МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет Магістерської та аспірантської підготовки
Кафедра інформаційних технологій

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка агрегатору соціальних мереж з використанням Universal Windows Platform»

Виконав студент 2 курсу групи МК-2
спеціальності 122 Комп'ютерні науки
Яковенко Микита Володимирович

Керівник к.геогр.н.,доцент Кузніченко Світлана Дмитрівна

Консультант _____

Рецензент к.т.н.,доцент Гнатовська Ганна Арнольдівна

Одеса 2018

АНОТАЦІЯ

на магістерську роботу

«Розробка агрегатору соціальних мереж з використанням Universal Windows Platform»

Об’єкт дослідження – соціальні мережі і процес їх об’єднання в єдиний соціальний агрегатор.

Предмет дослідження – методи проектування і розробки додатків з використанням технології UWP.

Методи дослідження: UML моделювання, об’єктно-орієнтоване моделювання, концептуальне моделювання.

Метою магістерської роботи є розробка сервісу з адміністрування соціальних мереж, за допомогою якого користувач матиме можливість вести облік своїх пристроїв, а також матиме інструмент для роботи з ними та інформацією, що з них отримується.

Актуальність дослідження обумовлена необхідністю автоматизування процесу роботи з різноманітними соціальними мережами.

Результатом виконання магістерської роботи є розроблений додаток для різноманітних пристроїв під операційну систему Windows 10. Функціональність застосування передбачає можливість створення публікації у різних соціальних мережах, їх керування та процес налаштування під необхідні потреби.

Структура магістерської роботи складається з анотації, вступу, сім розділів, висновків, переліку посилань на 17 найменувань. Повний обсяг роботи становить 68 сторінок і містить 37 рисунків.

Ключові слова: ІНФОРМАЦІЙНА СИСТЕМА, ТЕСТУВАННЯ, СОЦІАЛЬНІ МЕРЕЖІ, АГРЕГАТОР, UWP.

ANNOTATION

for master’s thesis

« Development of a Social Network Aggregator by Means of Universal Windows Platform»

The object of research - social networks and the process of their integration into a single social aggregator.

The subject of research is the methods of designing and developing applications using UWP technology.

Research Methods: UML Modeling, Object-Oriented Modeling, Conceptual Modeling.

The purpose of the master's thesis is to develop a service for administering social networks, through which the user will be able to keep track of their devices, and will have a tool for working with them and the information that comes from them.

The urgency of the study is due to the need to automate the process of working with a variety of social networks.

The result of the master's thesis is an application developed for a variety of devices under the Windows 10 operating system. The functionality of the application includes the ability to create publications in various social networks, their management and customization process to meet the necessary needs.

The structure of the master's thesis consists of annotation, introduction, seven sections, conclusions, list of references to 17 titles. The full work volume is 68 pages and contains 37 pictures.

Keywords: INFORMATION SYSTEM, TESTING, SOCIAL NETWORKS, AGGREGATOR, UWP.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ

БД – база даних
ІС – інформаційна система
СУБД – система управління базами даних
API – Application Programming Interface
CLI – Common Language Intermediate
CLR – Common Language Runtime
CSS – Cascading Style Sheets
DDL – Data Definition Language
DML – Data Manipulation Language
HTML – HyperText Markup Language
IDEF – Integrated Computer-Aided Manufacturing
JSON – JavaScript Object Notation
OAuth – Open Authorization
RDBMS – Relational Database Management System
RDSMS – Relational Data Stream Management System
SQL – Structured Query Language
UWP – Universal Windows Platform
VCS – Version Control System
WPF – Windows Presentation Foundation
XAML – eXtensible Application Markup Language
XML – eXtensible Markup Language

ВСТУП

Соціальні мережі отримують все більше і більше впливу. Щодня в них реєструються сотні тисяч людей. Вартість же найбільших з них обчислюється мільйонами і мільярдами доларів.

Соціальні мережі вже не просто місце спілкування великої кількості людей – це ігри, відео, торгові майданчики і маркетингові інструменти. Не так давно соціальні мережі ще й стали політичною зброєю. Саме завдяки їм спалахнуло кілька революцій в східних країнах.

Ефект соціальних мереж і їх вплив на життя суспільства сьогодні – предмет довгого і пильного вивчення. Ця область користується великим інтересом у підприємств, що просувають таким чином свої товари.

У соціальних мережах люди спілкуються, знаходять собі друзів і коханих, дізнаються новини, об'єднуються в групи за інтересами і роблять покупки. І часто навіть не помічають, як їх вправно заманюють в групу того чи іншого інтернет-магазину або компанії, що здійснює свій бізнес через інтернет.

Зараз групи в соціальних мережах – це ефективний маркетинговий канал для продажу товарів і послуг. На сторінках компаній в соціальних мережах користувачі дізнаються про саму фірму, переймаються до неї довірою, підписуються на новинки, рекламні матеріали. І рано чи пізно стають клієнтами, причому постійними і вірними. Всім цим керує адміністратор соціальних мереж, або адміністратор групи, як його ще називають.

Адміністратор соціальних мереж (адміністратор групи) – це технічний фахівець, який створює, оформляє групу в соціальній мережі (однієї або декількох), і щодня займається її просуванням і підтримкою на основі вказівок, одержуваних від маркетолога або SMM-менеджера компанії-роботодавця.

Агрегатор соціальних мереж – це програмний продукт або сервіс, який збирає інформацію з різних соціальних мереж, блогів та інших ресурсів в одне джерело. Бувають в on-line і off-line виконанні, а також у вигляді додатків для мобільних пристроїв. Ідея агрегатора – спростити використання декількох ресурсів і надати додаткову функціональність як для звичайних користувачів, так і для бізнесу.

Агрегатори подібного спрямування розраховані на звичайного користувача соціальних мереж, який має два і більше акаунтів. Також вони застосовуються в бізнес-процесах. Наприклад для цілей SMM або соціальних медіа, так як надають можливість оперативного розміщення інформації в деякі-

лькох джерелах одночасно, відстеження реакції передплатників новинних стрічок і багато іншого.

В основному, агрегатори використовують REST API соціальних мереж, які, визначають рівень доступу до функцій і ресурсів при використанні даних протоколів. Дана процедура дає можливість використовувати сервіси соцмереж, наприклад, авторизувати користувачів через їх рахунок у будь-якої з соціальних мереж.

Основні функції, які надає агрегатор соціальних мереж, це: збір, обробка, групування, аналіз, а також видача відповідно до запиту користувача інформації як продукту.

Метою магістерської роботи є розробка сервісу з адміністрування соціальних мереж, за допомогою якого користувач матиме можливість вести облік своїх пристроїв, а також матиме інструмент для роботи з ними та інформацією що з них отримується.

Для досягнення поставленої мети в роботі необхідно вирішити наступні завдання:

- описати предметну область та виконати аналіз подібних існуючих систем;
- обґрунтувати вибір програмних засобів і технологій для розробки програмного продукту;
- обґрунтувати які соціальні мережі будуть підтримуватися у додатку;
- виконати проектування;
- створення предметної області та її компонентів;
- виконати процес сертифікації та публікації додатку;
- описати користувацький інтерфейс та його можливості.

1 ХАРАКТЕРИСТИКА ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАВДАННЯ

1.1 Опис предметної області

Маркетинг в соціальних мережах – процес залучення уваги через соціальні платформи. Це комплекс заходів щодо використання соціальних медіа в якості каналів для просування компаній і вирішення інших бізнес-завдань. Marketing в абревіатурі недостатньо точне слово, так як під ним мається на увазі просування, яке входить в комплекс маркетингу. Тобто, більш точну назву – просування в соціальних мережах від англ. Social media promotion (SMP).

Основний упор робиться на створенні контенту, який люди будуть поширювати через соціальні мережі самостійно, вже без участі організатора. Вважається, що повідомлення, що передаються по соціальним мережам, викликають більше довіри у потенційних споживачів товару або послуги. Це пов'язується з рекомендаційної схемою поширення в соціальних медіа за рахунок соціальних зв'язків, що лежать в основі взаємодії.

Просування в соціальних мережах дозволяє точково впливати на цільову аудиторію, вибирати майданчики, де ця аудиторія більшою мірою представлена, і найбільш підходящі способи комунікації з нею, при цьому в найменшій мірі зачіпаючи незацікавлених в цій рекламі людей.

Важливо відзначити, що просування в соціальних мережах застосовується не тільки на товари і послуги. Активно використовують дану технологію засоби масової інформації. Вони створюють свої облікові записи в соціальних мережах, розміщують свій контент і тим самим збирають передплатників (читачів свого продукту).

1.2 Постановка задачі

Мета даної дипломної роботи – розробка сервісу з адміністрування соціальних мереж, за допомогою якого користувач матиме можливість вести облік своїх пристроїв, а також матиме інструмент для роботи з ними та інформацією що з них отримується.

Для реалізації поставленої мети були виділені основні задачі розробки програмного продукту.

1) Зручність. Всі необхідні функції для управління соціальними мережами завжди під рукою.

2) Доступність. Управління соціальними мережами з комп'ютера або гаджета в будь-якій точці світу.

3) Дії з умови. Можливість здавна умов, після яких буде виконуватися необхідну дію.

4) Правила публікації. Гнучкі настройки для публікації контенту в соціальні мережі.

5) Безкоштовні оновлення. Сервіс постійно оновлюється відповідно до побажань користувачів.

6) Легкість налаштувань. На налаштування сервісу знадобиться витратити не більше 5 хвилин часу.

1.3 Огляд і аналіз існуючих систем

Розглянемо підбірку з 10 ефективних інструментів, що дозволяють зручно і ефективно управляти обраними акаунтами в соціальних мережах, збирати дані про взаємодію з користувачами, планувати публікації на тижні вперед і спілкуватися з сотнями передплатників одночасно.

Управління відразу декількома акаунтами в соціальних мережах може стати великою проблемою при дефіциті часу. Facebook, Twitter, Google+, Flickr – кожна з цих платформ вимагає присутності адміністратора і належного рівня активності. В іншому випадку появляється ризик втратити контакт з користувачами і упустити цільову аудиторію.

Everypost – командний пункт управління всіма вашими соціальними мережами одночасно. За допомогою програми користувач може опублікувати пост відразу на декількох ресурсах – Facebook, LinkedIn, Twitter, Pinterest, Tumblr і навіть через email. До публікації можна прикріплювати фотографії, відео і хештеги. Друга важлива функція Everypost – скорочення ваших повідомлень в Twitter до допустимих 140 знаків. Незважаючи на можливість виникнення помилок, Everypost – корисний і, головне, безкоштовний інструмент [1].

Everypost спрощує спостереження за візуальним контентом з різних джерел, налаштовує і розписує повідомлення, а також контролює ваші соціальні сторінки. Everypost спрощує збір мультимедійного контенту з різних джерел, таких як YouTube, Instagram, Flickr і RSS-канали, а також публікується на всіх ваших платформах соціальних мереж.

Everypost дозволяє вам створювати свої повідомлення, коли вам це зручно, і в той же час досягаючи вашої аудиторії в найефективніші моменти часу.

Sprout Social дозволяє не тільки відправляти і публікувати повідомлення в різних соціальних мережах з одного місця, він також збирає дані про надходження нових повідомлень і дає можливість відповісти вашим передплатникам на Facebook і Twitter [2]. Крім того, у вашому розпорядженні будуть візуалізовані дані про активність ваших потенційних клієнтів. Sprout Social доступний в пробної версії, після закінчення її терміну вам буде запропоновано придбати один з трьох пакетів, вартість яких становить 39, 59 та 99 доларів.

Crowdbooster спеціалізується на зборі даних з двох популярних ресурсів – Twitter і Facebook. Crowdbooster надає користувачеві відомості про переглядах публікованих матеріалів, відрізках часу, в які жителі Мережі найчастіше переглядають ваші фотографії і записи, рівень їх відгуку і залученості в життя ваших віртуальних спільнот. Ще одна корисна функція Crowdbooster – оповіщення ваших передплатників в Twitter. Всі перераховані можливості дозволяють скорегувати стратегію роботи з соціальними медіа і домогтися більшої уваги передплатників [3]. На жаль, Crowdbooste не пропонує безкоштовних пакетів послуг, але найдешевший з них обійдеться всього в 9 доларів на місяць – невелике вкладення для охоплення 50 000 аудиторії в соціальних мережах.

SocialFlow – інструмент, який визначає найоптимальніший час для розсилки повідомлень вашим передплатникам і публікації нових матеріалів. При побудові своїх алгоритмів сервіс враховує 3 головні чинники – самі релевантні повідомлення (повідомлення, які з більшою часткою ймовірності досягнуть своєї рекламної мети); правильна аудиторія (користувачі, що знаходяться в Інтернеті в даний момент); ідеальний час (проміжок часу, найбільш підходящий для публікацій і розсилок). SocialFlow – платна програма. На вибір користувачів надається кілька тарифних планів, найдешевший з яких обійдеться в 99 доларів в місяць [4].

Buffer – один з найпопулярніших інструментів для управління публікаціями в ваших спільнотах. Всього в один крок користувач може опублікувати пост в Twitter, LinkedIn і Google+. Головна перевага Buffer – можливість планування публікацій на певний час, що дозволяє вам бути скрізь і завжди з вашими передплатниками [5]. Buffer безкоштовний, але тільки на короткий

час. Найдешевший тарифний план складає 10 доларів в місяць, але за цей термін користувач може запланувати стільки постів, скільки вам завгодно.

2 ВИБІР АРХИТЕКТУРИ, ПРОГРАМНИХ ЗАСОБІВ

2.1 Вибір мови реалізації

C# – це мова програмування, створена Microsoft для використання на платформі .NET. Це мова об'єктно-орієнтованого програмування (ООП) і статично типізована. Ім'я цієї мови походить від C, з якого вона успадковує аналогічний синтаксис. C# був створений Microsoft і стандартизований ISO і ECMA. Він був офіційно випущений в 2002 році. З тих пір на цій мові з'явилося безліч поліпшень, причому останньою версією є C# 7.0.

Розробники програмного забезпечення C# часто називають розробниками .NET, оскільки мова C# майже виключно використовується з .NET Framework. Це досить популярна мова, який зазвичай входить в п'ятірку кращих на різних графіках популярності. Він найбільш часто використовується в розробці корпоративного програмного забезпечення, але також має процвітаючу екосистему з відкритим вихідним кодом.

Мова C# була створена для роботи в CLI (Common Language Infrastructure) і використовує .NET Framework. Вона включає декілька парадигм програмування, таких як об'єктно-орієнтована і функціональний. Мова скомпільована і статично типізована, що означає, що тип всіх змінних перевіряється компілятором при компіляції програми. Однак у версії 4.0 в C# було введено динамічне ключове слово для прив'язки динамічної змінної.

Спочатку побудований для роботи в Windows, він був швидко перенесений на Linux і Mac OS X через проект Mono. Сьогодні C# є відкритим вихідним кодом і працює на крос-платформенном .NET Core.

Специфікація C# визначає мінімальний набір бібліотек типів і класів, на який має розраховувати компілятор. На практиці, C# найчастіше використовується з якоюсь реалізацією CommonLanguageInfrastructure(CLI), яка стандартизована як ECMA-335 CommonLanguageInfrastructure (CLI).

2.2 Вибір технологій побудови графічного інтерфейсу

Windows Presentation Foundation (WPF) – це графічна підсистема Microsoft для візуалізації користувацьких інтерфейсів у додатках на базі Windows. WPF, раніше відомий як «Авалон», з самого початку був випущений як частина .NET Framework 3.0 в 2006 році. WPF використовує DirectX і намагається забезпечити узгоджену модель програмування для створення до-

датків. Він відділяє користувальницький інтерфейс від бізнес-логіки і нагадує аналогічні об'єктно-орієнтовані XML-моделі, наприклад, реалізовані в XUL і SVG. WPF використовує XAML, мову на основі XML, для визначення та зв'язування різних елементів інтерфейсу. WPF додатки можуть бути розгорнуті як автономні настільні програми або розміщені як вбудовані об'єкти на вебсайті. WPF прагне уніфікувати ряд загальних елементів користувацького інтерфейсу, таких як 2D/3D-рендеринг, фіксовані та адаптивні документи, типографіка, векторна графіка, анімація виконання часу та попередньо оброблений носій. Ці елементи можуть бути пов'язані і управлятися на основі різних подій, користувацьких взаємодій та прив’язки даних.

