Національний університет «Полтавська політехніка імені Юрія Кондратюка»
(повне найменування вищого навчального закладу)
Навчально-науковий інститут інформаційних технологій та робототехніки
(повна назва факультету)
Кафедра комп'ютерних та інформаційних технологій і систем
(повна назва кафедри)

Пояснювальна записка
до дипломного проекту (роботи)
магістра
(освітньо-кваліфікаційний рівень)
на тему

Програмне забезпечення з графічним інтерфейсом для створення й обробки QR-кодів

Виконав: студент 6 курсу, групи 602-ТН
спеціальності
122 Комп'ютерні науки
(шифр і назва напряму)
Бобоха Я. Ю.
(прізвище та ініціали)
Керівник Головко Г. В.
(прізвище та ініціали)
Рецензент _____
(прізвище та ініціали)

Полтава – 2024 року

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИНАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ПОЛТАВСЬКА ПОЛІТЕХНІКА ІМЕНІ ЮРІЯ КОНДРАТЮКА»

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА РОБОТОТЕХНІКИ

КАФЕДРА КОМП'ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І СИСТЕМ

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА
спеціальність 122 «Комп'ютерні науки»

на тему

«Програмне забезпечення з графічним інтерфейсом для створення й обробки QR-кодів»

Студента групи 602-ТН Бобохи Ярослава Юрійовича

Керівник роботи
кандидат технічних наук,
доцент Головко Г. В.

Завідувач кафедри
кандидат технічних наук,
доцент Двірна О.А.

Полтава – 2024

РЕФЕРАТ

Кваліфікаційна робота магістра: 78 с., 58 рисунків, 1 додаток, 22 джерела.

Об’єкт дослідження: програмне забезпечення з графічним інтерфейсом для створення та обробки QR-кодів.

Мета роботи: розробити програмне забезпечення для створення та розшифрування QR-кодів, яке забезпечує зручність використання, підтримку різних мов інтерфейсу та візуалізацію використання ресурсів системи.

Методи: аналіз існуючих рішень, структурне програмування, використання бібліотек Python для роботи з QR-кодами, обробка зображень та моніторинг системних ресурсів.

Ключові слова: QR-код, Python, qrcode, pyzbar, tkinter, обробка зображень, графічний інтерфейс.

ANNOTATION

Qualification work of master's degree: 78 p., 58 figures, 1 application, 22 sources.

Object of study: software with a graphical interface for creating and processing QR codes.

The goal of the work: to develop software for creating and decoding QR codes, which provides ease of use, support for various language interfaces and visualization of system resource usage.

Methods: analysis of existing solutions, structured programming, use of Python libraries for working with QR codes, image processing and monitoring of system resources.

Keywords: QR code, Python, qrcode, pyzbar, tkinter, image processing, graphical interface.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

- QR-код – Quick Response Code, двомірний штрих-код, який може зберігати різноманітну інформацію, таку як текст або URL-адреси;
- PIL – Python Imaging Library, бібліотека для роботи з графікою в Python;
- Tkinter – стандартна бібліотека Python для створення графічних інтерфейсів;
- PyZbar – бібліотека для декодування QR-кодів і інших штрих-кодів;
- PSUtil – бібліотека для отримання інформації про систему, таку як використання ЦП та пам’яті;
- Matplotlib – бібліотека для візуалізації даних у Python;
- GUI – Graphical User Interface, графічний інтерфейс користувача;
- CPU – Central Processing Unit, центральний процесор;
- RAM – Random Access Memory, оперативна пам’ять;
- URL – Uniform Resource Locator, уніфікований локатор ресурсу (інтернет-адреса);
- JSON – JavaScript Object Notation, формат обміну даними;
- API – Application Programming Interface, інтерфейс програмування додатків, який дозволяє різним програмам взаємодіяти між собою;
- SDK – Software Development Kit, набір інструментів для розробки програмного забезпечення;
- CSV – Comma-Separated Values, формат файлу, який використовується для зберігання табличних даних у текстовому вигляді;
- ASCII – American Standard Code for Information Interchange, стандартний код для представлення текстової інформації;

- UTF-8 – Unicode Transformation Format – 8-bit, кодування, що підтримує всі символи Unicode;
- PDF – Portable Document Format, формат файлу, який зберігає документацію в незалежному від програмного забезпечення вигляді;
- GUI – Graphical User Interface, графічний інтерфейс користувача;
- IDE – Integrated Development Environment, інтегроване середовище розробки, що надає засоби для програмування;
- URL – Uniform Resource Locator, стандартний формат для адреси ресурсів в Інтернеті;
- OCR – Optical Character Recognition, технологія для розпізнавання тексту в зображеннях.

ВСТУП

QR-коди є невід'ємною частиною цифрових комунікацій, надаючи можливість швидкого і зручного доступу до інформації. Їхня універсальність та простота використання роблять їх популярними у різних сферах діяльності, від маркетингу до мобільних платежів.

Однак для ефективного та безпечного використання QR-кодів необхідні спеціалізовані інструменти, які дозволяють користувачам легко створювати, налаштовувати та декодувати ці коди. Розробка такого програмного забезпечення з інтуїтивно зрозумілим графічним інтерфейсом є актуальною задачею, оскільки воно не лише спрощує роботу з QR-кодами, але й забезпечує захист від можливих загроз, пов'язаних з їх використанням.

Мета цієї дипломної роботи полягає у створенні програмного продукту, який об'єднає функції генерації та декодування QR-кодів у зручній і зрозумілій формі для користувачів різного рівня підготовки. Розробка цього програмного забезпечення включатиме в себе розв’язання технічних завдань, пов’язаних з генерацією та обробкою QR-кодів, а також створення сучасного інтерфейсу, який буде доступним для широкого кола користувачів.

