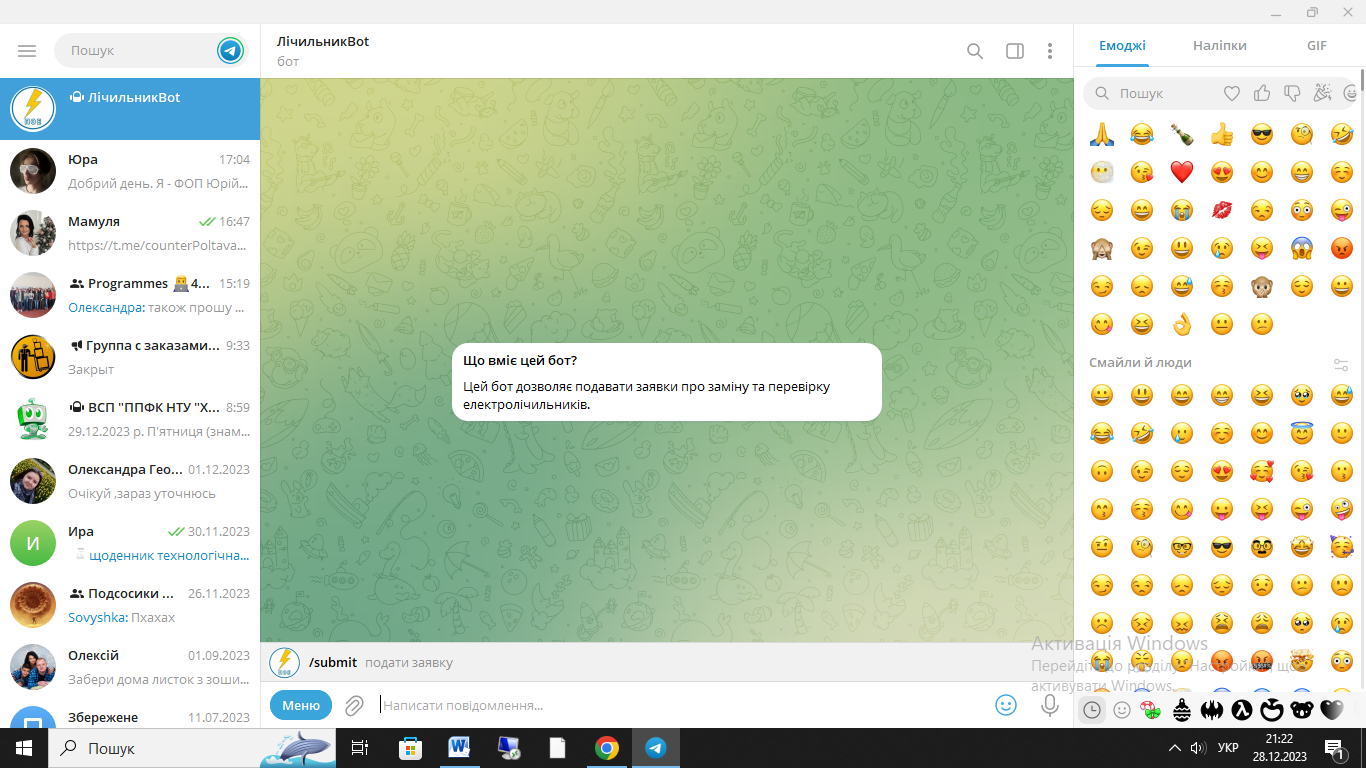


Рисунок Е.4­­­­­­­ — подача заявки

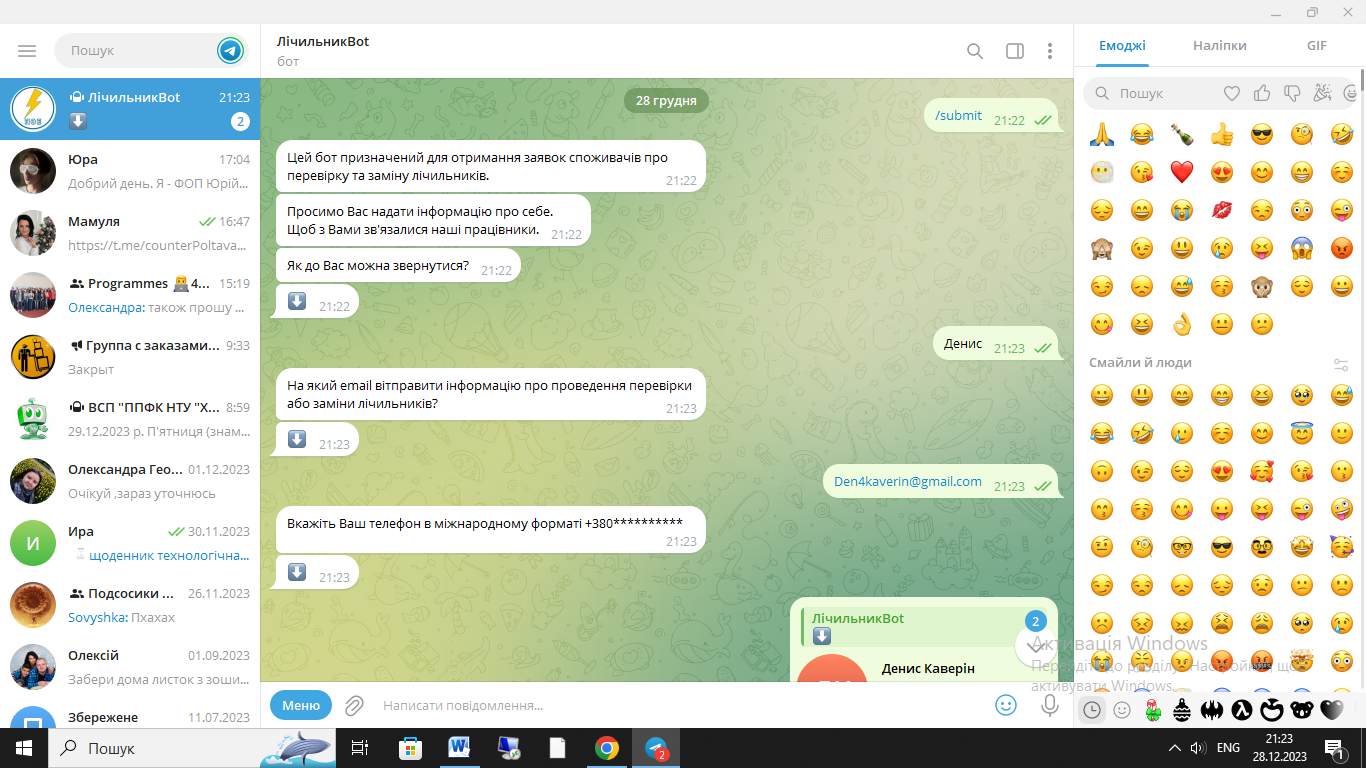