Якщо при створенні традиційних додатків на основі WinForms за отрисовку елементів управління і графіки відповідали такі частини ОС Windows, як User32 і GDI+, то додатки WPF засновані на DirectX. У цьому полягає ключова особливість рендеринга графіки в WPF: використовуючи WPF, значна частина роботи по відображенні графіки, як найпростіших кнопочок, так і складних 3D-моделей, лягати на графічний процесор на відеокарті, що також дозволяє скористатися апаратним прискоренням графіки.

Однією з важливих особливостей є використання мови декларативною розмітки інтерфейсу XAML, заснованого на XML: користувач може створювати насичений графічний інтерфейс, використовуючи або декларативне оголошення інтерфейсу, або код на керованих мовах C# і VB.NET, або поєднувати і те, і інше.

Переваги WPF:

– використання традиційних мов .NET-платформи – C# і VB.NET для створення логіки додатка;

– можливість декларативного визначення графічного інтерфейсу за допомогою спеціальної мови розмітки XAML, заснованому на xml і представляє альтернативу програмному створення графіки та елементів управління, а також можливість комбінувати XAML і C#/VB.NET;

– незалежність від дозволу екрану: оскільки в WPF всі елементи вимірюються в незалежних від пристрою одиницях, додатки на WPF легко масштабуються під різні екрани з різним дозволом;

– нові можливості, яких складно було досягти в WinForms, наприклад, створення тривимірних моделей, прив'язка даних, використання таких елементів, як стилі, шаблони, теми і ін;

– гарна взаємодія з WinForms, завдяки чому, наприклад, в додатках WPF можна використовувати традиційні елементи управління з WinForms;

– багаті можливості по створенню різних додатків: це і мультимедіа, і двомірна і тривимірна графіка, і багатий набір вбудованих елементів управління, а також можливість самим створювати нові елементи, створення анімацій, прив'язка даних, стилі, шаблони, теми і багато іншого;

– апаратне прискорення графіки – незалежно від того, чи працюєте користувач з 2D або 3D, графікою або текстом, все компоненти програми транслюються в об'єкти, зрозумілі Direct3D, і потім візуалізуються за допомогою процесора на відеокарті, що підвищує продуктивність, робить графіком більш плавною;

– створення додатків під безліч ОС сімейства Windows – від Windows XP до Windows 10.

2.3 Архітектура мережевих протоколів REST

2.3.1 Опис архітектури REST

REST або RESTful API призначений для використання існуючих протоколів. Хоча REST можна використовувати практично для будь-якого протоколу, він зазвичай використовує HTTP при використанні для веб-API. Це означає, що розробникам не потрібно встановлювати бібліотеки або додаткове програмне забезпечення, щоб використовувати дизайн REST API. Дизайн REST API був визначений доктором Роєм Філдінгом в його докторської дисертації 2000 року. Він відрізняється неймовірним рівнем гнучкості. Оскільки дані не прив'язані до методів і ресурсів, REST має можливість обробляти кілька типів викликів, повертати різні формати даних і навіть змінювати структурно з правильною реалізацією гіпермедіа.

Ця свобода і гнучкість, властиві дизайну REST API, дозволяють створювати API, який відповідає вашим потребам, а також задовольняє потреби різних клієнтів. На відміну від SOAP, REST не обмежується XML, але замість цього може повертати XML, JSON, YAML або будь-який інший формат в залежності від того, що клієнт запитує. І на відміну від RPC, користувачам не потрібно знати імена процедур або конкретні параметри в певному порядку.

Однак є недоліки в дизайні REST API. Користувач може втратити здатність підтримувати стан в REST, наприклад, в сеансах, і це може бути важче для новіших розробників. Також важливо зрозуміти, що робить REST API RESTful, і чому ці обмеження існують до створення API. Зрештою, якщо ко-

ристувач не розуміє, чому щось створено таким чином, він може перешкодити зусиллям розробника, навіть не усвідомлюючи цього. Антиподом REST є підхід, заснований на виклику віддалених процедур (Remote Procedure Call, RPC). Підхід RPC дозволяє використовувати невелику кількість мережевих ресурсів з великою кількістю методів і складним протоколом. При підході REST кількість методів і складність протоколу суворо обмежені, що призводить до того, що кількість окремих ресурсів має бути великою.

REST – це архітектурний стиль для розподілених гіпертекстових систем.

2.3.2 Архітектурні елементи REST

REST – це архітектурний стиль програмного забезпечення, який визначає набір обмежень, які будуть використовуватися для створення веб-сервісів. Веб-сервіси, відповідні архітектурному стилю REST, називаються веб-службами RESTful, забезпечують взаємодію між комп'ютерними системами в Інтернеті. Веб-служби RESTful дозволяють системам отримувати доступ до текстових уявленням веб-ресурсів та маніпулювати ними, використовуючи єдиний і визначений набір операцій без стану. Інші види веб-сервісів, такі як веб-служби SOAP, виставляють свої власні довільні набори операцій

Компоненти REST системи спілкуються передаючи один одному представлення ресурсу в форматі що обирається з оновлюваного набору стандартних форматів даних. Формат обирається динамічно відповідно до бажань компонента-клієнта і можливостей сервера. Чи представлення має той самий формат що й сам ресурс чи є результатом якогось перетворення – це деталь реалізації яка ховається за інтерфейсом.

Ресурс – це ключовий елемент даних в REST. Ресурсом може бути що завгодно що можна назвати: якийсь документ (наприклад зображення), динамічне значення (наприклад погода у Львові), щось з реального світу (наприклад працівник компанії). Але якщо точніше, то ресурс – це функція приналежності що відображає моменти в часі на множину однотипних сутностей чи значень. Множина може бути порожньою, тобто REST дозволяє посилання на якийсь об'єкт якого ще не існує.

Ресурс може бути динамічним, наприклад ресурс «стаття про REST у вікіпедії» час від часу оновлює свій вміст, а може бути статичним, і після появи ніколи не змінювати свого значення, наприклад «перша версія статті про REST вікіпедії». У REST такі два ресурси вважаються різними, хоча в пев-

ний момент часу вони можуть вказувати на одну й ту ж сутність. Єдине що важливо – семантика відображення імені ресурсу на його вміст.

Для того, щоб посилатись на ресурси, використовуються ідентифікатори ресурсів. Компонент, який надав ресурсу ідентифікатор і дозволяє звертатись до нього за цим ідентифікатором, відповідає за збереження функції приналежності незмінною. Якість ідентифікатора залежить від якості компонента, який цей ідентифікатор надає, тому деякі ідентифікатори стають «мертвими посиланнями», коли інформацію переміщують або знищують.

Представлення (англ. representation) – це послідовність байтів та метадані представлення, для того щоб описати ці байти. Часто, представлення називають документом, файлом, повідомленням HTTP тощо

Метадані також можуть бути не лише в представлення ресурсу, а й в самого ресурсу. Прикладами метаданих ресурсу є посилання на джерело, `<link rel="alternates" ... />`, заголовок HTTP vary.

Контрольні дані в представленні описують ціль повідомлення між компонентами, наприклад прохання про дію (створити, змінити видалити ресурс), або значення відповіді (наприклад поточний стан ресурсу, чи значення якогось іншого ресурсу, наприклад опис помилки).

Також, контрольні дані включені в запити чи відповіді можуть керувати поведінкою кеша. Прикладом таких контрольних даних можуть бути заголовки HTTP if-modified-since та cache-control.

Формат даних представлення називають типом медіа (англ. media type). Одні типи медіа краще підходять для автоматичної обробки, інші – для того щоб бути показаними користувачу. Композитні типи медіа можуть використовуватись для того аби поєднати кілька видів представлення в одному.

Від формату даних дуже залежить латентність застосунку яку сприймає користувач. Наприклад браузер може почати показувати сторінку ще до того як завантажиться ввесь HTML, це збільшує видиму швидкість роботи.

Конектори надають інтерфейс для комунікації компонентів, приховуючи реалізації ресурсів та механізм комунікації.

Конектори подібні на віддалений виклик процедур, але з певними нюансами щодо передачі параметрів та результату виклику. Параметри складаються з ідентифікатора ресурсу, контрольних даних та необов'язково, представлення. Результат – з контрольних даних відповіді і представлення. Можна абстрагуватись і вважати такий виклик синхронним, але насправді передача даних відбувається потоково, тому обробку даних можна починати ще до того як отримані всі дані, таким чином зменшуючи латентність.

Двома найважливішими типами конекторів є клієнт і сервер. Відмінністю між ними є те що клієнт ініціює запит, в той час як сервер очікує запитів і відповідає на них даючи доступ до своїх сервісів. Компонент може містити одночасно як серверні так і клієнтські конектори.

Додатковим типом конектора є кеш. Кеш може бути як клієнтським, для уникнення зайвих запитів, так і серверним – для уникнення зайвого обчислення відповіді на запит. Тому що інтерфейс однорідний, кеш легко може дізнатись чи запит можна кешувати. За замовчуванням, відповідь на запит отримання ресурсу можна кешувати, а запити зміни ресурсу – ні. Проте ці замовчування можна перевантажити контрольними даними.

Резолвер (англ. resolver) – це ще один тип конектора, який перетворює ідентифікатори ресурсів в інформацію про мережеві адреси необхідну компонентам щоб отримати цей ресурс. Наприклад в URI міститься доменне ім'я, і для доступу до відповідного домена, потрібно дізнатись в DNS-сервера адресу. Тому в цьому випадку система DNS грає роль резолвера. Використання одного чи кількох резолверів може збільшити життєздатність ідентифікатора ресурсу, через те що він не вказує напряму на фізичне розташування ресурсу яке може змінитись.

Останньою формою конектора є тунель, який просто проводить запити через межу системи, наприклад через фаєрвол. Причиною через яку тунелі включені до архітектури REST а не закриті абстракцією мережі є те, що певні компоненти можуть перетворюватись в тунелі по запиту. Наприклад HTTP тунель активується при отриманні запиту з методом CONNECT.

2.4 Середовище проектування базами даних SQLite

2.4.1 Зберігання даних у SQLite

SQLite – це система управління реляційними базами даних, які містяться у бібліотеці програмування C. У відмінності від багатьох інших систем управління базами даних, SQLite не є механізмом баз даних клієнт-сервера. Скоріше, він вбудований в кінцеву програму [6].

SQLite сумісним з ACID і реалізує більшу частину стандарту SQL, використовуючи динамічна і слабо типізований синтез SQL, який не гарантує цілісності домену.

SQLite є популярним вибором в якості вбудованого програмного забезпечення баз даних для локальної/клієнтської зберігання в прикладному про-

грамному забезпеченні, такому як веб-браузери. Це, можливо, найбільш широко розгорнуте механізми баз даних, оскільки він використовується сьогодні багатьма широко розповсюдженими браузерами, операційними системами та вбудованими системами (такими, як мобільні телефони) та іншими. SQLite має прив'язки до багатьох мов програмування. Кілька процесів або потоків можуть одночасно без жодних проблем читати дані з однієї бази. Запис в базу можна здійснити тільки в тому випадку, коли жодних інших запитів у цей час не обслуговується; інакше спроба запису закінчується невдачею, і в програму повертається код помилки. Іншим варіантом розвитку подій є автоматичне повторення спроб запису протягом заданого інтервалу часу.

У комплекті постачання йде також функціональна клієнтська частина у вигляді виконуваного файлу sqlite3, за допомогою якого демонструється реалізація функцій основної бібліотеки. Клієнтська частина працює з командного рядка, і дозволяє звертатися до файлу БД на основі типових функцій ОС.

Завдяки архітектурі рушія можливо використовувати SQlite як на вбудовуваних (embedded) системах, так і на виділених машинах з гігабайтними масивами даних.

Особливості SQLite:

- транзакції атомарні, послідовні, ізольовані, і міцні (ACID) навіть після збоїв системи і збоїв живлення;
- встановлення без конфіграції – не потребує ані установки, ані адміністрування;
- реалізує значну частину стандарту SQL92;
- база даних зберігається в одному крос-платформовому файлі на диску;
- підтримка терабайтних розмірів баз даних і гігабайтного розміру рядків і BLOBів;
- малий розмір коду: менше ніж 350KB повністю налаштований, і менш 200KB з опущеними додатковими функціями;
- швидший за популярні рушії клієнт-серверних баз даних для найпоширеніших операцій;
- простий, легкий у використанні API;
- написана в ANSI C, включена прив'язка до TCL; доступні також прив'язки для десятків інших мов;
- добре прокоментований сирцевий код зі 100% тестовий покриттям гілок;

− доступний як єдиний файл сирцевого коду на ANSI C, який можна легко вставити в інший проект;

− автономність: немає зовнішніх залежностей;

− крос-платформовість: з коробки підтримується Unix (Linux і Mac OS X), OS/2, Windows (Win32 і WinCE). Легко переноситься на інші системи;

− сирці перебувають в суспільному надбанні;

− оставляється з автономним клієнтом інтерфейсу командного рядка, який може бути використаний для управління базами даних SQLite.

2.4.2 SQL – мова реляційної бази даних

Мова реляційної бази даних в системі SQL Server називається Transact-SQL. Це різновид самої значимої на сьогоднішній день мови бази даних – мови SQL (мова структурованих запитів) – це мова, специфічний для домену, який використовується в програмуванні і призначений для управління даними, що зберігаються в системі керування базами даних (RDBMS), або для потокової обробки в системі управління реляційними потоками даних (RDSMS). Це особливо корисно при обробці структурованих даних, де існують відносини між різними об'єктами/змінними даних. SQL пропонує дві основні переваги в порівнянні з більш старими API-інтерфейсами для читання/запису, такими як ISAM або VSAM: по-перше, він представив концепцію доступу до безлічі записів за допомогою однієї команди; і, по-друге, це усуває необхідність вказувати, як досягти записи, наприклад. з індексом або без нього [7].

Заснований на реляційній алгебри та реляційному численні кортежів, SQL складається з багатьох типів операторів, які можуть бути неформально класифіковані як підмови, зазвичай: мова запитів даних, мова визначення даних, мова керування даними і мова маніпулювання даними. Обсяг SQL включає в себе запит даних, маніпулювання даними (вставка, оновлення та видалення), визначення даних (створення і модифікація схеми) і управління доступом до даних. Хоча SQL часто описується як, і в значній мірі це декларативний мову (4GL), він також включає процедурні елементи [8].

Мова SQL містить два под'язика: мова опису даних DDL (Data Definition Language) і мову обробки даних DML (Data Manipulation Language). Інструкції мови DDL також застосовуються для опису схем таблиць баз даних. Мова DDL містить три загальні інструкції SQL: CREATE, ALTER і DROP. Ці інструкції використовуються для створення, зміни та ви-

далення, відповідно, об'єктів баз даних, таких як бази даних, таблиці, стовпці та індекси.

На відміну від мови DDL, мова DML охоплює всі операції з маніпулювання даними. Для маніпулювання базами даних завжди застосовуються чотири загальні операції: вилучення, вставка, видалення і модифікування даних (SELECT, INSERT, DELETE, UPDATE).

2.5 Система управління версіями

2.5.1 Описі технології системи управління версіями

Система управління версіями (від англ. Version Control System, VCS або Revision Control System) – програмне забезпечення для полегшення роботи зі змінною інформацією. Система управління версіями дозволяє зберігати кілька версій одного і того ж документа, при необхідності повертатися до попередніх версій, визначати, хто і коли зробив ту чи іншу зміну, і багато іншого.

Такі системи найбільш широко використовуються при розробці програмного забезпечення для зберігання вихідних кодів програми, що розробляється. Однак вони можуть з успіхом застосовуватися і в інших областях, в яких ведеться робота з великою кількістю безперервно змінюються електронних документів. Зокрема, системи управління версіями застосовуються в САПР, зазвичай в складі систем управління даними про виріб (PDM). Управління версіями використовується в інструментах конфігураційного управління (Software Configuration Management Tools).

Програмне забезпечення Вікіпедії веде історію змін для всіх її статей, використовуючи методи, аналогічні тим, які застосовуються в системах управління версіями.

2.5.2 Загальні відомості о системах управління версіями

Управління версіями, також відоме як контроль версій або управління вихідними кодами – це управління змінами в документах, комп'ютерних програмах, веб-сайтах або збору інформації.

Ці зміни часто позначаються як номери або буквенний код, що називається номером ревізії. Кожна ревізія пов'язана з метаданними, такими як мет-

ка часу, автор (людина, внесла зміни) тощо. Перегляд можна порівняти, відновити або об'єднати [9].