Предметом дослідження є процеси створення, декодування та обробки QR-кодів, а також методи їх інтеграції в графічні інтерфейси програмного забезпечення. Це включає вивчення алгоритмів генерації QR-кодів, технологій декодування даних з QR-кодів та способів оптимізації інтерфейсу користувача для забезпечення зручності та безпеки при роботі з QR-кодами.

Об'єктом дослідження є програмне забезпечення для роботи з QR-кодами, яке включає в себе засоби для генерації та декодування QR-кодів, а також графічний інтерфейс, який забезпечує взаємодію користувача з програмним продуктом. Основна увага приділяється розробці та оцінці ефективності інтерфейсу користувача, який повинен бути інтуїтивно зрозумілим і доступним для широкого кола користувачів.

РОЗДІЛ І
ТЕОРЕТИЧНІ ОСНОВИ ТА ОСОБЛИВОСТІ ВИКОРИСТАННЯ QR-КОДІВ

1.1. Основи QR-кодів

Концепція штрих-коду виникла в США в 1940-х роках. Це важлива віха в дослідженні носія інформації у вигляді зображення людини в сучасному світі. Від найпершого використання в обліку товарів у супермаркетах до маркування та відстеження даних про різні предмети.

Однак форма носія інформації у вигляді одновимірного штрих-коду через розташування та комбінацію штрихів та порожніх місць має великі обмеження у змісті та кількості інформації, що зберігається, через обмеження ідентифікації довжини штрих-коду, тому він не може зберігати велику кількість інформації за один раз. Тому з розвитком технологій двовимірний код, який вміло використовує поняття бітових потоків «0» і «1», що формують внутрішню логічну основу комп'ютера в кодуванні, використовує кілька геометричних фігур, що відповідають двійковим, для представлення буквеної і числової інформації.

Тип штрих-коду, який може автоматично зчитуватися через обладнання для введення зображень або фотоелектричне скануюче обладнання для реалізації автоматичної обробки інформації, став основним носієм інформації про зображення в сучасну інформаційну епоху. QR-код збільшує інформаційну щільність штрих-коду за допомогою горизонтального та вертикального розташування комбінацій, так що щільність інформації може досягати десятків і сотень разів більше, ніж у одновимірного коду, щоб вмістити 1100-3800 байт та 500-1800 китайських ієрогліфів. Він також може кодувати текст, графіку, звук та іншу інформацію, не покладаючись на створення бази даних заздалегідь (див. Рис. 1.1-1.5) [1].