Рисунок Е.5 — заповнення особистих даних

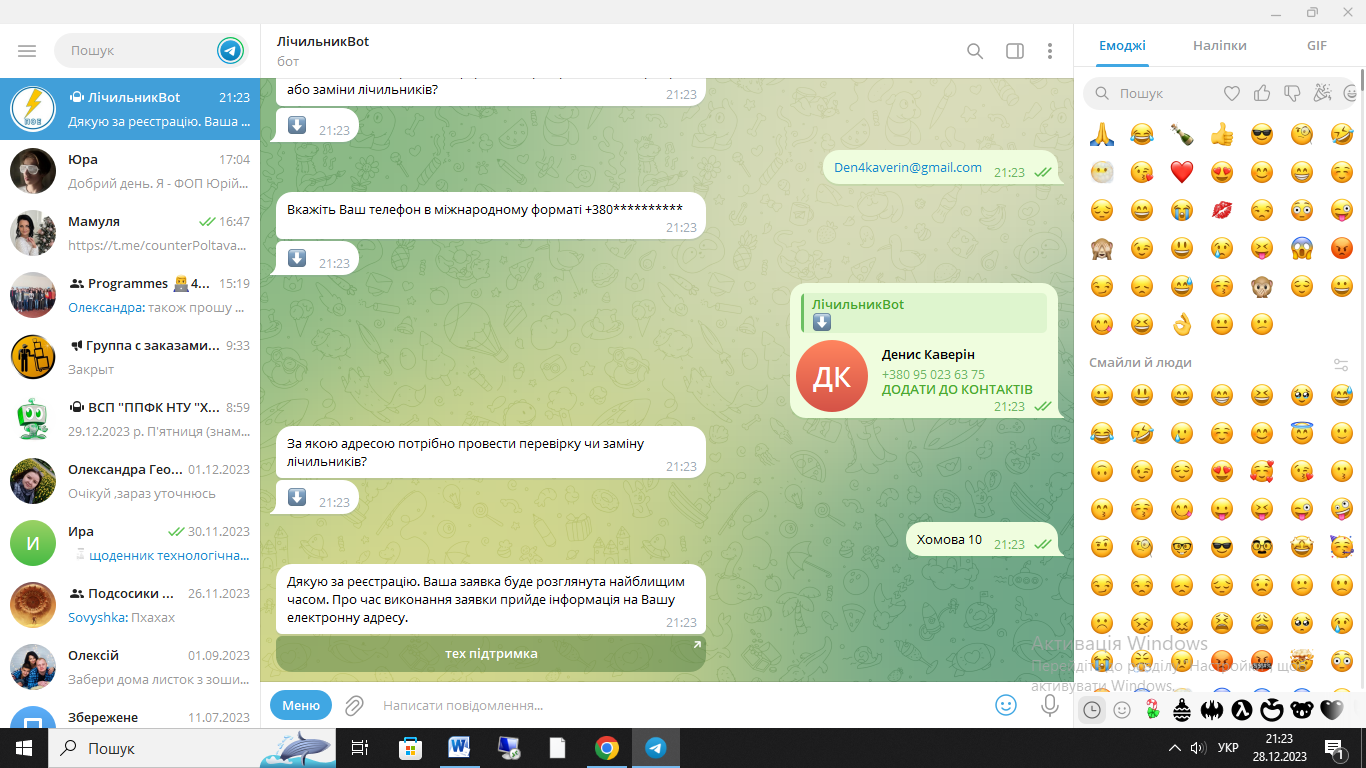