Традиційні системи управління версіями використовують централізовану модель, коли є єдине сховище документів, кероване спеціальним сервером, який і виконує велику частину функцій з управління версіями. Користувач, який працює з документами, повинен спочатку отримати потрібну йому версію документа зі сховища; зазвичай створюється локальна копія документа, так звана «робоча копія». Може бути отримана остання версія або будь-яка з попередніх, яка може бути обрана за номером версії або датою створення, іноді і за іншими ознаками. Після того, як в документ внесені потрібні зміни, нова версія поміщається в сховище. На відміну від простого збереження файлу, попередня версія не стирається, а також залишається в сховище і може бути звідти отримана в будь-який час. Сервер може використовувати т. н. дельта-компресію – такий спосіб зберігання документів, при якому зберігаються тільки зміни між послідовними версіями, що дозволяє зменшити обсяг збережених даних.

Оскільки зазвичай найбільш затребуваною є остання версія файлу, система може при збереженні нової версії зберігати її цілком, замінюючи в сховище останню раніше збережену версію на різницю між цією і останньою версією. Деякі системи (наприклад, ClearCase) підтримують збереження версій обох видів: більшість версій зберігається у вигляді дельт, але періодично (по спеціальній команді адміністратора) виконується збереження версій всіх файлів в повному вигляді; такий підхід забезпечує максимально повне відновлення історії в разі пошкодження сховища.

Іноді створення нової версії виконується непомітно для користувача (прозоро), або прикладною програмою, що має вбудовану підтримку такої функції, або за рахунок використання спеціальної файлової системи. У цьому випадку користувач просто працює з файлом, як зазвичай, і при збереженні файлу автоматично створюється нова версія.

Часто буває, що над одним проектом одночасно працюють кілька людей. Якщо дві людини змінюють один і той же файл, то один з них може випадково скасувати зміни, внесені іншим. Системи управління версіями відстежують такі конфлікти і пропонують засоби їх вирішення. Більшість систем може автоматично об'єднати (злити) зміни, зроблені різними розробниками. Однак таке автоматичне об'єднання змін, зазвичай, можливо тільки для текстових файлів і за умови, що змінювалися різні (непересічні) частини цього файлу. Таке обмеження пов'язане з тим, що більшість систем управління вер-

сіями орієнтовані на підтримку процесу розробки програмного забезпечення, а вихідні коди програм зберігаються в текстових файлах. Якщо автоматичне об'єднання виконати не вдалося, система може запропонувати вирішити проблему вручну.

Часто виконати злиття неможливо ні в автоматичному, ні в ручному режимі, наприклад, якщо формат файла невідомий або занадто складний. Деякі системи управління версіями дають можливість заблокувати файл в сховище. Блокування не дозволяє іншим користувачам отримати робочу копію або перешкоджає зміні робочої копії файлу (наприклад, засобами файлової системи) і забезпечує, таким чином, винятковий доступ тільки тому користувачеві, який працює з документом.

Багато систем управління версіями надають ряд інших можливостей:

− дозволяють створювати різні варіанти одного документа, т.з. гілки, із загальною історією змін до точки розгалуження і з різними – після неї;

− дають можливість дізнатися, хто і коли додав або змінив конкретний набір рядків у файлі;

− ведуть журнал змін, в який користувачі можуть записувати пояснення про те, що і чому вони змінили в цій версії;

− контролюють права доступу користувачів, дозволяючи або забороняючи читання або зміна даних, в залежності від того, хто запитує цю дію.

2.6 Опис системи керування версіями GIT

Система управління версіями (VCS) являє собою набір інструментів, які відстежують історію набору файлів. Це означає, що ви можете повідомити VCS (Git), щоб зберегти стан ваших файлів в будь-який момент. Потім користувач може продовжувати редагувати файли і зберігати цей стан. Збереження стану схоже на створення резервної копії робочого каталогу. При використанні Git користувач посилається на це збереження стану як на фіксацію.

Коли користувач робить фіксацію в Git, він додає повідомлення про фіксацію, яке на високому рівні пояснює, які зміни він додає в цю фіксацію. Git може показати історію всіх коммітов і їх повідомлень про скоєння. Це дає корисну історію того, що було зроблено, і може реально допомогти визначити, коли помилка проникла в систему.

Крім того, що він показуєте журнал змін, Git також дозволяє порівнювати файли між різними коммітами. Як було вже відмічено раніше, Git також

дозволить повернути будь-який файл (або всі файли) до більш раннього фіксації з невеликими зусиллями.

У міру того як більші групи почали працювати (і мережі стали більш поширеними), VCS змінилися для зберігання сховища на центральному сервері, який був наданий багатьма розробниками. Хоча це і допомогло вирішити багато проблем, воно також створило нові, такі як блокування файлів.

Наслідуючи приклад кількох інших продуктів, Git зламався з цією моделлю. Git не має центрального сервера, який має остаточну версію сховища. Всі користувачі мають повну копію сховища. Це означає, що змусити всіх розробників повернутися на одну і ту ж сторінку іноді буває складно, але це також означає, що розробники можуть працювати в автономному режимі більшу частину часу, тільки підключаючись до інших репозиторіїв, коли їм потрібно поділитися своєю роботою.

2.7 Вибір платформи реалізації

UWP – це платформа розробки додатків, створена Microsoft. Додаток UWP може працювати на декількох пристроях, включаючи ПК з ОС Windows, планшети і смартфони [9]. Деякі додатки UWP можуть працювати на інших типах пристроїв Microsoft, включаючи пристрої Xbox, HoloLens і IoT.

UWP надає загальну платформу для розробників для створення додатків для декількох типів обладнання. Універсальний API Windows Platform включає в себе широкий спектр бібліотек, функцій і елементів призначеного для користувача інтерфейсу, які розробники можуть інтегрувати в свої додатки. Включаючи кілька типів DeviceFamily в додатку UWP, розробник може налаштувати інтерфейс програми для декількох типів пристроїв.

Microsoft Visual Studio 2015 і більш пізні версії підтримують UWP API і надають розробникам гнучку набір інструментів, написаний середу розробки. Хоча API реалізований на C++, додатки UWP можуть бути написані на C++, C#, Visual Basic або навіть на JavaScript. Microsoft Visual Studio IDE скомпілює код як додаток UWP, якщо Windows.Universal буде налаштоване як сімейство цільових пристроїв.Додатки, які здатні реалізувати дану платформу, створюються з використанням Visual Studio 2015. Старі Metro-додатки для Windows 8.1 або Windows Phone 8.1 потребують зміни коду, щоб підтримувати UWP.

UWP була лише доповненням до Windows Runtime. Універсальні програми Windows створені з використанням технології UWP, які не потребують позначенні прив'язки до якої ОС вони призначені; крім того, вони підтримують як ПК, так і смартфони, планшети або Xbox One, використовуючи мости UWP. Дане розширення дозволяє автоматично підтримувати всі можливі платформи. Універсальний додаток може бути запущено на будь-якому мобільному телефоні або планшеті. Воно ж, запущене на смартфоні, може вести себе як ніби запущено на ПК, якщо підключено до останнього за допомогою док-станції.

3 ПРОЕКТУВАННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

3.1 Розробка UML діаграми класів

UML – уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого програмування. Є невід'ємною частиною уніфікованого процесу розробки програмного забезпечення. UML є мовою широкого профілю, це відкритий стандарт, що використовує графічні позначення для створення абстрактної моделі системи, яка називається UML-моделлю. UML був створений для визначення, візуалізації, проектування й документування в основному програмних систем. UML не є мовою програмування, але в засобах виконання UML-моделей як інтерпретованого коду можлива кодогенерація [10].

UML може бути застосовано на всіх етапах життєвого циклу аналізу бізнес-систем і розробки прикладних програм. Різні види діаграм які підтримуються UML, і найбагатший набір можливостей представлення певних аспектів системи робить UML універсальним засобом опису як програмних, так і ділових систем.

Діаграми дають можливість представити систему (як ділову, так і програмну) у такому вигляді, щоб її можна було легко перевести в програмний код.

Основною причиною використання мови UML є спілкування розробників між собою.

Крім того, UML спеціально створювалася для оптимізації процесу розробки програмних систем, що дозволяє збільшити ефективність їх реалізації у кілька разів і помітно поліпшити якість кінцевого продукту.

UML чудово зарекомендувала себе в багатьох успішних програмних проектах. Засоби автоматичної генерації кодів дозволяють перетворювати моделі мовою UML у вихідний код об’єктно-орієнтованих мов програмування, що ще більш прискорює процес розробки.

Практично усі CASE-засоби (програми автоматизації процесу аналізу і проектування) мають підтримку UML. Моделі розроблені в UML, дозволяють значно спростити процес кодування і направити зусилля програмістів безпосередньо на реалізацію системи. Діаграми підвищують підтримку проекту і полегшують розробку документації.

UML необхідний:

– керівникам проектів, які керують розподілом завдань і контролем за проектом;

– проектувальникам інформаційних систем які розробляють технічні завдання для програмістів;

– бізнес-аналітикам, які досліджують реальну систему і здійснюють інжиніринг і реінжиніринг бізнесу компанії;

– програмістам які реалізують модулі інформаційної системи.

При модифікації системи об'єктний підхід дозволяє легко включати в систему нові об'єкти і виключати застарілі без істотної зміни її життєздатності. Використання побудованої моделі при модифікаціях системи дає можливість усунути небажані наслідки змін, оскільки вони не ламають структури системи, а тільки змінюють поведінку об'єктів.

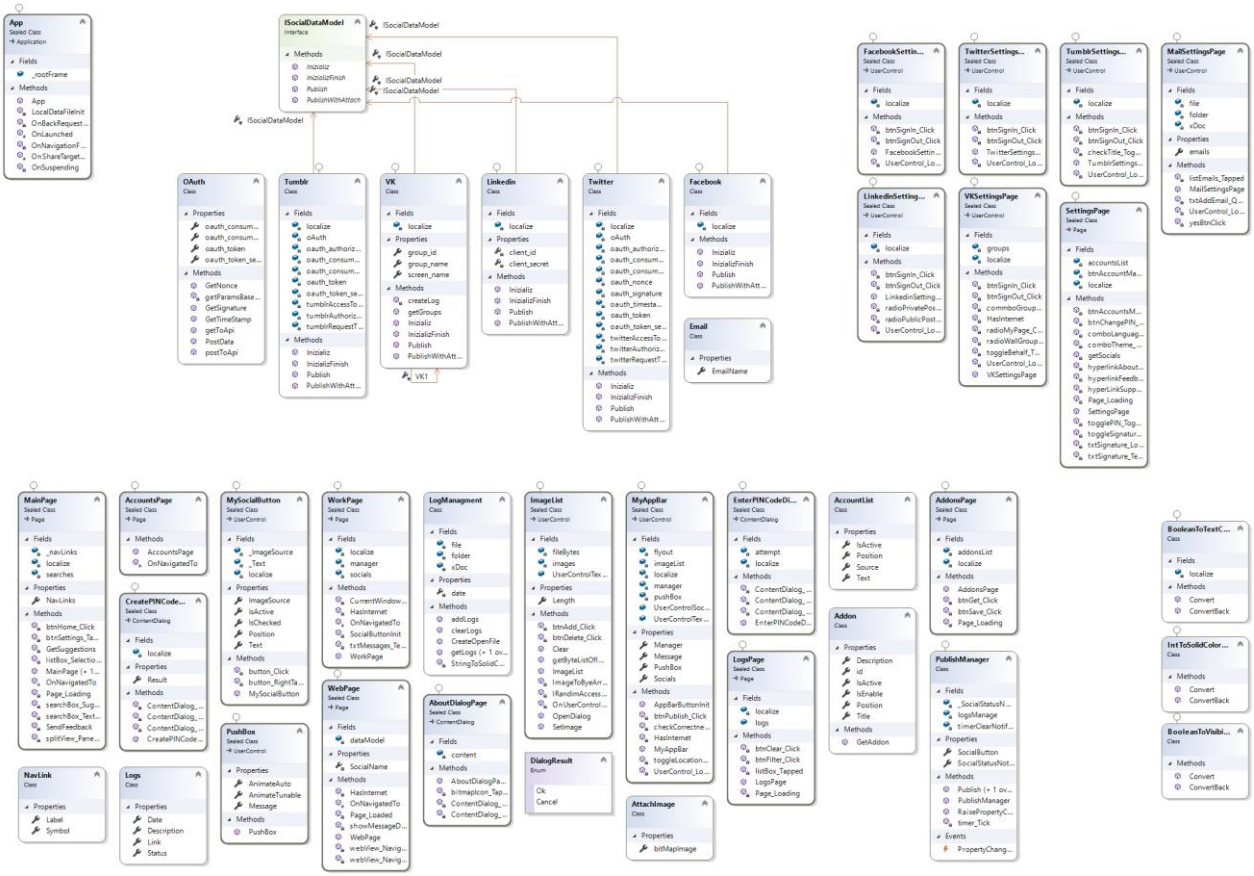

Рисунок 3.1 – UML діаграма класів

Створюваний додаток має наступну UML діаграму класів (рис. 3.1). На ній ми можемо бачити усі створені класи. Що відповідають за побудову призначеного для користувача інтерфейсу, класи-помічники для створення власних інструментів, також класи адаптери для роботи із колекціями і багато

інших. Також можна виділити базовий інтерфейс ISocialDataModel в ньому оголошені 4 методу, такі як: Inizializ, InizializFinish, Puplish, PublishFinish. Які в свою чергу реалізуються головними класами для роботи соціальних мереж. Після чого ми маємо єдиний «Контракт» для роботи з усіма модулями створюваного програмного продукту.

3.2 Опис файлової системи та роботи з файлами

Windows 10 надає новий API для роботи з файлами. Основу цього API становить простір імен Windows.Storage, в якому і зосереджений весь функціонал для роботи з файлами. Одними з найважливіших класів в цьому API є StorageFile і StorageFolder. Клас StorageFile повертає інформацію про певний файлі і його вмісті, а також надає способи для маніпуляції з цим файлом. А клас StorageFolder призначений для управління папками користувача.

При роботі з файлами і папками треба враховувати, що нам доступні певні види сховищ файлів і папок, з якими ми можемо взаємодіяти. Файл в Windows 10 може бути збережений в самих різних місцях: на локальному комп'ютері, на знімному жорсткому диску або флешці, в хмарі. Всі типи папок незалежно від місцезнаходження представлені класом StorageFolder (рис. 3.2).

XML – розширювана мова розмітки. Рекомендований Консорціумом Всесвітньої павутини (W3C). Специфікація XML описує XML-документи і частково описує поведінку XML-процесорів (програм, які читають XML-документи і забезпечують доступ до їх вмісту). XML розроблявся як мова з простим формальним синтаксисом, зручний для створення і обробки документів програмами і одночасно зручний для читання і створення документів людиною, з підкресленням націленості на використання в Інтернеті. Мова називається розширюваним, оскільки ним не фіксується розмітку, яка використовується в документах: розробник вільний створити розмітку відповідно до потреб конкретної області, будучи обмеженим лише синтаксичними правилами мови. Розширення XML – це конкретна граматика, створена на базі XML і представлена словником тегів і їх атрибутів, а також набором правил, що визначають які атрибути і елементи можуть входити до складу інших елементів. Поєднання простого формального синтаксису, зручності для людини, розширюваності, а також базування на кодуваннях Юнікод для подання змісту документів привело до широкого використання як власне XML, так

і безлічі похідних спеціалізованих мов на базі XML в найрізноманітніших програмних засобах.

```
public async void addLogs(Logs log)
{
    xDoc.Element("logs").Add(new XElement("log",
        new XElement("status", log.Status),
        new XElement("date", date),
        new XElement("description", log.Description),
        new XElement("link", log.Link)));

    await FileIO.WriteTextAsync(file, xDoc.ToString());
}
```

Рисунок 3.2 – Метод для вставки даних до .xml файлу

Щоб почати роботу з наявним xml-файлом, треба спочатку завантажити його за допомогою статичного методу XDocument.Load (), в який передається шлях до файлу.