Рисунок 1.1 – Можливості QR-кодів

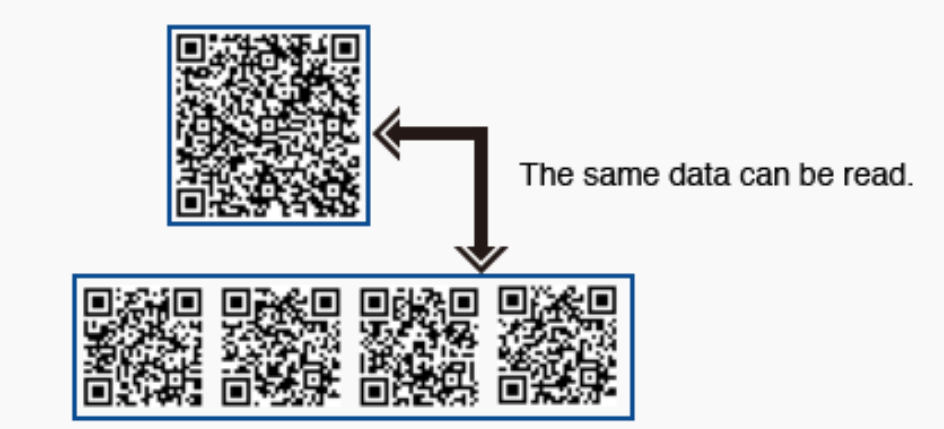

Рисунок 1.2 – Можливості QR-кодів

Рисунок 1.3 – Можливості QR-кодів

Рисунок 1.4 – Можливості QR-кодів

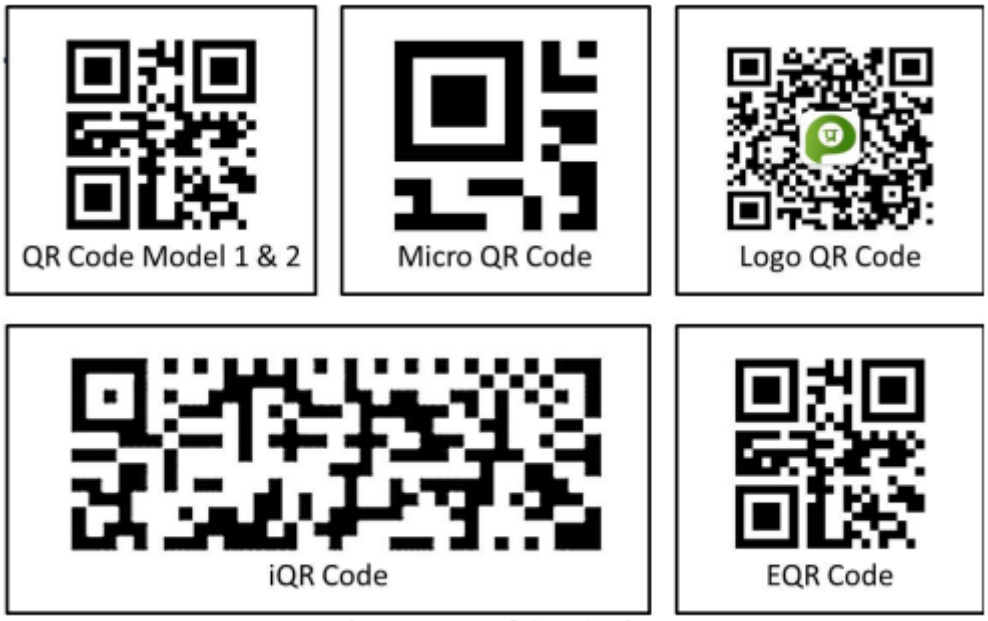

Рисунок 1.5 – Види QR-кодів

Люди можуть зберігати всю або більшу частину інформації про речі в QR-коді, який значною мірою реалізує зберігання та передрук інформації зі штрих-кодом. QR-код може бути відсутнім різними способами в різних ситуаціях, але завдяки функції виправлення помилок QR-коду, QR-код буде випадковим чином генерувати слова виправлення помилок при його генерації, тому навіть якщо є часткове пошкодження або відсутність, QR-код все одно може завершити операцію.

Кількість слів для виправлення помилок може становити від 2 до понад 89% від загальної кількості слів коду. Таким чином, інформація все ще може бути відновлена у випадку відсутності QR-коду, якщо вона становить 49%. Крім того, термін служби друкованого носія серйозно впливає на термін служби QR-коду.

Розташування матриці в графіці QR-коду не може бути змінено після виготовлення, тому QR-код на найважливіших документах або виробах має унікальність і захист від підробки, що значно покращує безпеку і захист від підробки при відтворенні даних. У той же час, в процесі підготовки QR-коду, шифрування може бути завершено шляхом зміни розташування матриці і комбінації певної частини, так що виробники і відвідувачі без ключів не

можуть правильно прочитати або підробити той же тип QR-коду, що може принципово захистити внутрішню інформаційну безпеку QR-коду і захистити права користування і конфіденційність користувачів і виробників.як QR-код є графічним файлом даних області безпосередньо піддається графічний код зберігання інформації, може бути надрукований на будь-яких поліграфічних або друкованих матеріалах, відігравати певну роль в автоматичній передачі інформації, носіння і боротьби з підробкою.

Крім того, виробництво двовимірного коду дуже просте, оскільки принтер може друкувати носій, який зможе виробляти двовимірний код, а його виробничі витрати значно нижчі, ніж інші способи [1].

1.2. Захист даних при використанні QR-кодів та обробка помилок

Відкритий доступ до технологічних рішень та комунікаційних мереж сприяє збільшенню обсягів даних, що генеруються та передаються щодня. Кожен власник пристрою з доступом до Інтернету може отримати доступ до десятків інформаційних ресурсів, які, в свою чергу, збирають і передають дані, які ми свідомо чи несвідомо розкриваємо, такі як місцезнаходження, включаючи місце роботи та/або проживання, раса, стать, вік, контактні дані, особисті інтереси, сімейний стан або релігійні переконання.

Ці дані збираються та зберігаються організаціями з метою подальшого використання для виявлення можливостей або загроз, підготовки стратегії прийняття рішень та адаптації до поточних або майбутніх тенденцій. Як уже згадувалося, ми всі оприлюднюємо величезні обсяги даних, і організаціям не потрібні всі дані, які ми оприлюднюємо, незалежно від того, чи є вони справжніми, чи ні. Збирачі даних повинні переконатися, що ця інформація відповідає їхнім вимогам до збору. Саме тут і з'являється валідація даних.

Валідація даних – це важлива практика забезпечення цілісності, точності та структурної надійності даних перед їхнім використанням у різних бізнес-операціях. Вона охоплює низку типів валідації, включаючи перевірку типу

даних, обмежень, структури, узгодженості та коду, кожна з яких призначена для перевірки відповідності даних необхідним критеріям для подальшої обробки. Водночас валідацію даних можна визначити як форму очищення даних, яка передбачає перевірку вихідних даних на точність і якість перед будь-якою обробкою, імпортом або використанням. Це визначення узгоджується з перспективою, представленою вище, де також наголошувалося на важливості забезпечення відповідності даних конкретним стандартам і цілям у межах обмежень, що існують у місці призначення. З огляду на величезні обсяги даних, які перевозяться щодня, захист цих даних є одним із пріоритетів організацій, які збирають ці дані. Це є зобов'язанням для відповідних організацій. Наприклад, згідно з правилами GDPR, компанії та організації, які збирають, зберігають та обробляють персональні дані, зобов'язані забезпечити гарантії захисту цих даних.

Таким чином, відповідні організації повинні розглянути можливість захисту даних у 2 відомих формах: дані в русі та збережені дані. З цією метою організації зазвичай використовують алгоритми контрольної суми даних, хеш-алгоритми та алгоритми шифрування. Кожна з цих категорій алгоритмів відіграє важливу роль у забезпеченні цілісності та конфіденційності даних.

Алгоритми контрольної суми використовуються для забезпечення цілісності даних. Ці алгоритми генерують значення фіксованого розміру, відомі як контрольні суми, які однозначно представляють вміст певного набору даних. Ці алгоритми спрямовані на виявлення помилок або змін у даних під час передачі або зберігання. Контрольна сума обчислюється у джерелі і надсилається або зберігається разом з даними під час їх передачі або зберігання. При отриманні або отриманні даних контрольна сума перераховується, і якщо вона відрізняється від початкової переданої або збереженої контрольної суми, це свідчить про можливу помилку або пошкодження даних.