Рисунок Е.6 — Заповнення особистих даних

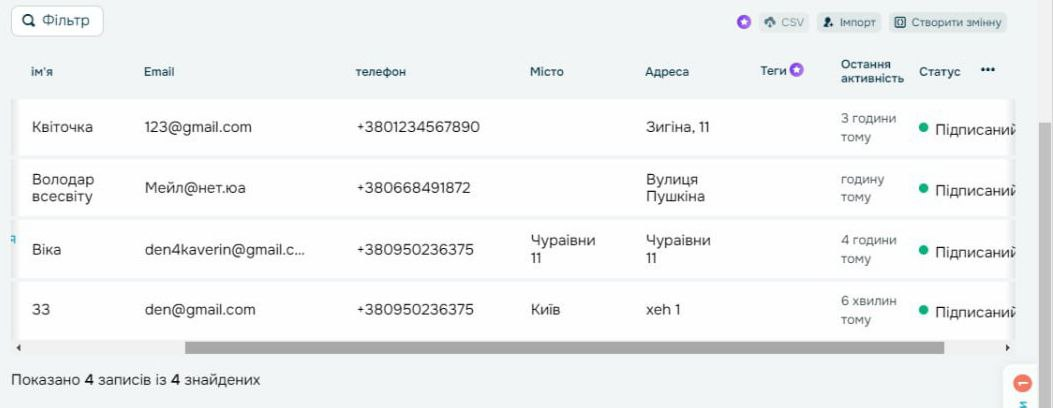


Рисунок Е.7 ­— Збереження даних споживача