Оскільки xml зберігає ієрархічно вибудувані елементи, то і для доступу до елементів треба йти починаючи з вищого рівня в цій ієрархії і далі вниз. Так, для отримання елементів log і доступу до них треба спочатку звернутися до кореневого елементу, а через нього вже до елементів log: xdoc.Element ("logs"). Elements ("log") Метод Element ("імя_елемента") повертає перший знайдений елемент з таким ім'ям (рис. 3.3).

```
public List<Logs> getLogs()
{
    List<Logs> logs = xDoc.Root.Elements("log").Select(a =>
    {
        return new Logs()
        {
            Status = Convert.ToInt32(a.Element("status").Value),
            Date = a.Element("date").Value,
            Description = a.Element("description").Value,
            Link = a.Element("link").Value
        };
    }).OrderByDescending(b=>b.Date).ToList();

    return logs;
}
```

Рисунок 3.3 – Метод для отримання даних з .xml файлу

Метод Elements ("імя_елемента") повертає колекцію однойменних елементів. В даному випадку ми отримуємо колекцію елементів log і тому можемо перебрати її в циклі. Спускаючись далі по ієрархії вниз, ми можемо отримати атрибути або вкладені елементи, наприклад,

XElement linkElement = logElement.Element ("Link")

Значення простих елементів, які містять один текст, можна отримати за допомогою властивості Value: string link = logElement.Element ("Link").Value

4 НАЛАШТУВАННЯ REST API ТА ВИКОРИСТАННЯ ЙОГО З РІЗНИМИ СОЦІАЛЬНИМИ МЕРЕЖАМИ

REST – архітектурний стиль взаємодії компонентів розподіленого додатка в мережі. REST є узгоджений набір обмежень, що враховуються при проектуванні розподіленої гіпермедіа-системи. У певних випадках (інтернет-магазини, пошукові системи, інші системи, засновані на даних) це призводить до підвищення продуктивності і спрощення архітектури. У широкому сенсі компоненти в REST взаємодіють на зразок взаємодії клієнтів і серверів у Всесвітній павутині. REST є альтернативою RPC. В мережі Інтернет виклик віддаленої процедури може являти собою звичайний HTTP-запит (зазвичай «GET» або «POST»; такий запит називають «REST-запит»), а необхідні дані передаються в якості параметрів запиту [11].

Для веб-служб, побудованих з урахуванням REST (тобто не порушують накладаються їм обмежень), застосовують термін «RESTful». На відміну від веб-сервісів (веб-служб) на основі SOAP, не існує «офіційного» стандарту для RESTful веб-API. Справа в тому, що REST є архітектурним стилем, в той час як SOAP є протоколом. Незважаючи на те, що REST не є стандартом сам по собі, більшість RESTful-реалізацій використовують стандарти, такі як HTTP, URL, JSON і XML.

OAuth – це відкритий стандарт авторизації, який дозволяє користувачам відкривати доступ до своїх приватних даних (фотографії, відео, списки контактів), що зберігаються на одному сайті, іншому сайту, без необхідності вводу імені користувача та паролю.

OAuth дозволяє користувачам роздавати сайтам маркери доступу, до даних що розміщуються на сайтах-сервісах. Кожен маркер доступу надає доступ конкретному сайту (наприклад, сайту редагування відео) до конкретних ресурсів (наприклад, тільки відео від конкретного альбому) та на визначений термін (наприклад, на наступні 2 години). Це дозволяє користувачам надавати доступ третім сайтам до їх інформації, що зберігається на інших сайтах – постачальниках послуг, не передаючи повною мірою самих даних та без застосування імені/паролю.

Якщо взяти, наприклад, API Твіттера, то інтерфейс цього сервісу може видати вам інформацію про «твіти» користувача, його читачів і про тих, хто його читає, і так далі. Це лише мала частина можливостей, які будь-хто може втілити, використовуючи API стороннього сервісу або створюючи свій власний.

На основі API будуються такі речі, як карти 2GIS, всілякі мобільні і десктопні клієнти для Twitter та Vkontakte. Всі їх функції стали можливими саме завдяки тому, що відповідні сервіси мають якісні і детально задокументовані API.

Стандартний запит даних від стороннього API виглядає приблизно так: https://api.github.com/users/Freika

На випадок, якщо хто-то ще не знає, варто зауважити, що curl не має ніякого відношення до API і використовується в операційних системах для відправки та отримання даних через термінал.

Подібним чином можна надсилати запит на будь-якій мові, в тому числі і на Ruby. Відповіддю на запит буде приблизно така інформація, яка містить: логін, аватар, посилання на профіль на сайті і в API, статус користувача, кількість публічних репозиторіїв та іншу корисну інформацію (рис. 4.1).


```
{
  "login": "Freika",
  "id": 3738638,
  "avatar_url": "https://avatars.githubusercontent.com/u/3738638?v=3",
  "gravatar_id": "",
  "url": "https://api.github.com/users/Freika",
  "html_url": "https://github.com/Freika",
  "followers_url": "https://api.github.com/users/Freika/followers",
  "following_url": "https://api.github.com/users/Freika/following{/other_user}",
  "gists_url": "https://api.github.com/users/Freika/gists{/gist_id}",
  "starred_url": "https://api.github.com/users/Freika/starred{/owner}{/repo}",
  "subscriptions_url": "https://api.github.com/users/Freika/subscriptions",
  "organizations_url": "https://api.github.com/users/Freika/orgs",
  "repos_url": "https://api.github.com/users/Freika/repos",
  "events_url": "https://api.github.com/users/Freika/events{/privacy}",
  "received_events_url": "https://api.github.com/users/Freika/received_events",
  "type": "User",
  "site_admin": false,
  "name": "Evgeniy",
  "company": "",
  "blog": "http://frey.su/",
  "location": "Barnaul",
  "email": "",
  "hireable": true,
  "bio": null,
  "public_repos": 39,
  "public_gists": 13,
  "followers": 15,
  "following": 21,
  "created_at": "2013-03-01T13:48:52Z",
  "updated_at": "2014-12-15T13:55:03Z"
}
```


Рисунок 4.1 – Відповідь від API сервісу

5 РОЗРОБКА СИСТЕМИ ТА ЇЇ КОМПОНЕНТІВ

5.1 Локалізація рядків в маніфесті пакета програми та інтерфейсу користувача

Щоб додаток підтримувало різні мови інтерфейсу, а в коді або розмітці XAML або маніфесті пакета додатка є рядкові літерали, потрібно перемістити ці рядки з коду або розмітки в файл ресурсів (.resw). Потім можна створити перекладену копію цього файлу ресурсів для кожної мови, що підтримується додатком [12].

Жорстко закодовані рядкові літерали можуть з'являтися в імперативний коді або в розмітці XAML, наприклад у вигляді властивості Text об'єкта TextBlock. Вони також можуть з'являтися в маніфесті пакета додатка (фото Package.appxmanifest), наприклад в якості значення для Імен на вкладці "Додаток" в конструкторі маніфесту Visual Studio. Необхідно перемістити ці рядки в файл ресурсів (.resw) і замінити жорстко закодовані рядкові літерали в своєму додатку і в маніфесті посиланнями на ідентифікатори ресурсів (рис. 5.1).

На відміну від ресурсів зображень, де в файлі ресурсів міститься тільки один ресурс зображення, в файлі строкових ресурсів міститься багато строкових ресурсів. Файл строкових ресурсів являє собою файл ресурсів (.resw), і зазвичай такий файл ресурсів створюється в папці \Strings проекту.

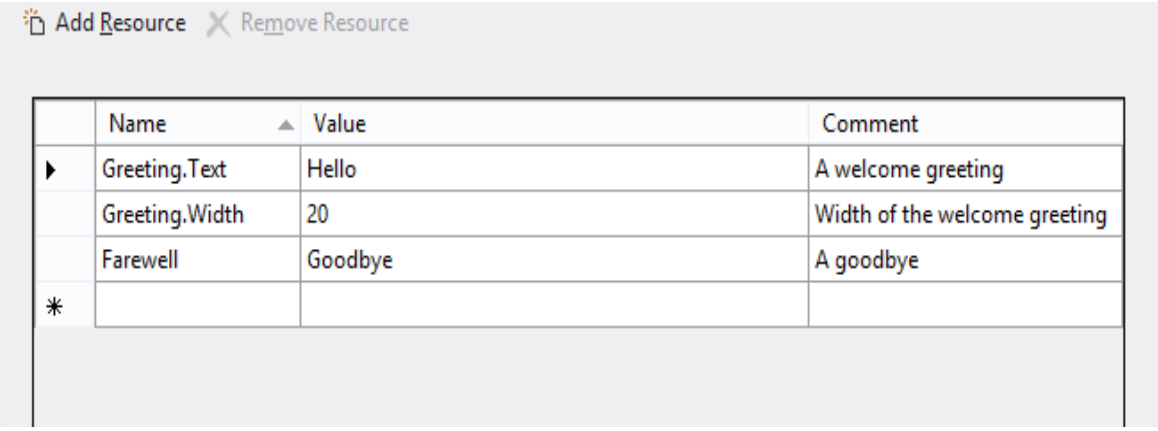

Рисунок 5.1 – Приклад файлу локалізації

Процес створення файл ресурсів (.resw) і заповнення в нього строкових ресурсів.

1) необхідно встановити мову за замовчуванням для програми:

а) відкрити рішення в Visual Studio, далі файл Package.appxmanifest;
б) на вкладці «Додатки» переконається, що мова за замовчуванням заданий відповідним чином (наприклад, en або en-US);

2) далі необхідно створити файл ресурсів (.resw) для мови за замовчуванням:
   а) у вузлі проекту необхідно створити нову папку і назвати її Strings;
   б) у розділі Strings створити нову вкладену папку і називаємо її en-US;
   в) у розділі en-US необхідно створити новий файл ресурсів (.resw), та переконатися що він називається Resources.resw;
3) відкрити Resources.resw і додати ці строкові ресурси;
4) strings/en-US/Resources.resw.

5.2 Конвертери значень у UWP

Конвертери значень (value converter) дозволяють перетворити значення, що приходять від джерела прив'язки, до того типу, який зрозумілий приймача прив'язки. Не завжди два пов'язують прив'язкою властивості можуть мати сумісні типи. І в цьому випадку якраз і потрібен конвертер значень (рис. 5.2).

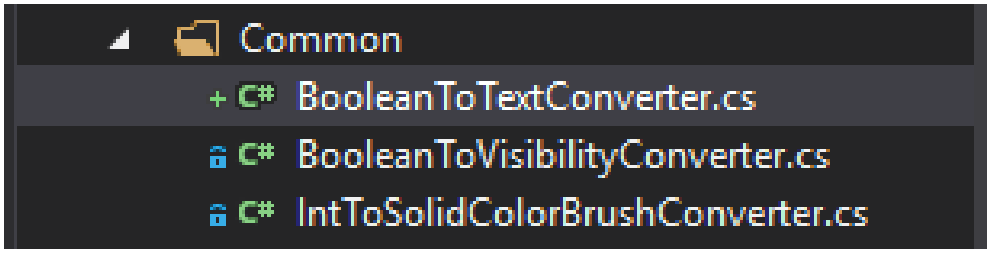

Рисунок 5.2 – Демонстрація існуючих конверторів

Конвертер значень повинен реалізувати інтерфейс Windows.UI.Xaml.Data.IValueConverter. У ньому визначається два методи: Convert() (перетворює значення від джерела прив'язки в тип, зрозумілий для приймача прив'язки) і ConvertBack() (виконує протилежну операцію). Обидва методи приймають такі параметри:

- object value: значення, яке треба перетворити;
- type targetType: тип, до якого треба перетворити значення value;
- object parameter: допоміжний параметр прив'язки;
- string language: мовний код поточної культури, наприклад, "en-US".

У нашому конвертері значень метод Convert повертає строкове представлення дати у форматі "dd.MM.yyyy". Тобто ми очікуємо, що в якості параметра value буде передаватися об'єкт DateTimeOffset.

У додатку булу розроблено декілька конверторів один з яких відповідає за перетворення значень логічної переміни в об’єкт з типом Visibility (рис. 5.3).

```
namespace SendfullUA.Common
{
    public class BooleanToVisibilityConverter : IValueConverter
    {
        public object Convert(object value, Type targetType, object parameter, string language)
        {
            Boolean val = (Boolean)value;
            Visibility visibility = Visibility.Collapsed;

            switch (val)
            {
                case true:
                    visibility = Visibility.Visible;
                    break;
                case false:
                    visibility = Visibility.Collapsed;
                    break;
            }

            return visibility;
        }

        public object ConvertBack(object value, Type targetType, object parameter, string language)
        {
            throw new NotImplementedException();
        }
    }
}
```

Рисунок 5.3 – Демонстрація конвертора «BooleanToVisibilityConverter»

5.3 Адаптивний дизайн додатку

Адаптивний дизайн – дизайн, що забезпечує правильне відображення додатку на різних пристроях, підключених до інтернету і динамічно підстроюється під задані розміри екрану.

Світ пристроїв на Windows 10 досить сильно фрагментований – тут і звичайні десктопи з широкоформатними моніторами, ноутбуки, нетбуки, і планшети, і смартфони, і гігантський Surface Hub, і, крім того, Xbox, Microsoft Band, пристрої, що використовують Raspberry Pi, а також окуляри віртуальної реальності HoloLense.

Всі ці пристрої можуть використовувати Windows 10, проте при цьому всі вони мають різний дозвіл екрана. В результаті один і той же додаток мо-

же нормально виглядати на робочому столі, але при цьому на мобільних пристроях мати не найкращий вигляд. Але на те і додатки під Windows 10 і є універсальними (про що і говорить назва платформи – UniversalWindows Platform), що дозволяють створювати універсальний адаптивний інтерфейс, який буде коректно відображатися на самих різних видах форм-факторів.

Найпростіший спосіб створення адаптивного дизайну полягає в ручній установці корисних властивостей для елементів управління в залежності від ширини екрану. І для цього можна використовувати обробник події SizeChanged (рис. 5.4).

Дана подія спочатку виникає при завантаженні програми, коли воно тільки розкривається. Потім воно повторно генерується при зміні розмірів додатки, наприклад, при зміні орієнтації екрану (з альбомної на портретну або навпаки) або при ручному стисканні/розширення меж вікна програми.

```
<VisualStateManager.VisualStateGroups>
    <VisualStateGroup x:Name="VisualStateGroup">
        <VisualState x:Name="LargeState">
            <VisualState.StateTriggers>
                <AdaptiveTrigger MinWindowWidth="940"/>
            </VisualState.StateTriggers>
            <VisualState.Setters>
                <Setter Target="splitView.(SplitView.DisplayMode)" Value="CompactInline"/>
                <Setter Target="splitView.(SplitView.IsPaneOpen)" Value="true"/>
                <Setter Target="searchBox.(UIElement.Visibility)" Value="Visible"/>
                <Setter Target="btnSearch.(UIElement.Visibility)" Value="Collapsed"/>
            </VisualState.Setters>
        </VisualState>
        <VisualState x:Name="NormalState">
            <VisualState.StateTriggers>
                <AdaptiveTrigger MinWindowWidth="450"/>
            </VisualState.StateTriggers>
        </VisualState>
        <VisualState x:Name="NarrowState">
            <VisualState.Setters>
                <Setter Target="splitView.(SplitView.DisplayMode)" Value="Overlay"/>
                <Setter Target="btnHome.(UIElement.Visibility)" Value="Collapsed"/>
                <Setter Target="HamburgerMenuMobile.(UIElement.Visibility)" Value="Visible"/>
            </VisualState.Setters>
            <VisualState.StateTriggers>
                <AdaptiveTrigger MinWindowWidth="0"></AdaptiveTrigger>
            </VisualState.StateTriggers>
        </VisualState>
    </VisualStateGroup>
</VisualStateManager.VisualStateGroups>
```

Рисунок 5.4 – Технологій для побудови адаптивного дизайну

5.4 Навігація в UWP

Перш за все треба відзначити, що додаток UWP має одне єдине вікно, яке представлено об'єктом Window. Вікно містить один єдиний фрейм, який займає весь простір вікна. Фрейм представлений об'єктом Frame. А фрейм вже містить сторінки (представлені класом Page), які, як правило, займають