Існують три типи алгоритмів контрольної суми. Перший алгоритм – CRC32 (Cyclic Redundancy Check 32) – алгоритм контрольної суми, заснований

на діленні поліномів, який хешує послідовності байтів до 32-бітових значень. Іншим алгоритмом, що використовується, є алгоритм Флетчера, який був розроблений Джоном Г. Флетчером (John G. Fletcher). Останній можливий алгоритм контрольної суми – Adler-32 – це алгоритм, написаний Марком Адлером шляхом модифікації контрольної суми Флетчера. У порівнянні з алгоритмом Флетчера, Adler32 в основному відомий своєю простотою і швидкістю, в той час як Fletcher32 вибирають для додатків, де ефективність і простота є пріоритетними над криптографічною стійкістю.

Алгоритми хешування є односторонніми або незворотними, тому текст не може бути розшифрований і декодований кимось іншим. Таким чином, хешування захищає дані у стані спокою, так що навіть якщо хтось отримає доступ до сервера, де вони зберігаються, елементи залишаються нечитабельними. У цьому дослідженні ми використовували SHA-1-HMAC, алгоритм хешування з ключем, заснований на хеш-функції SHA1, який використовується як код автентифікації повідомлень на основі хешування (HMAC). Процес HMAC змішує секретний ключ з даними повідомлення, хешує результат за допомогою хеш-функції, знову змішує значення хешу з секретним ключем, а потім застосовує хеш-функцію вдруге.

З іншої сторони, шифрування даних – це захід безпеки, який передбачає перетворення читабельної інформації, яка називається відкритим текстом, у нечитабельний формат, який називається зашифрованим текстом, за допомогою алгоритмів і криптографічних ключів. Цей процес слугує для захисту конфіденційних даних від несанкціонованого доступу, зберігаючи при цьому конфіденційність і приватність.

У шифруванні математичний метод використовує криптографічний ключ для зміни вихідних даних, роблячи їх нечитабельними без відповідного ключа для розшифрування. Цей криптографічний підхід має вирішальне значення для захисту інформації під час передачі або зберігання, запобігаючи загрозам з боку кіберзлочинців, неавторизованих користувачів та інших шкідливих суб'єктів. Шифрування відіграє важливу роль у захисті різних

цифрових комунікацій і транзакцій, таких як онлайн-банкінг, транзакції електронної комерції та передача конфіденційної інформації. Воно слугує фундаментальним компонентом стратегій кібербезпеки, запобігаючи інтерпретації та використанню конфіденційних даних несанкціонованими сторонами.

Існує два типи алгоритмів шифрування, а саме: симетричні алгоритми шифрування, в яких один ключ використовується як для шифрування, так і для розшифрування даних, та асиметричні алгоритми шифрування, які передбачають використання двох ключів: відкритого ключа, який використовується для шифрування даних, та закритого ключа, який використовується для розшифрування даних.

В якості алгоритмів шифрування можна розглянути два асиметричних алгоритми: RSA та ECDSA. RSA, або Rivest-Shamir-Adleman – це алгоритм шифрування, який забезпечує універсальний підхід до захисту даних за допомогою асиметричної криптографії, пропонуючи два різних методи. Один з методів передбачає шифрування конфіденційної інформації за допомогою відкритого ключа одержувача, гарантуючи, що тільки власник відповідного закритого ключа може розшифрувати дані. Цей метод ідеально підходить для безпечної передачі даних через мережі, оскільки відправник використовує відкритий ключ одержувача для шифрування. Інший метод передбачає шифрування повідомлення за допомогою закритого ключа відправника і надсилання одержувачу як зашифрованих даних, так і відкритого ключа відправника. Тут одержувач може розшифрувати дані за допомогою відкритого ключа відправника, перевіряючи особу відправника.

ECDSA (Elliptic Curve Digital Signature Algorithm) – це асиметричний криптографічний примітив, який вирізняється ефективністю цифрового підпису порівняно з іншими алгоритмами, що досягається за рахунок використання менших ключів при збереженні високого рівня безпеки. ECDSA генерує сертифікати – електронні документи для автентифікації власника сертифіката, що містять інформацію про ключ, дані власника та підпис

емітента. В основі безпеки ECDSA лежить аналіз еліптичних кривих, що робить його стійким до сучасних методів злому шифрування. Незважаючи на те, що ECDSA було стандартизовано у 2005 році, новіші протоколи надають перевагу ECDSA через його відносну новизну, що скорочує час, необхідний хакерам для його зламу.

Однак RSA залишається широко використовуваним, насамперед через його довшу присутність з моменту стандартизації в 1995 році, хоча зловмисники мали більше часу, щоб спробувати його зламати. Хоча ECDSA пропонує підвищений рівень безпеки і є кращим у певних контекстах, його недоліком є складність у впровадженні порівняно з більш простим налаштуванням RSA, що робить належне впровадження критично важливим для уникнення вразливостей.

Щоб оцінити поведінку та ефективність кожного з описаних вище підходів, було знайдено згенеровані зразки кодів з використанням підходів та методів.

У всіх випадках використовувався середній рівень корекції помилок. Для кожного коду було записано декілька параметрів, включаючи фізичний простір, зайнятий кожним кодом (визначається версією QR, необхідною для розміщення кожного з рядків, що кодуються), а також біти безпеки, що забезпечуються кожним з алгоритмів перевірки та іншими специфікаціями. Використовуючи реалізацію алгоритму контрольної суми та криптографічні примітиви, описані вище, були згенеровані коди відповідно до кожної зі схем перевірки, представлених вище. Результати можна знайти на Рис. 1.6-1.10.