весь простір кадру. Таким чином, в один момент часу ми можемо бачити тільки одну сторінку. І як правило, під навігацією технічно мається на увазі перехід від однієї сторінки до іншої (рис. 5.5).

```
protected override async void OnLaunched(LaunchActivatedEventArgs e)
{
    //задаем локальные файлы
    await LocalDataFileInit();


    //задаем культуру приложения
    var localSettings = Windows.Storage.ApplicationData.Current.LocalSettings;
    if (localSettings.Values["SelectedLanguage"] != null)
    {
        var culture = new System.Globalization.CultureInfo(localSettings.Values["SelectedLanguage"].ToString());
        System.Globalization.CultureInfo.CurrentCulture = culture;
        System.Globalization.CultureInfo.CurrentUICulture = culture;
        //дополнительная инициализация (тест)
        Windows.Globalization.ApplicationLanguages.PrimaryLanguageOverride = localSettings.Values["SelectedLanguage"].ToString();
    }




#if DEBUG
    if (System.Diagnostics.Debugger.IsAttached)
    {
        this.DebugSettings.EnableFrameRateCounter = true;
    }
#endif


    _rootFrame = new Frame();
    Frame rootFrame = Window.Current.Content as Frame;

    // Do not repeat app initialization when the Window already has content,
    // just ensure that the window is active
    if (rootFrame == null)
    {
        // Create a Frame to act as the navigation context and navigate to the first page
        rootFrame = new Frame();

        rootFrame.NavigationFailed += OnNavigationFailed;
        //rootFrame.Navigated += OnNavigated;

        if (e.PreviousExecutionState == ApplicationExecutionState.Terminated)
        {
            //TODO: Load state from previously suspended application
        }

        // Place the frame in the current Window
        Window.Current.Content = new MainPage(_rootFrame);

        SystemNavigationManager.GetForCurrentView().BackRequested += OnBackRequested;

        //SystemNavigationManager.GetForCurrentView().AppViewBackButtonVisibility =
        //    rootFrame.CanGoBack ?
        //    AppViewBackButtonVisibility.Visible :
        //    AppViewBackButtonVisibility.Collapsed;
    }

    if (rootFrame.Content == null)
    {
        // When the navigation stack isn't restored navigate to the first page,
        // configuring the new page by passing required information as a navigation
        // parameter
        rootFrame.Navigate(typeof(MainPage), e.Arguments);
    }
    // Ensure the current window is active
    Window.Current.Activate();
}
```

Рисунок 5.5 – Розробка навігаційних можливостей додатку

Панель навігації являє один з поширених способів навігації по розділах всередині програми. Разом з панеллю навігації, як правило, використовується спеціальна кнопка із зображенням трьох смужок. За ці три смужки подібну організацію меню назвали «гамбургером» [13]. При натисканні на кнопку панель навігації розкривається, відображаючи список елементів меню для вибору, а при повторному натисканні закривається. Панель навігації добре підходить для тих випадків, коли в нашому додатку є кілька різних розділів і коли необхідно, щоб меню ховалося, звільняло місце основного контенту сторінки (рис. 5.6).

```
<StackPanel Name="HamburgerMenuMobile"  Grid.Row="0" Orientation="Horizontal" Visibility="Collapsed">
    <AppBarButton Name="btnHomeMobile" Background="{ThemeResource mySplitView}"  Canvas.ZIndex="1" Height="48" Width="48" VerticalAlignment="Top" HorizontalAlignment="Left" Click="btnHome_Click" >
        <AppBarButton.Icon>
            <FontIcon FontSize="20" FontFamily="Segoe MDL2 Assets" Glyph="&#xE10F;"/>
        </AppBarButton.Icon>
    </AppBarButton>

    <TextBlock Name="labelTitle" Margin="15,0,0,0" FontSize="22" VerticalAlignment="Center">Sendfull</TextBlock>
</StackPanel>
```

Рисунок 5.6 – Розробка меню «Гамбургер»

5.5 Стилі та діалогові вікна додатку

Стилі застосовуються для створення однакового відображення елементів, також вони спрощують підключення групи ресурсів до окремих елементів. Стилі дозволяють визначити набір деяких властивостей і їх значень, які потім можуть застосовуватися до елементів в xaml [14]. Стилі зберігаються в ресурсах і відокремлюють значення властивостей елементів від призначеного для користувача інтерфейсу. Аналогом стилів можуть служити каскадні таблиці стилів (CSS), які застосовуються в коді html на веб-сторінках (рис. 5.7).

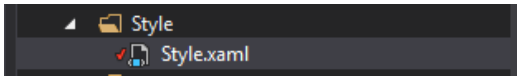

Рисунок 5.7 – Директорія файлів з стилями

Стиль створюється як ресурс за допомогою об'єкта Style, який представляє клас Windows.UI.Xaml.Style. Як у будь-якого ресурсу, у стилю встановлюється ключ за допомогою атрибута x: Key, а також властивість TargetType – воно вказує на тип елементів, до яких застосовується стиль.

Клас ContentDialog використовується для створення діалогових вікон. Діалогове вікно можна створити у вигляді коду або розмітки (рис. 5.8).

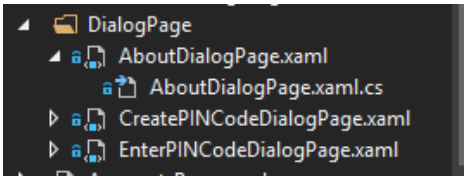

Рисунок 5.8 – Директорія файлів з діалоговими вікнами

Для діалогового вікна можна задати ряд властивостей. Перш за все, властивість Title встановлює заголовок вікна, а властивість Content – його

текст. Властивість PrimaryButtonText визначає текст на першій кнопці, а SecondaryButtonText – на другий (рис. 5.9). Для відображення діалогового вікна треба викликати асінхонний метод ShowAsync(). Його результатом є об'єкт ContentDialogResult, з якого ми можемо дізнатися яку кнопку натиснув користувач [15]. Якщо результат дорівнює ContentDialogResult.Primary, то натиснута перша кнопка (рис. 5.10).

```
<ContentDialog
    x:Class="SendfullUA.View.DialogPage.AboutDialogPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:local="using:SendfullUA.View.DialogPage"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d"
    Title="About"
    x:Uid="dialogPage_About"
    SecondaryButtonText="Close"
    SecondaryButtonClick="ContentDialog_SecondaryButtonClick" Loaded="ContentDialog_Loaded">

    <Grid>
        <ScrollViewer VerticalScrollBarVisibility="Hidden">
            <StackPanel>
                <Image Height="130" Width="130" Source="ms-appx:///Assets/YakoLogoNik.png"/>
                <TextBlock x:Uid="textBlock_DevelopedBy" HorizontalAlignment="Center" Text="Developed by:"/>
                <TextBlock x:Uid="textBlock_FIO" FontSize="18" HorizontalAlignment="Center" Text="Nikita Yakovenko"/>
                <StackPanel Name="stackPanel" Orientation="Horizontal" HorizontalAlignment="Center"/>
                <TextBlock Name="labelBuild" HorizontalAlignment="Center" Text="Build: "/>
            </StackPanel>
        </ScrollViewer>
    </Grid>
</ContentDialog>
```

Рисунок 5.9 – Файл розмітки діалогового вікна

```
private void ContentDialog_Loaded(object sender, RoutedEventArgs e)
{
    if (Convert.ToInt32(Windows.Storage.ApplicationData.Current.LocalSettings.Values["SelectedTheme"]) == 1)
        RequestedTheme = ElementTheme.Light;
    else
        RequestedTheme = ElementTheme.Dark;

    var build = Windows.ApplicationModel.Package.Current.Id.Version;
    labelBuild.Text += build.Major + "." + build.Minor + "." + build.Build;


    for (int i = 0; i < content.Length; i++)
    {
        BitmapIcon bitmapIcon = new BitmapIcon();
        bitmapIcon.Margin = new Thickness(5, 5, 5, 5);
        if (Convert.ToInt32(Windows.Storage.ApplicationData.Current.LocalSettings.Values["SelectedTheme"]) == 1)
            bitmapIcon.Foreground = new SolidColorBrush(Windows.UI.Colors.DimGray);
        else
            bitmapIcon.Foreground = new SolidColorBrush(Windows.UI.Colors.AliceBlue);
        bitmapIcon.Height = 40;
        bitmapIcon.Width = 40;
        bitmapIcon.Name = "i" + i;
        bitmapIcon.UriSource = new Uri("ms-appx:///Assets/" + content[i] + "_icon.png");
        bitmapIcon.Tapped += bitmapIcon_Tapped;
        stackPanel.Children.Add(bitmapIcon);
    }
}
```

Рисунок 5.10 – Файл коду діалогового вікна

6 ПРОЦЕС СЕРТИФІКАЦІЇ ТА ПУБЛІКАЦІЇ ДОДАТКУ

6.1 Процес сертифікації програмного продукту

Перевірка додатку для Windows за допомогою комплекту сертифікації додатків для Windows з командного рядка [16].

Комплект сертифікації додатків для Windows повинен виконуватися в контексті активного сеансу користувача.

1) У командному вікні необхідно перейти в каталог, що містить комплект сертифікації додатків для Windows.

Шлях за замовчуванням – C:\Program Files\Windows Kits\10\App Certification Kit\.

2) Далі ввести наступні команди в зазначеному порядку для тестування програми, яке вже встановлено на комп'ютері тестування: `appcert.exe reset`

```
appcert.exe test -packagefullname [package full name] -reportoutputpath [report file name]
```

Або можна використовувати наступні команди, якщо додаток не встановлено.

Комплект сертифікації додатків для Windows відкриє пакет і застосує відповідний робочий процес перевірки: `appcert.exe reset`

```
appcert.exe test -appxpackagepath [package path] -reportoutputpath [report file name]
```

3) Після закінчення тестування відкрийте файл звіту з ім'ям [report file name] і перевірте результати тесту.

Після завершення створення відправки додатки, необхідно клацнути «Відправити в Store», щоб перейти на крок сертифікації. Як правило, цей процес повинен бути завершений протягом декількох годин, хоча в деяких випадках може зайняти до трьох робочих днів.

Після сертифікації відправки може пройти до 24 годин, перед тим як користувачі побачать опис програми (або поновлення раніше опублікованого додатка) в «Магазині».

Після публікації можна побачити повідомлення, а стан додатки на інформаційній панелі зміниться на «В магазині».

Після успішного відправлення пакетів додатки, спрямованого на сертифікацію, пакети ставляться в чергу на тестування. Якщо в процесі попередньої обробки буде виявлена помилка, з'явиться відповідне повідомлення.

6.2 Тестування та публікація програмного продукту

На цьому етапі проводиться кілька тестів:

− тести безпеки – в ході першого тесту пакети вашого застосування перевіряються на наявність вірусів і шкідливого ПЗ. Якщо програма не проходить цей тест, то знадобиться перевірити систему розробки, запустивши актуальне антивірусне ПЗ, а потім знову зібрати пакет додатка в чистій системі;

− перевірка відповідності технічним вимогам – відповідність технічним вимогам перевіряється за допомогою комплекту сертифікації додатків для Windows;

− відповідність вмісту вимогам – тривалість цього етапу залежить від складності додатка, від обсягу його візуального вмісту та кількості додатків, відправлених за останній час. Вкажіть корисну для тест-інженерів інформацію на сторінці «Нотатки» по сертифікації.

Після завершення процесу сертифікації необхідно отримати звіт з повідомленням про те, чи пройшло ваше додаток сертифікацію. Якщо програма не пройшло сертифікацію, то в звіті буде показано, який саме тест не пройдено або вимоги якої політики виявилися невиконаними. Після усунення проблеми можна створити нову відправку для вашого застосування для повторного запуску процедури сертифікації.

Якщо додаток пройшло сертифікацію, воно готове для переходу до етапу «Публікація». Якщо вказано, що відправка повинна бути опублікована якомога швидше, вона буде опублікована негайно. Якщо вказати, що її не слід випускати до настання певної дати, ми почекаємо до цієї дати, за умови, що не буде натиснуто посилання «Змінити» дату випуску. Якщо вказано, що хочеться опублікувати відправку вручну, вона не будемо публікувати, поки не буде натиснута кнопку «Опублікувати» або не клацнуте посилання «Змінити дату випуску» і не обрана іншу дата.

Пакети додатку отримують цифровий підпис для захисту від незаконної зміни після випуску. Після початку цього етапу можна скасувати відправку або змінити дату випуску.

Коли додаток знаходиться на етапі публікації, посилання «Детальніше про твіт» в стовпці «Стан відправки» дозволить дізнатися про нові пакети і опис програми в магазині стане доступними клієнтам на кожній підтримуваній версії ОС. Дії, які ще не були завершені будуть позначені повідомленням «Очікування». Додаток залишиться на етапі публікації до завершення процесу і появи нових пакетів і описів для всіх потенційних користувачів вашого додатка. Це може зайняти до 24 годин.

Після успішного завершення описаних вище етапів, стан відправки зміниться з «Публікація» на «В магазині». Після цього додаток стане доступно для скачування в Microsoft Store (якщо користувач не вибрав інше значення параметра «Можливість виявлення»).

7 ОПИС КОРИСТУВАЦЬКОГО ІНТЕРФЕЙСУ ТА ЙОГО МОЖЛИВОСТЕЙ

При першому запуску програми, користувача зустрічає повідомлення з інформацією про те що він використовує тестову версію програмного забезпечення (рис. 7.1). Після закриття цього його відкривається головне вікно (рис. 7.2), яке містить базові елементи управління та головне меню, завдяки якому можна пересуватися між різноманітними вікнами програми.

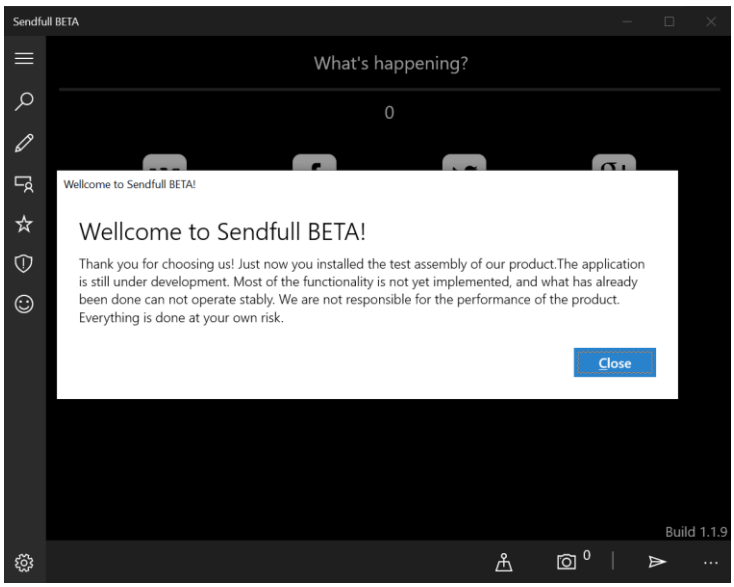

Рисунок 7.1 – Повідомлення про використання бета версії додатку

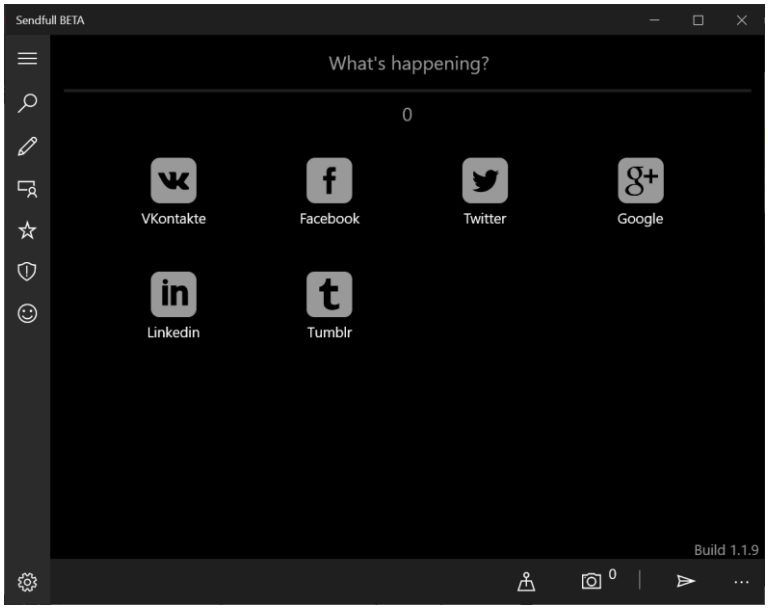

Рисунок 7.2 – Головне вікно програми

Додаток є повністю «адаптивним», що дозволяє виглядати добре на будь-якому пристрої: будь то телефон або велика діагональ телевізора (рис. 7.3).