Рисунок 1.6 – Мінімум даних, що кодуються для перевірки контрольної суми

Рисунок 1.7 – Мінімальна кількість даних, закодованих для перевірки HMAC

Рисунок 1.8 – Повні дані закодовані для перевірки HMAC

Рисунок 1.9 – Повні дані закодовані для перевірки RSA

Рисунок 1.10 – Повні дані закодовані для перевірки ECDSA

Як можна побачити з наведених вище рисунків, розмір QR-кодів широко варіюється в залежності від схеми валідації, яка буде використовуватися. Відповідно до цього і з огляду на зростаючу тенденцію до використання QR-кодів на друкованих документах, підхід, який краще використовувати, в значній мірі визначається наявним фізичним простором. Таким чином, порівняння простору, необхідного для кожного підходу для кодування даних, представлено за допомогою Таблиці 1.1 для мінімального друкованого розміру стандартне співвідношення 12 модулів/см [2].

Таблиця 1.1 – Порівняння методів кодування

Метод	**Розмір даних (Біт)**	**Відносний розмір коду (%)**	**Версія QR-коду**	**Кількість модулів**	**Мінімальний розмір друку (см²)**
1	25	21,6	2	25 x 25	2.08 x 2.08
2	48	60.0	4	33 x 33	2.75 x 2.75
3	428	10.0	16	81 x 81	6.75 x 6.75
4	727	47.2	22	105 x 105	8.75 x 8.75
5	561	31.4	19	93 x 93	7.75 x 7.75

Частиною стійкості QR-кодів у фізичному середовищі є їх здатність витримувати «пошкодження» і продовжувати функціонувати навіть тоді, коли частина зображення QR-коду затемнена, зіпсована або видалена.

Це досягається за допомогою алгоритму корекції помилок Ріда-Соломона – серйозної алгебри, яка виконується у фоновому режимі під час створення QR-коду. Оригінальні дані в QR-коді перетворюються в поліном, визначається кількість унікальних точок, необхідних для однозначного визначення цього полінома, і цей набір точок додається назад в QR-код, щоб потім він також містив оригінальні дані, виражені у вигляді полінома.

Існує 4 рівні виправлення помилок для QR-кодів, кожен з яких додає різну кількість «резервних» даних в залежності від того, наскільки сильно QR-код може бути пошкоджений (див. Рис. 1.11) в передбачуваному середовищі, і, отже, наскільки сильно може знадобитися виправлення помилок:

- Рівень L – до 7% пошкоджень;
- Рівень M – до 15% пошкоджень;
- Рівень Q – до 25% пошкоджень;
- Рівень H – до 30% пошкоджень.

Рисунок 1.11 – Рівень пошкодження QR-коду

Фундаментальною частиною роботи QR-кодів є те, що чим більше даних ви вносите в них, тим більше рядків і стовпців модулів буде введено в QR-код, щоб компенсувати збільшене навантаження даних. Оскільки рівень корекції помилок зростає, це означає, що також збільшується кількість рядків і стовпців модулів, необхідних для зберігання оригінальних даних, а також зростає кількість резервних кодових слів. Це показано на діаграмі нижче – QR-код стає більш щільним зі збільшенням рівня корекції помилок від рівня L до рівня H, навіть якщо QR-коди містять точно таку ж URL-адресу веб-сайту.

Досить зручно, що в нижньому лівому кутку кожного QR-коду є також 2 модулі, які показують, який рівень виправлення помилок використовується в цьому QR-коді (див. Рис. 1.12) [3].