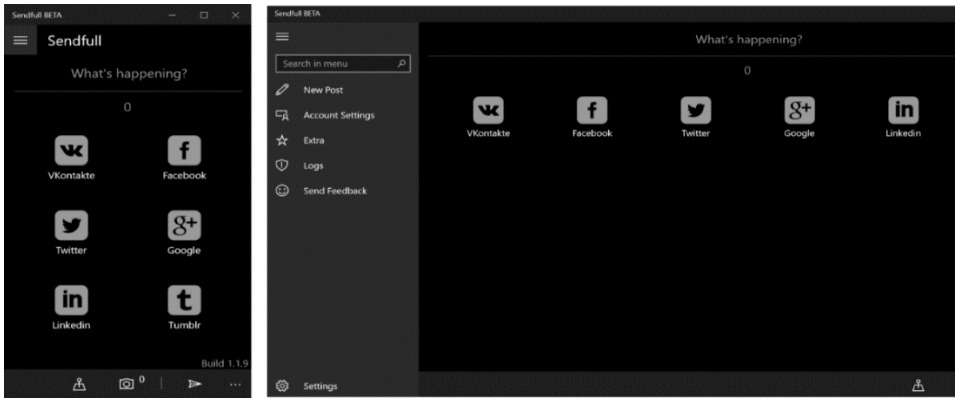

Рисунок 7.3 – Демонстрація «адаптивного» дизайну

У головному меню програми присутні такі пункти (рис. 7.4):

– «Новий пост»
– «Налаштування акаунтів»
– «Додатково»
– «Логи»
– «Відправитивідгук»
– «Налаштування»

При натисканні по кожному з елементів меню ми будемо потрапляти до відповідної функціональності.

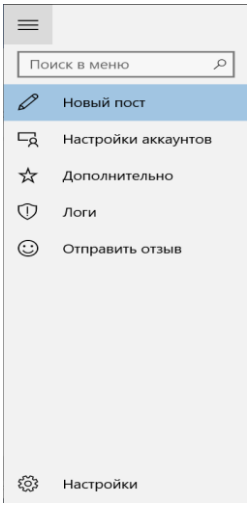

Рисунок 7.4 – Меню програми

Тепер більш детально пройдемося по кожному пункту меню:

«Новий пост». У головному вікні програми можна побачити область для роботи з соціальними мережами. У ній користувач може вибрати необхідну йому соціальну мережу в які він хоче зробити публікацію, а також налаштувати саму публікацію, вказавши наприклад які документи він хоче прикріпити до неї, вказувати геолокацію в публікації чи ні і т.д. Всі додаткові настройки по публікаціям можна налаштувати у відповідному меню додатоку «Додатково».

У верху вікна ми можемо побачити область з текстовим полем, в якому користувач повинен вводити текст для майбутньої публікації (рис. 7.5). Під даним полем встановлений лічильник, який відповідальний за підрахунок символів введених користувачем. Це потрібно наприклад для «Twitter» – соціальної мережі в якій обмежено кількість символів в інформації, що публікується записи.

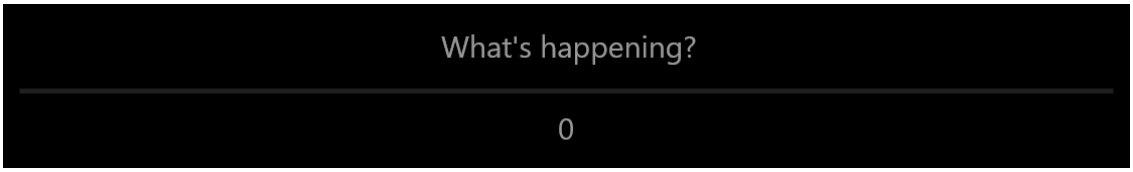

Рисунок 7.5 – Поле для введення тексту публікації

Під полем введення можна бачити різну кількість іконок соціальних мереж (рис. 7.6). Вони потрібні для того, що б вибрати на який портал користувач хоче здійснено публікацію. При натисканні на іконці, вона виділяється кольором, показуючи те, що буде здійснюється відправка тексту в дану соціальну мережу. Якщо користувач не авторизований в обраній мережі, відкриється вікно з пропозицією «увійти» на портал і зробити це (рис. 7.7). Після чого інформація буде збережена в додатку і використовуватися в подальших запитах.

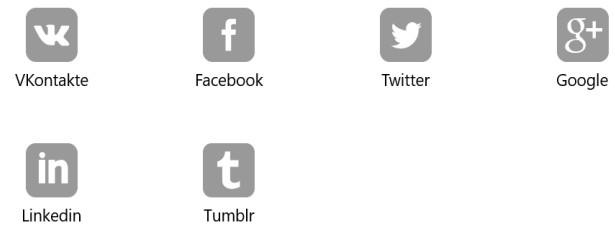

Рисунок 7.6 – Область для вибору соціальної мережі

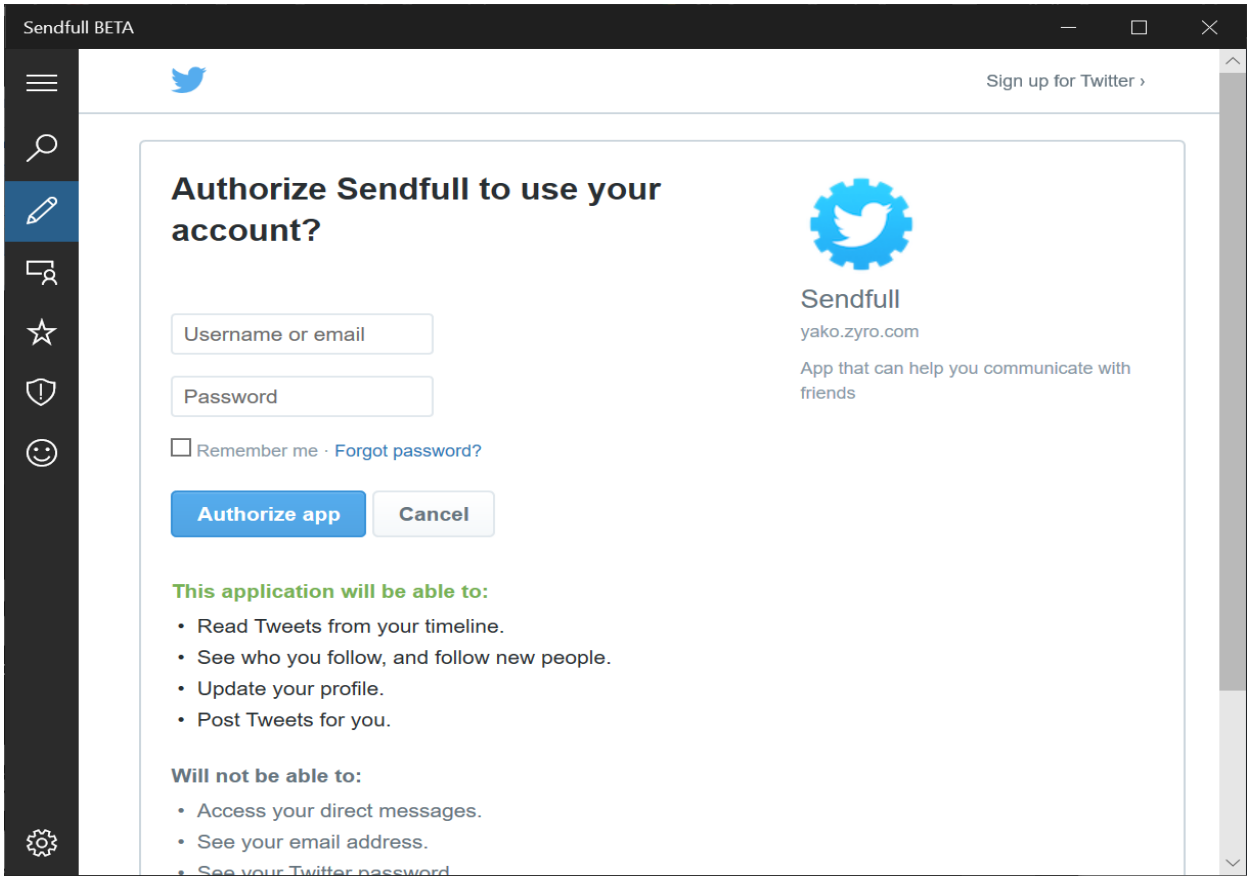

Рисунок 7.7 Вікно авторизації в соціальній мережі – Twitter

У самому низу вікна розташована область в який можна налаштувати способи публікації, вказавши додаткову інформацію, яку хочеться додати до повідомлення (рис. 7.8).

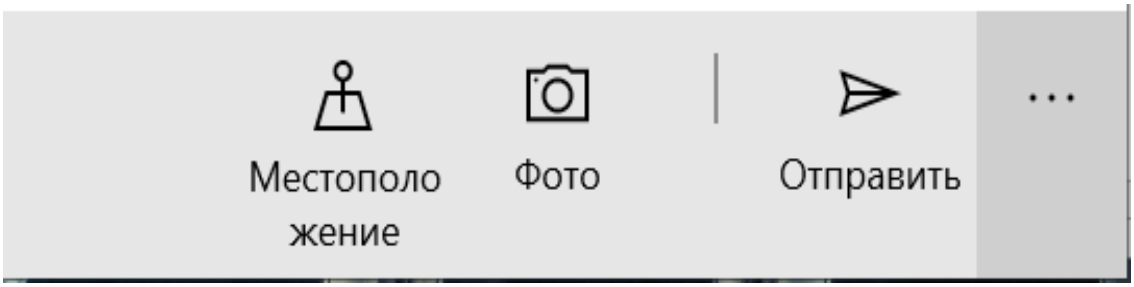

Рисунок 7.8 – Інструменти для настройки публікації

Якщо користувач натискає на кнопку «Відправити» – починається процес створення публікацій з раніше заданим критеріям. Після завершення даного процесу буде відображено спливаюче повідомлення яке говорить про стан створюваного поста (рис. 7.9).

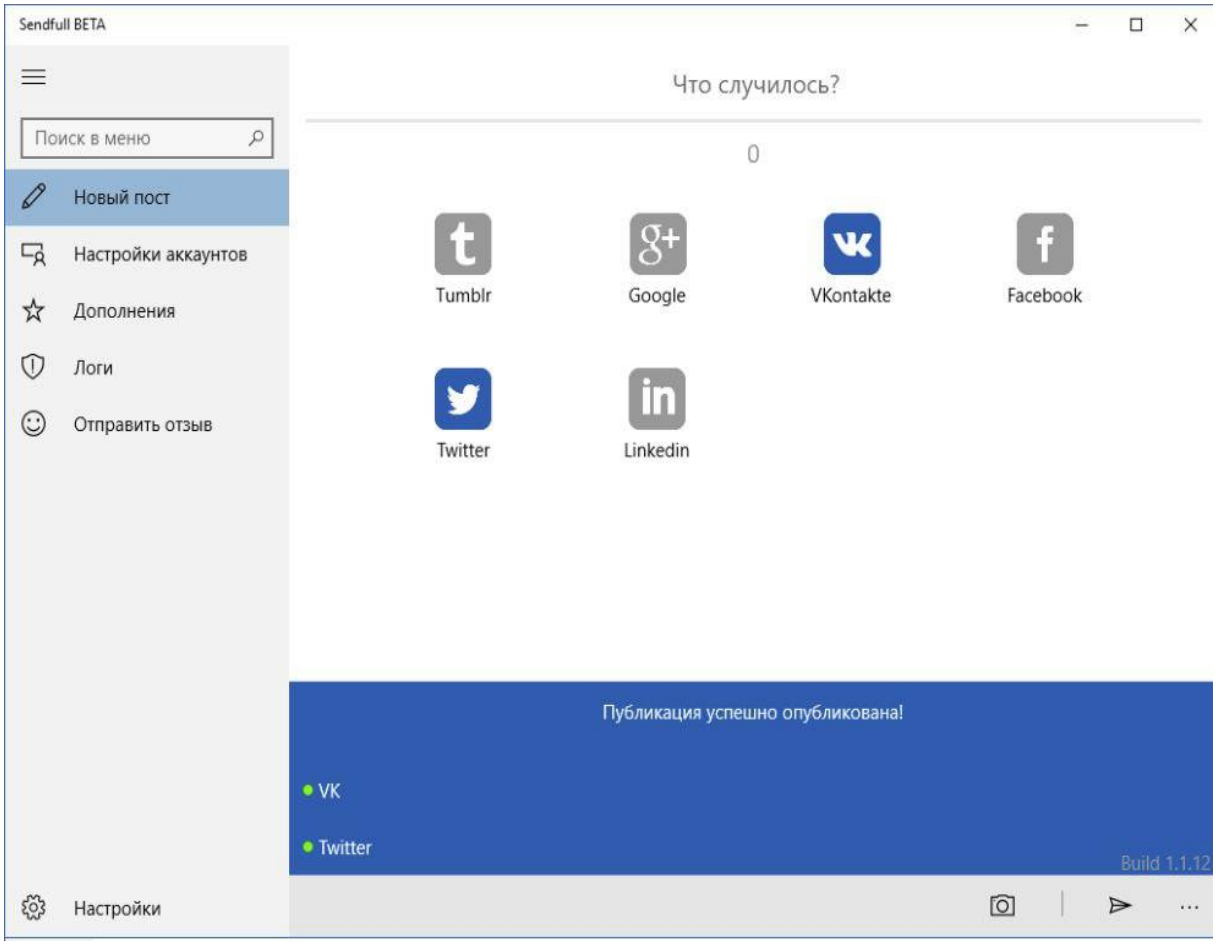

Рисунок 7.9 – Процес створення публікації

«Налаштування»

В даному пункті меню, можна налаштувати додаток бажаним чином. Кожна дія в цьому меню супроводжується відповідними повідомленнями в нижній частині додатка (рис. 7.10).

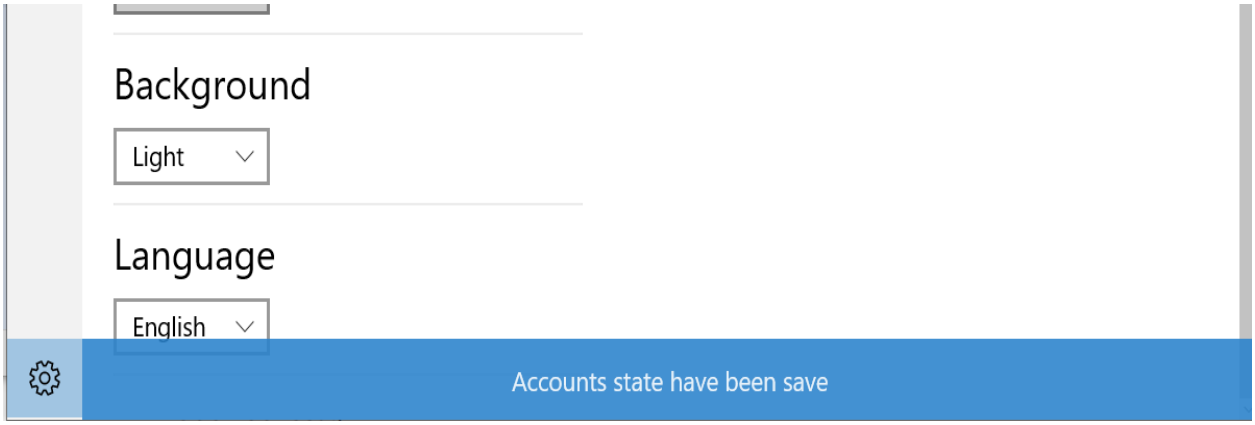

Рисунок 7.10 – Рядок повідомлень

У верхній частині вікна ми можемо налаштувати таку функціональність:

– «PIN»
– «Підпис»
– «Акаунти»
– «Тема»
– «Мова»

Розглянемо кожен з них детальніше.

«PIN»

При виборі цього пункту меню, користувачеві надається можливість по встановленню пароля на додаток. Яке буде з'являється при кожному наступному запуску програми. Ця функція дозволить захистити налаштовані акаунти від несанкціонованого доступу (рис. 7.11).