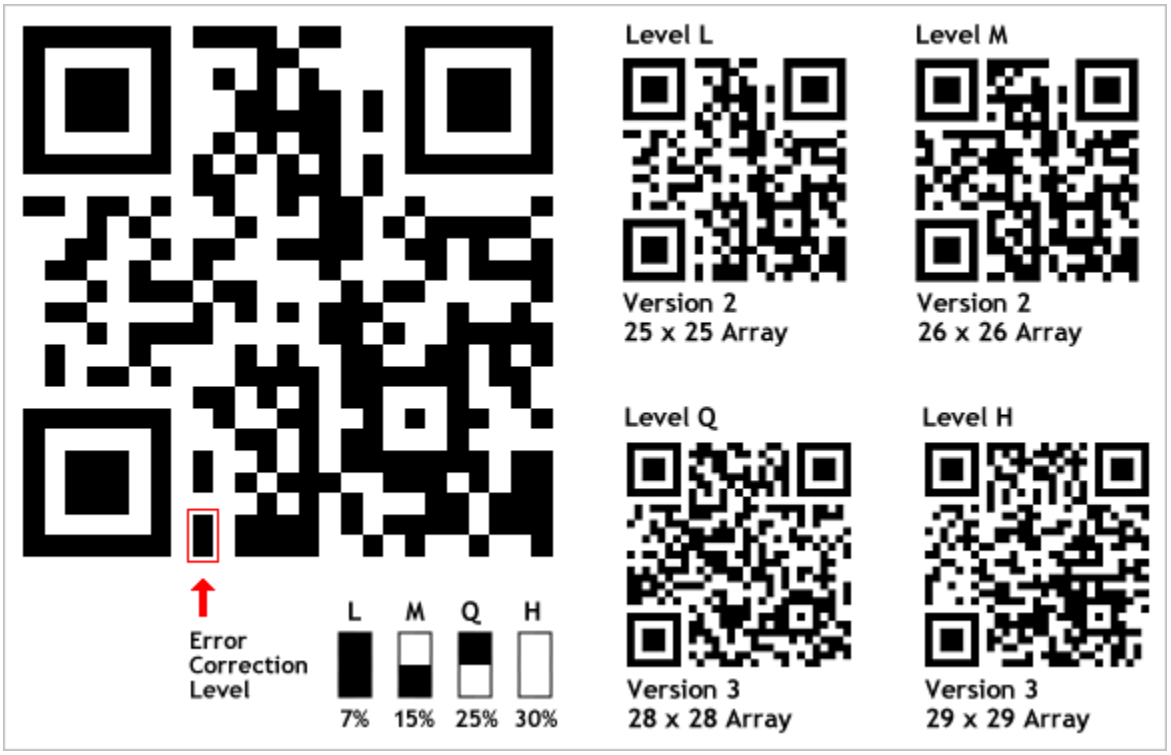

Рисунок 1.12 – Демонстрація рівнів корекції помилок QR-кодів

Таким чином, чим нижчий рівень корекції помилок, тим менш щільним є зображення QR-коду, що покращує мінімальний розмір друку. Чим вищий рівень виправлення помилок, тим більше пошкоджень він може витримати, перш ніж стане нечитабельним.

Рівень L або M є найкращим компромісом між щільністю та міцністю для загального маркетингового використання. Рівні Q і H зазвичай рекомендуються для промислового середовища, де збереження чистоти і неушкодженості QR-коду є складним завданням.

Це також одна з причин, чому QR-код, що містить однакові дані, буде виглядати по-різному в залежності від того, який генератор QR-кодів ви використовуєте – це залежить від рівня корекції помилок, який використовується на конкретному веб-сайті. Незважаючи на те, що існує єдиний стандарт ISO для QR-кодів, в рамках стандарту ISO є змінні (рівень корекції помилок є однією з них), які призводять до різного вигляду зображення QR-коду в залежності від того, як конкретний веб-сайт для створення QR-кодів встановлює ці змінні.

Це не означає, що якийсь конкретний генератор QR-кодів є більш або менш сумісним зі стандартами, ніж будь-який інший, це просто означає, що люди, які стоять за різними генераторами, зробили різний вибір при встановленні основних технічних характеристик і параметрів для QR-кодів, які створюються на їх веб-сайтах [3].

1.3. Вміст, структура та шаблони QR-кодів

QR-коди стали поширеним явищем у споживчій рекламі. Зазвичай смартфон використовується як сканер QR-коду, який відображає код і перетворює його в якусь корисну форму (наприклад, стандартну URL-адресу веб-сайту, тим самим позбавляючи користувача необхідності вводити її у веб-браузер).

QR-код став центром уваги рекламної стратегії, оскільки він забезпечує швидший доступ до веб-сайту бренду, ніж ручне введення URL-адреси. Окрім простої зручності для споживача, важливість цієї можливості полягає в тому, що вона збільшує коефіцієнт конверсії: ймовірність того, що контакт з рекламою перетвориться на купівлю. Вона спонукає зацікавлених

потенційних клієнтів рухатися далі вниз по воронці конверсії без особливих зусиль і затримок, негайно приводячи глядача на сайт рекламодавця, тоді як більш тривала і цілеспрямована пропозиція може втратити інтерес глядача.

Хоча спочатку QR-коди використовувалися для відстеження деталей у виробництві транспортних засобів, вони мають набагато ширший спектр застосування. Вони включають комерційне відстеження, контроль складських запасів, розваги та продаж квитків на транспорт, маркетинг продуктів та лояльності, а також маркування товарів у магазині. Приклади маркетингу включають в себе, коли знижку компанії та відсоток знижки можна отримати за допомогою декодера QR-коду, який є мобільним додатком, або зберігання інформації про компанію, такої як адреса та пов'язана з нею інформація, поряд з буквено-цифровими текстовими даними, як це можна побачити на жовтих сторінках телефонного довідника.

Вони також можуть використовуватися для зберігання особистої інформації організацій. Прикладом цього є Національне бюро розслідувань Філіппін (НБР), де допуск до роботи в НБР тепер супроводжується QR-кодом. Багато з цих додатків орієнтовані на користувачів мобільних телефонів (за допомогою мобільних тегів). Після сканування QR-коду користувачі можуть отримати текст, додати контакт у вигляді візитки на свій пристрій, відкрити URL-адресу, написати електронне або текстове повідомлення. Вони можуть генерувати і роздруковувати власні QR-коди, щоб інші могли їх сканувати і використовувати, відвідавши один з декількох платних або безкоштовних сайтів або додатків для генерації QR-кодів. Google мав API, який зараз застарілий, для генерації QR-кодів, а додатки для сканування QR-кодів можна знайти майже на всіх пристроях смартфонів [4].

QR-коди, що зберігають адреси та URL-адреси, можуть з'являтися в журналах, на вивісках, в автобусах, на візитних картках або практично на будь-якому об'єкті, про який користувачі можуть захотіти отримати інформацію. Користувачі телефонів з камерою, оснащених відповідною програмою для зчитування, можуть відсканувати зображення QR-коду, щоб

відобразити текст і контактну інформацію, підключитися до бездротової мережі або відкрити веб-сторінку в браузері телефону. Цей акт зв'язування з об'єктами фізичного світу називається жорстким зв'язуванням або гіперзв'язуванням об'єктів. QR-коди також можуть бути прив'язані до місця, щоб відстежити, де код був відсканований. Або програма, яка сканує QR-код, отримує геоінформацію за допомогою GPS і тріангуляції веж стільникового зв'язку (aGPS), або URL-адреса, закодована в самому QR-коді, пов'язана з місцезнаходженням.

У 2008 році японський каменяр оголосив про плани гравірувати QR-коди на надгробках, що дозволить відвідувачам переглядати інформацію про померлого, а членам сім'ї відстежувати відвідування. Психолог Річард Вайзман був одним з перших авторів, який включив QR-коди в книгу «Паранормальне явище»: Чому ми бачимо те, чого немає (2011). Microsoft Office та LibreOffice мають функцію вставки QR-коду в документи.

QR-коди були включені у валюту. У червні 2011 року Королівський монетний двір Нідерландів (Koninklijke Nederlandse Munt) випустив першу в світі офіційну монету з QR-кодом, щоб відсвяткувати сторіччя своєї нинішньої будівлі і приміщень. Монету можна відсканувати смартфоном і отримати посилання на спеціальний веб-сайт з інформацією про історичну подію та дизайн монети. У 2014 році Центральний банк Нігерії випустив банкноту номіналом 100 найр на честь свого сторіччя, яка стала першою банкнотою з QR-кодом у своєму дизайні. При скануванні мобільним пристроєм з доступом до Інтернету код переходить на веб-сайт, який розповідає столітню історію Нігерії.

У 2017 році Банк Гани випустив банкноту номіналом 5 седісів на честь 60-річчя центрального банку Гани. Банкнота містить QR-код, який при скануванні мобільним пристроєм з доступом до Інтернету переходить на офіційний веб-сайт Банку Гани.

Функціонал кредитних карток знаходиться на стадії розробки. У вересні 2016 року Резервний банк Індії (RBI) запустив однойменний BharatQR,

загальний QR-код, спільно розроблений усіма чотирма найбільшими картковими платіжними компаніями – Національною платіжною корпорацією Індії, яка обслуговує картки RuPay, а також Mastercard, Visa та American Express.

Прозорість бренду передбачає відкритість і чесність у стосунках із клієнтами та потенційними споживачами. Сучасні споживачі все більше прагнуть отримувати інформацію про походження та спосіб виробництва матеріалів, що використовуються компаніями. Як відомо, чесність є найкращою стратегією.

Відео QR-код надає можливість реалізувати цю маркетингову стратегію інноваційним і ефективним способом (див. Рис. 1.13). Можливість ділитися плейлистом відео дозволяє компаніям залучати споживачів до коротких документальних відеоматеріалів, що розповідають про походження матеріалів, робочий процес та шлях товарів від джерела до полиць. Для компаній, які прагнуть оновити свій бренд, це є чудовою можливістю продемонструвати основи нового іміджу як постійним, так і новим клієнтам. Такі відео можуть бути виконані в гумористичному, сатиричному, освітньому або суто інформаційному стилі. Це дозволяє підвищити рівень взаємодії з цільовою аудиторією.

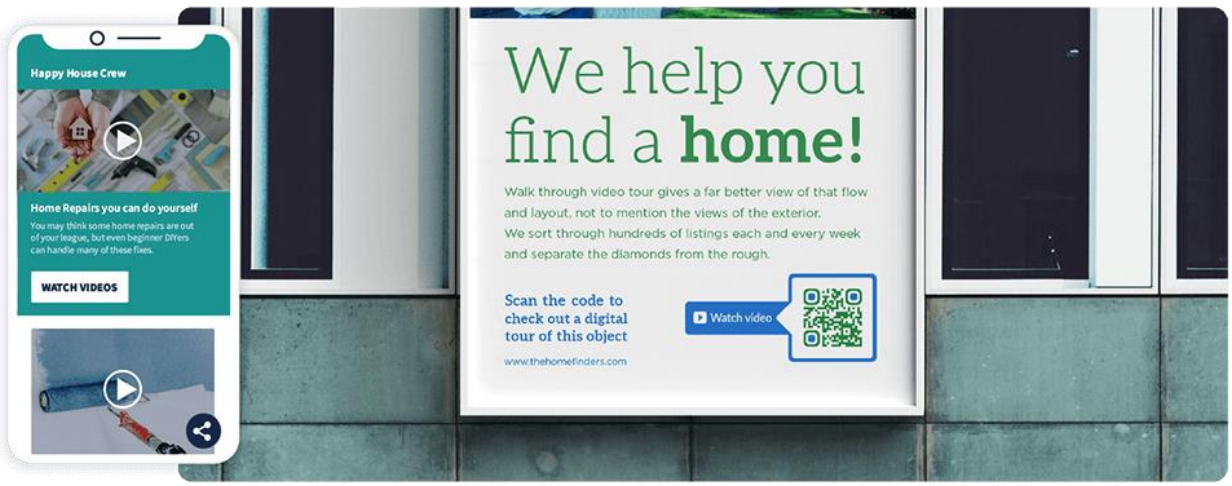

Рисунок 1.13 – QR-код із посиланням на відео

Оскільки 45% світового населення користуються смартфонами, важливо залучати їх через платформи, на яких вони проводять значну частину свого часу. Форми зворотного зв'язку з QR-кодом адаптуються до будь-яких мобільних пристроїв і дозволяють редагувати категорії навіть після того, як код був згенерований та роздрукований (див. Рис. 1.14). Це забезпечує три основні переваги: можливість редагування, відстеження та зміни. Таким чином, форми можна адаптувати відповідно до потреб у будь-який час і в будь-якому місці [4].

Рисунок 1.14 – QR-код із посиланням на розділ відгуків

Залучення клієнтів може відбуватися як онлайн, так і офлайн, але платформи соціальних мереж є критично важливими інструментами в маркетинговій діяльності. Використання QR-кодів дозволяє поєднати цифрові та традиційні кампанії для забезпечення зв'язку з клієнтами на всіх доступних платформах.