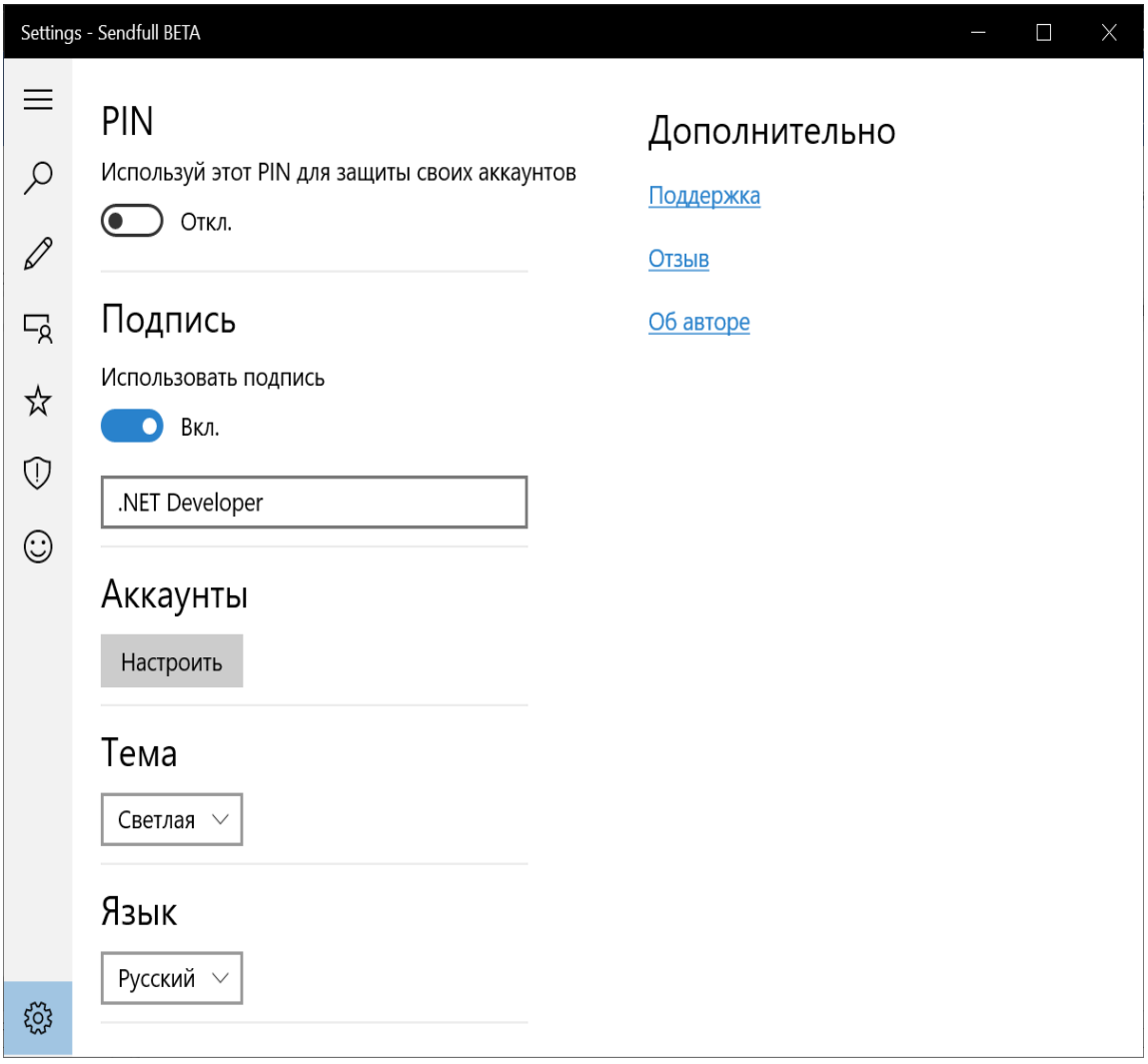

Рисунок 7.11 – Меню «Налаштування»

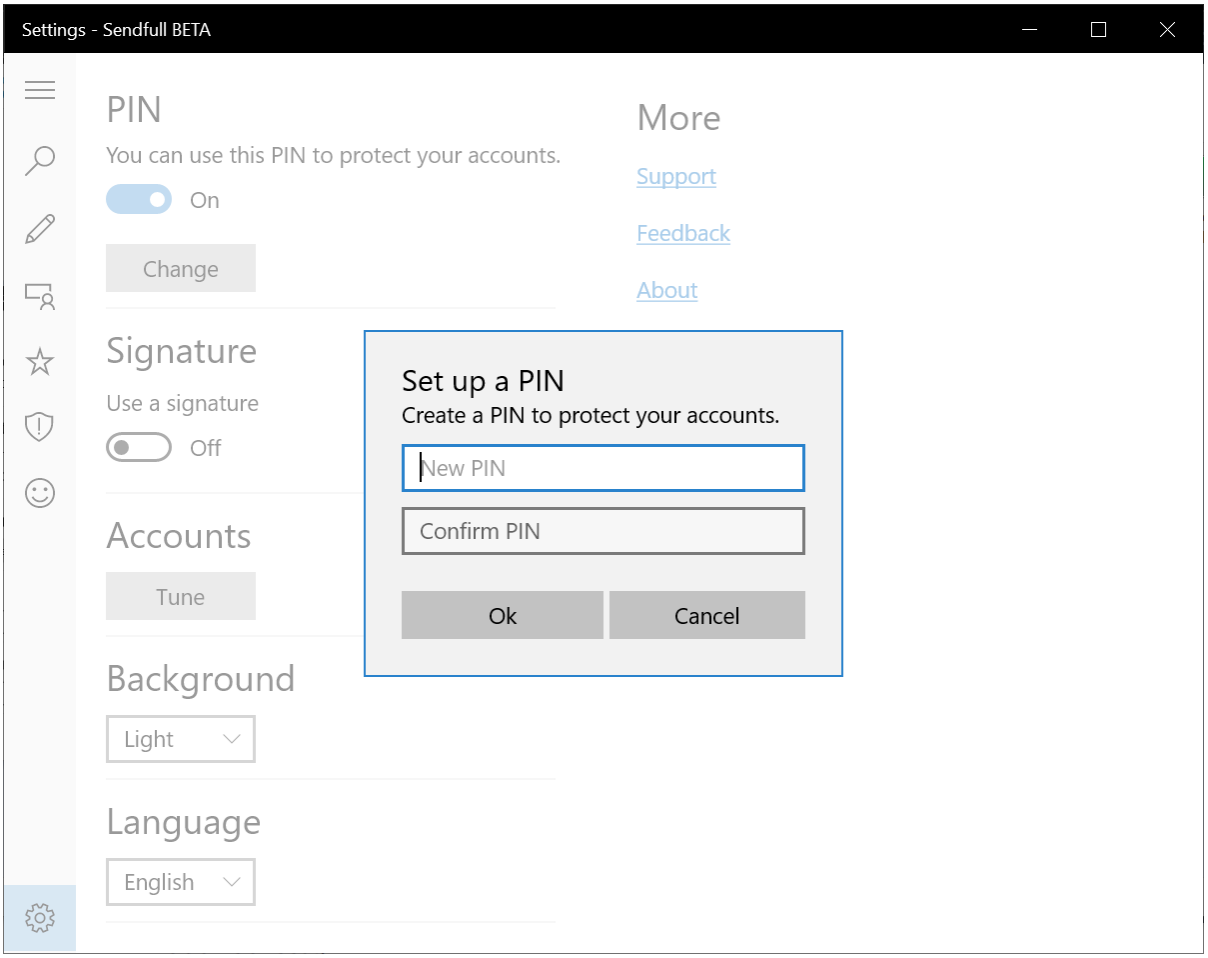

Рисунок 7.12 – Вікно настройки пароля «Підпис»

У даній опції можна налаштувати текст повідомлення, яке буде додаватися після кожного відправленого поста в обрані соціальні мережі (рис. 7.13).

Рисунок 7.13 – Меню встановлення підпису

«Акаунти»

Ця область створена для управління соціальними мережами, що будуть доступні з головного меню (рис. 7.14). Тут можна змінити їх порядок, а також включити або вимкнути необхідні в залежності від потреб.

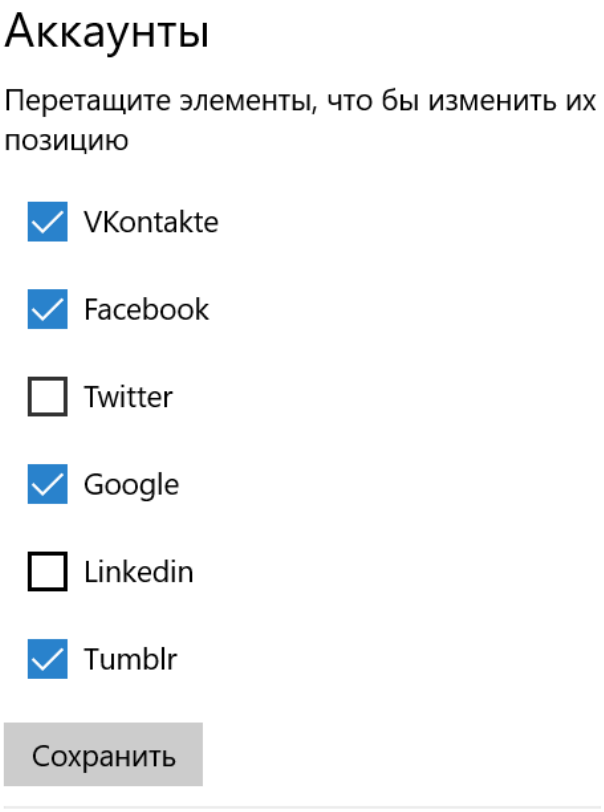

Рисунок 7.14 – Область настройки використовуваних соціальних мереж

«Темы»

Ця опція відповідає за зміну оформлення програми. Доступно дві теми: темна і світла. За замовчуванням береться тема з ОС Windows.

«Языки»

Тут можна вибрати мову інтерфейсу програми. Доступно дві мови: російська та англійська. За замовчуванням береться мова як і в ОС Windows, якщо такої немає, вибирається мова за замовчуванням – англійська.

«Поддержка»

Через цю опцію можна зв'язатися з розробником програми. Для уточнення будь-яких питань.

«Отзывы»

Переадресує на сторінку додатка в Windows Store. В якому можна залишити відгук про програмному продукті.

«Про автора»

При натисканні на цьому меню. З'являється спливаюче вікно, в якому можна побачити інформацію про автора програми (рис. 7.15).

«Настройка акаунтов»

Дане меню дозволяє настроїти список доступних соціальних мереж. До вибору користувача доступні такі пункти як:

– «Mail»
– «VKontakte»
– «Facebook»
– «Twitter»
– «Google+»
– «Linkedin»
– «Tumblr»

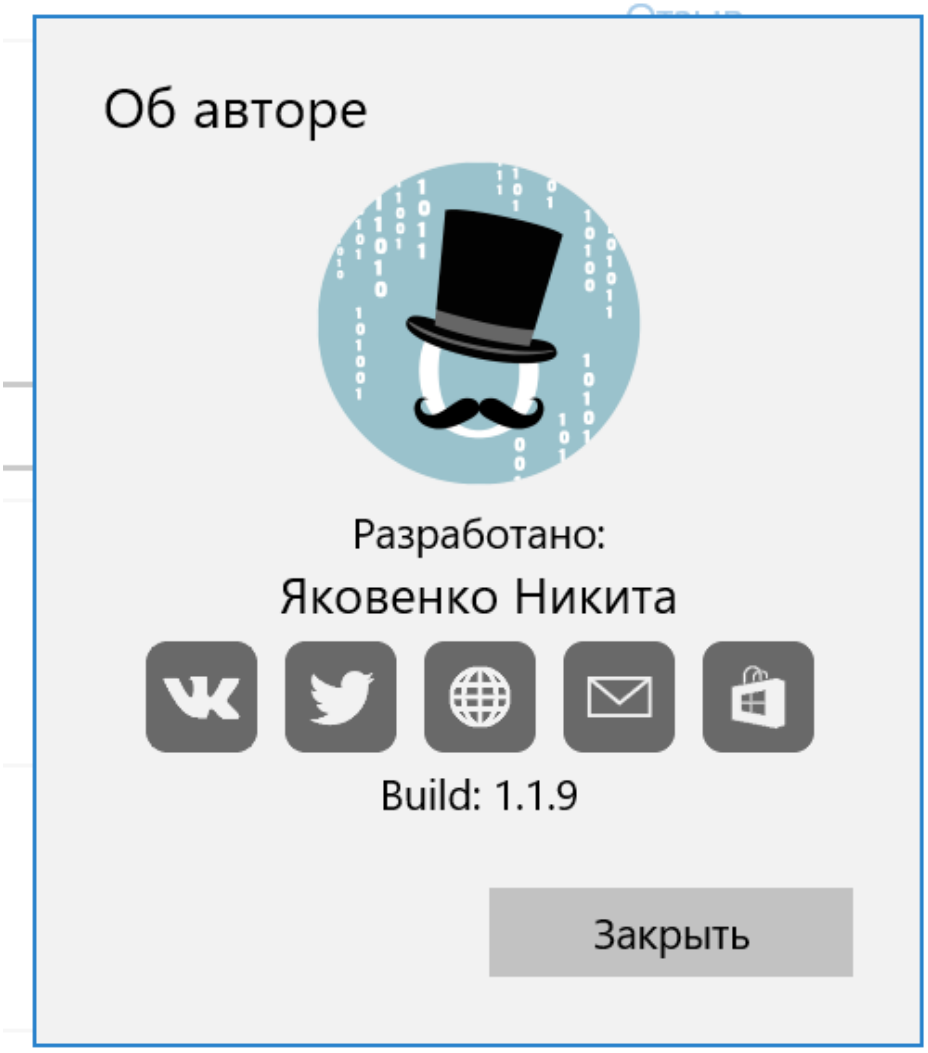

Рисунок 7.15 – Вікно інформації про автора

При натисканні на кожному з них користувач потрапляє у вікно налаштування облікового запису, де він може Авторизуватися, вказати яким чином він хоче робити пости, це може бути приватні повідомлення собі на «стінку», в групу, робити записи від свого імені або від імені групи та інше. Ці настройки унікальні для кожної соціальної мережі (рис. 7.16).

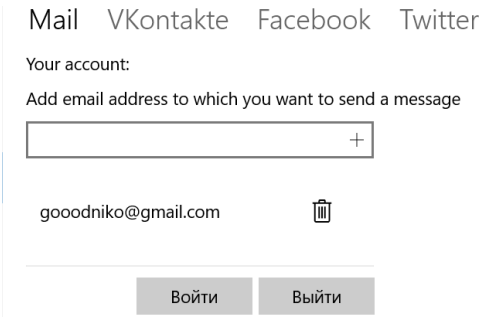

Рисунок 7.16 – Меню налаштувань обраної соціальної мережі

«Додатково»

Ця вкладка відповідає за налаштування «плагінів – доповнень» які покращують способи взаємодії з системою, на даній вкладці вбудований магазин розширень в якому користувач може встановити те чи інше розширення (рис. 7.17). А також можливість включити або виключити розширення в залежності від потреби.

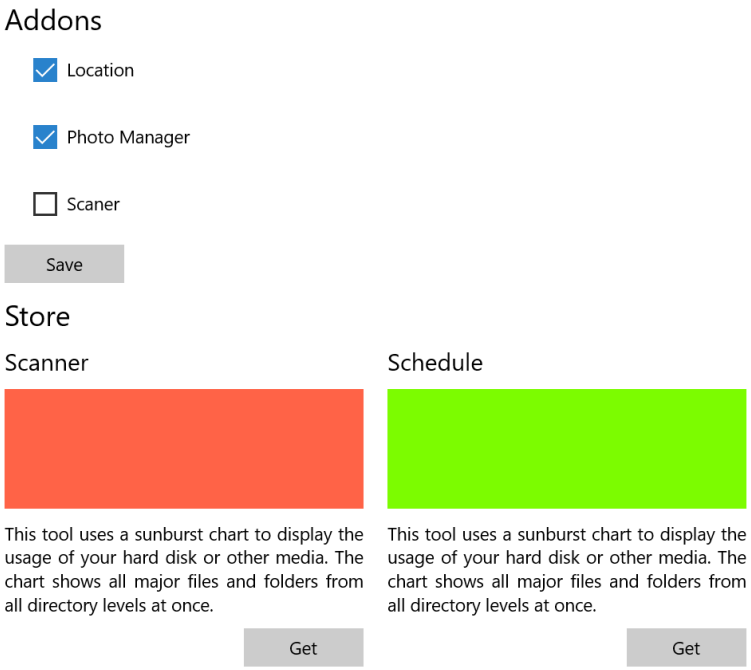

Рисунок 7.17 – Вікно доповнень

При використанні розширень які вимагають яких-небудь угод з боку користувача буде виведено діалогове вікно в якому попросять дати угоду або ж доповнення не буде включено (рис. 7.18).

Приклад розширення «Фото менеджер». Дане доповнення дозволяє прикріплювати бажані зображення до публікацій. Можна вказати будь-яку кількість різних картинок які ми хотіли б бачити з публікацією і після натискання на кнопку відправити, вони будуть відправлені (рис. 7.19). Іконка цього розширення є динамічною і змінює кількість файлів, що додаються в режимі реального часу.

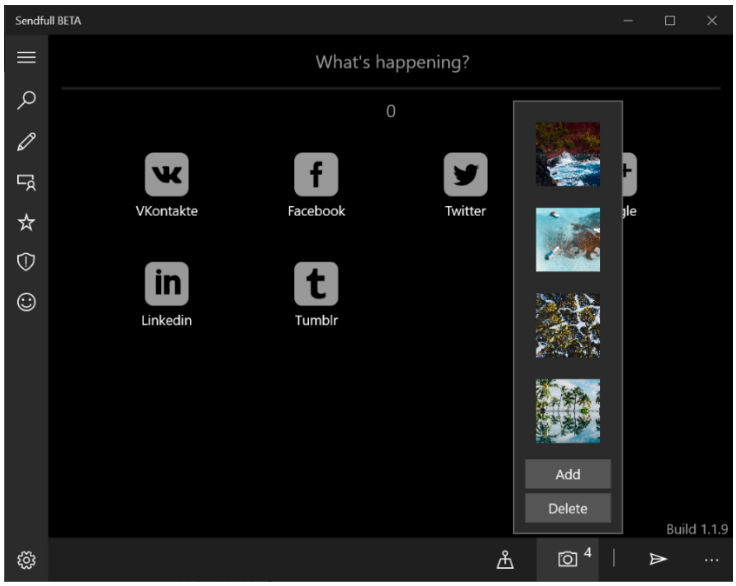

Рисунок 7.18 – Менеджер фотографій

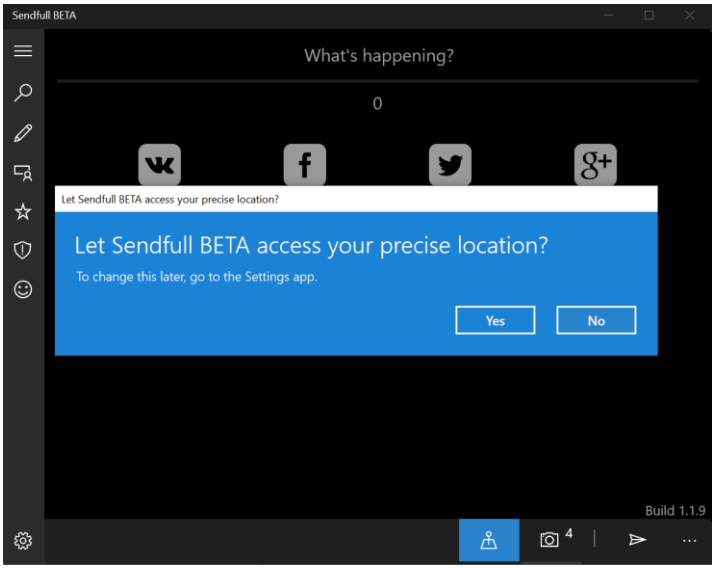

Рисунок 7.19 – Діалогове вікно доступу

«Логи»

У цьому вікні користувач отримує доступ до всієї історії публікація додатку. Тут можна фільтрувати за різними критеріями зроблені раніше публікації. Дивитися їх статус, а також можна перейти безпосередньо до посту, якщо натиснути по запису (рис. 7.20).

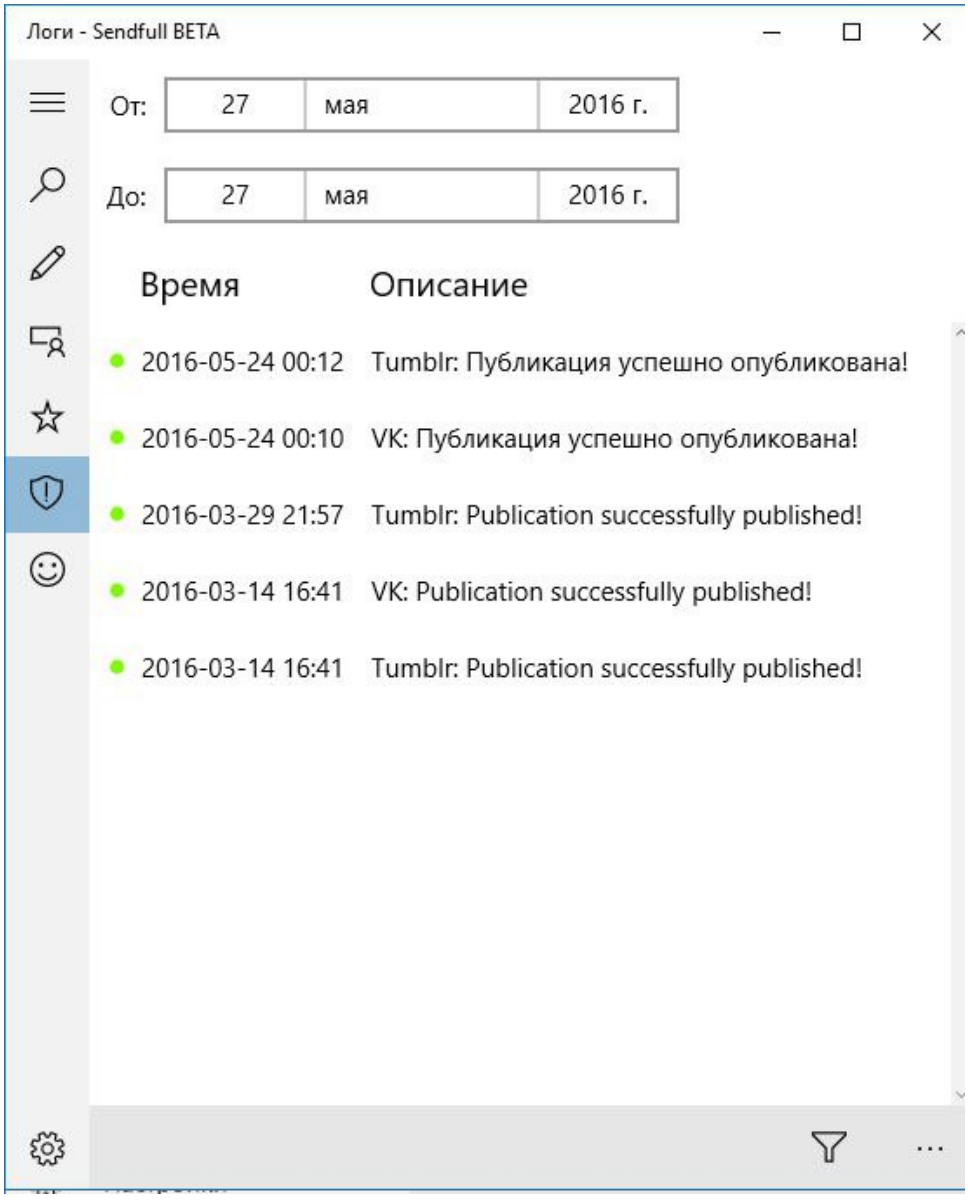

Рисунок 7.20 – Меню перегляду історій публікації

«Надіслати відгук»

При натисканні на даному меню додатка користувач перенаправляється в магазин додатків, на сторінку даного ПЗ, на якій він може подивитися більш детальну інформацію про нього, а також залишити відгук якщо він цього бажає (рис. 7.21).

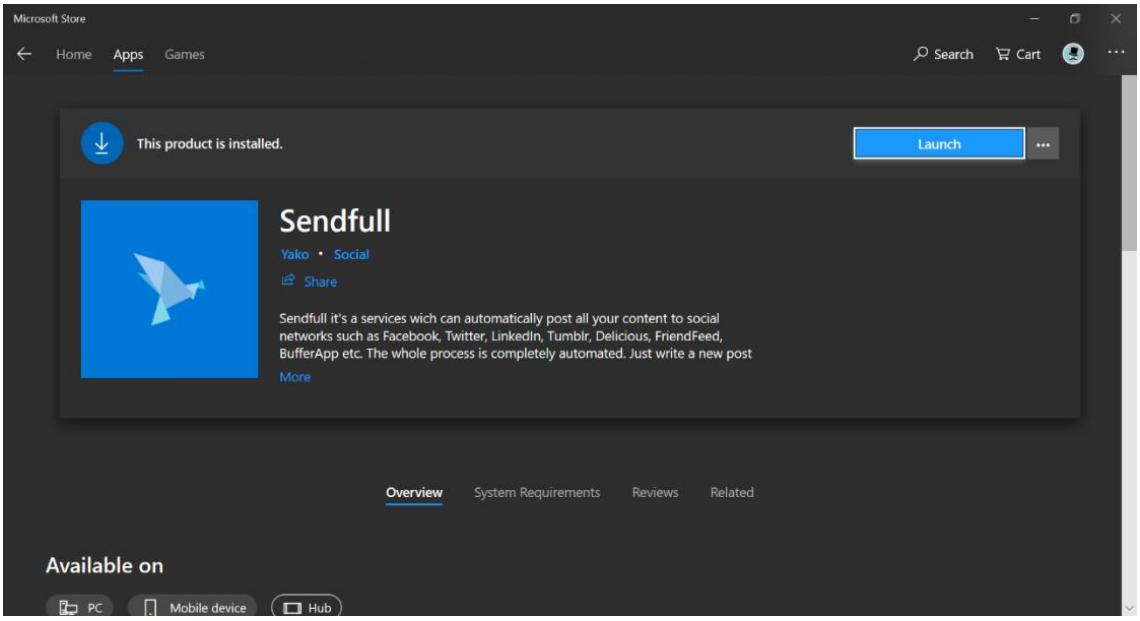

Рисунок 7.21 – Додаток в магазині Microsoft Store

Додаток вбудовано в ОС Windows і має можливість взаємодіяти з іншими додатками через контекстне меню функції «Share» (рис. 7.22). Наприклад якщо користувач працював над сценарієм для своєї публікації в OneNote і після завершення роботи над нею, він може в один клік відправити свої ідеї в додаток Sendfull, який підготує їх та в звичній формі дозволить відправити їх між усіма вибраними соціальними мережами (рис. 7.23).

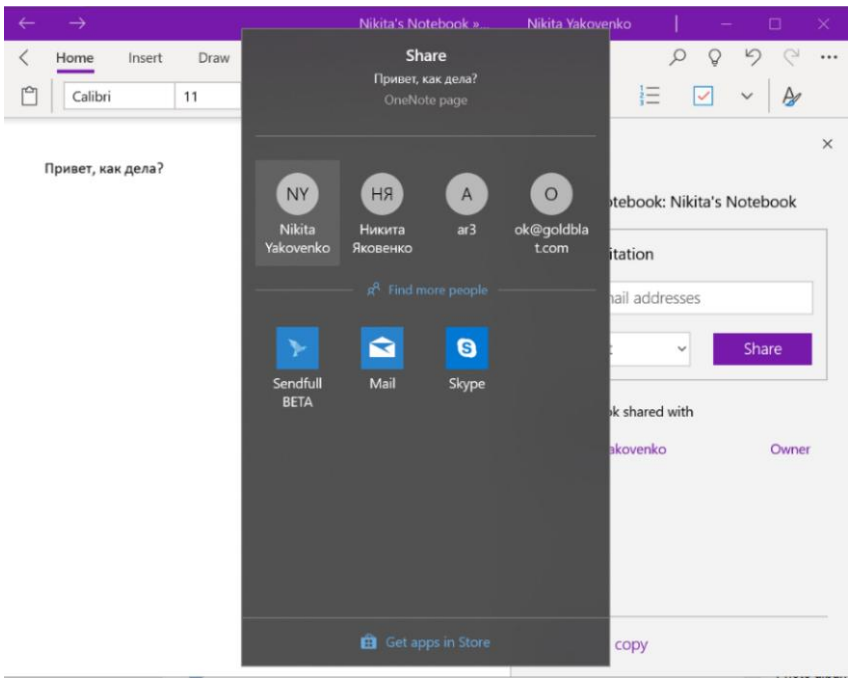

Рисунок 7.22 – Контекстне меню «Share»

Рисунок 7.23 – Адаптивна версія додатку для роботи з контекстом «Share»

ВИСНОВКИ

В магістерській роботи було виконано проектування та розробку соціального агрегатору під платформу Universal Windows Application. Завдяки якої був створений додаток для: Windows 10 Mobile, Windows 10 Desktop, IoT, та для ігрової платформи Xbox One.

Даний програмний продукт має наступну функціональність, він допомагає робити публікацію декількох соціальних мереж на одній сторінці. Така можливість мати один універсальний додаток для декількох соціальних мереж – просто необхідність для зайнятого та онлайн-активного користувача.

Доступність системи і можливість роботи з даними без використання спеціальних знань в програмуванні повинні дозволити широкому колу користувачів сконцентруватися на вирішенні конкретних завдань, а також забезпечити гарантії достовірності отриманих результатів.

Розроблений продукт відрізняється від аналогів своєю гнучкістю, можливостями повністю налаштувати користувацький інтерфейс під свої потреби. Великою кількістю підтримуваних соціальних мереж, а також можливістю завантажувати різноманітні розширення для більш комфортної взаємодії з додатком.

Тестування програмного продукту показало, що додаток відміно виконує свої функції, швидко працює на усіх підтримуваних платформах, та придатне для використання у робочих цілях.

ПЕРЕЛІК ПОСИЛАНЬ

1. Everypost. [Електронний ресурс] – Режим доступу: everypost.me/
2. Sprout Social. [Електронний ресурс] – Режим доступу: socialbakers.com
3. Crowdbooster. [Електронний ресурс] – Режим доступу: crowdbooster.io
4. SocialFlow. [Електронний ресурс] – Режим доступу: socialflow.com/
5. Buffer. [Електронний ресурс] – Режим доступу: https://buffer.com/
6. Руководство по работе с приложениями универсальной платформы Windows (UWP). [Електронний ресурс] – Режим доступу: https://docs.microsoft.com/en-us/windows/uwp/get-started/universal-application-platform-guide
7. Герасимов, О. В. Коллективная разработка функциональной модели информационной системы [Електронний ресурс] / О. В. Герасимов. – Режим доступа: http://www.ict.edu.ru/vconf/files/7316.doc, свободный. – Название с экрана.
8. Дженнингс, Роджер. Использование Microsoft Access 2000. Специальноеиздание [Текст]: учеб. пос. / РоджерДженнингс. – М.: Издательский дом «Вильямс», 2000. – 1147 с.
9. Єрьоміна, Н. В. Проектування баз даних [Текст]: навч. посібник / Н. В. Єрьоміна. – К.: КНЕУ, 1998. – 208 с.
10. Кузин, А. В. Базы данных [Текст]: учеб. пособ. для студ. высш. учеб. заведений / А. В. Кузин, С. В. Левонисова. – 2-е изд. – М.: Издательский центр «Академия», 2008. – 320 с.
11. Рыбанов, А. А. Инструментальные средства автоматизированного проектирования баз данных [Електронний ресурс] / А. А. Рыбанов. –Режим доступа: http://window.edu.ru/window_catalog/redir?id= 47119&file=rybanov_
12. Создание таблицы в режиме конструктора [Електронний ресурс]. – Режим доступа: http://www.officepack.ru/access/11/, свободный. – Название с экрана.
13. Ткаченко, В. А. Системы управления базами данных и экспертные системы [Электронный ресурс] / В. А. Ткаченко. – Режим доступа: http://www.lessons-tva.info/edu/e-inf2/m2t4.html, свободный. – Название с экрана.
14. Карпов Т.С. Базы данных: модели, разработка, реализация. – СПб.: Питер, 2001. – 304 с.

15. Дейт К.Дж. Введение в системы баз данных: Пер. с англ. № 6-е изд. Н.К.: Диалектика, 1998. – 784 с.

16. Процесс сертификации приложения. [Електронний ресурс] – Режим доступу: docs.microsoft.com/ru-RU/windows/uwp/publish/the-app-certification-process