Facebook, який наразі нараховує 2,5 мільярда користувачів щомісяця, з яких 1,66 мільярда є активними користувачами щодня, надає можливість безпосереднього доступу до широкої аудиторії (див. Рис. 1.15). Бізнес-сторінка на цій платформі також дозволяє використовувати її як відправну точку для зв'язку з підписниками поза межами Facebook через конкурси, розіграші та інформаційні розсилки.

Рисунок 1.15 – QR-код із посиланням на Facebook

QR-код Facebook може допомогти залучити цільову аудиторію, яка активно взаємодіятиме з вашими публікаціями та потенційно стане довгостроковими клієнтами. Цей код з'єднує сканер безпосередньо з профілем у Facebook, надаючи коротку інформацію про ваш бізнес на зручній для мобільних пристроїв сторінці. Він також містить зображення та кольори бренду, що налаштовуються, і інтегровану кнопку «мені подобається». Окрім цього, під інформаційною сторінкою розміщено додаткове посилання на ваш веб-сайт або цільову сторінку, що сприяє залученню клієнтів на кількох платформах одночасно [16].

Twitter забезпечує платформу для початку розмов, отримання зворотного зв'язку від клієнтів та можливість стати популярним у реальному часі (див. Рис. 1.16). Незважаючи на критику, пов'язану з політикою США, швидкі дії з видалення близько 70 мільйонів фейкових акаунтів відновили довіру до Twitter. Наразі платформа має 330 мільйонів активних користувачів щомісяця, з яких 145 мільйонів є активними щодня, і 67% усіх B2B-бізнесів використовують Twitter у своєму цифровому маркетингу. До того ж, 40% користувачів Twitter повідомили, що придбали товар після того, як побачили його у своїй стрічці.

Рисунок 1.16 – QR-код із посиланням на Twitter

QR-код для Twitter може виконати одну з двох функцій у вашій маркетинговій кампанії: залучити аудиторію до запуску нового продукту шляхом переходу на твіт із заздалегідь заповненим повідомленням або збільшити кількість підписників, миттєво приєднуючи їх до вашого профілю. Це дозволяє вашій аудиторії легко підтримувати ваш бренд через миттєві твіти без необхідності вводити текст та шукати ваш профіль.

Якщо у вас є кілька акаунтів на різних платформах, ви можете об'єднати їх усі за допомогою одного QR-коду для соціальних мереж. Це забезпечує зручний доступ до ваших профілів на Facebook, Twitter, Instagram, LinkedIn, YouTube, Xing та інших платформах, що значно спрощує взаємодію з аудиторією [5].

Для власників бізнесу або медичних працівників, які прагнуть надати клієнтам або пацієнтам миттєвий доступ до значної кількості письмових матеріалів, QR-код PDF є оптимальним рішенням (див. Рис. 1.17). Він дозволяє зберігати та надавати доступ до додаткових досліджень, реєстраційних листів або посібників користувача в зручному форматі, що значно спрощує обмін інформацією [20].

Рисунок 1.17 – QR-код із посиланням на PDF-файл

Завдяки електронному QR-коду ви можете забезпечити миттєвий зв'язок між клієнтами та вашими послугами, що сприяє полегшенню процесу бронювання, залишення відгуків або подання запитів. Це ефективний спосіб інтеграції офлайн-послуг із зручними онлайн-платформами.

SMS QR-код пропонує ще один крок уперед, надаючи сканеру попередньо заповнений текст для відправлення на цільовий номер телефону (див. Рис. 1.18). Ця технологія дозволяє швидко встановити зв'язок із потенційними клієнтами, забезпечуючи миттєвий доступ до ваших акцій, конкурсів або розіграшів, без необхідності підключення до Інтернету [5].

Рисунок 1.18 – QR-код із посиланням на SMS-чат

Підсумовуючи, QR-код може містити в собі майже будь-яку інформацію. Це продемонстровано за допомогою Таблиці 1.2 [6].

Таблиця 1.2 – Види інформаціїї, що містить в собі QR-код

Тип QR-коду	**Призначення**
vCard Plus Code	Підключіть користувача до цифрової візитної картки
Код Facebook	Посилання на Facebook, щоб отримати більше підписників
Код програми	Встановіть програму з кількох магазинів програм за допомогою одного QR-коду
Код PDF	Додайте PDF-файл, який можна завантажити, до будь-якого друкованого матеріалу
Кодекс соціальних мереж	Направляйте користувачів на кілька облікових записів соціальних мереж одночасно
Код купона	Покажіть купон на знижку на продукт або послугу
Підприємницький кодекс	Рекламуйте свій бізнес на сторінці, яку легко читати
Код відео	Поділіться одним або кількома відео
Рейтинговий код	Легко збирайте оцінки
Код зворотного зв'язку	Поділіться опитуваннями з відгуками
Код події	Плануйте та збільшуйте відвідуваність події
Код динамічної URL-адреси	Посилання на будь-який веб-сайт, яке можна змінити в будь-який час
Код галереї зображень	Переглядайте кілька зображень в одному місці
MP3 код	Поділіться аудіофайлом у форматі MP3
Код електронної пошти	Миттєво збирайте та діліться електронними адресами
Код vCard	Поділіться контактними даними, які будуть додані за лічені секунди
SMS код	Надсилайте автоматичні повідомлення безпосередньо
Текстовий код	Надсилайте текст будь-якою мовою та діліться ним
Статичний URL	Стандартний спосіб підключення QR-коду до будь-якої веб-сторінки

Кожен символ QR-коду повинен бути побудований з квадратних модулів, розташованих у правильному квадратному масиві, і повинен складатися з функціональних шаблонів та області кодування. І весь символ повинен бути оточений з усіх чотирьох сторін межею тихої зони. Функціональні шаблони – це фігури, які повинні бути розміщені в певних областях QR-коду, щоб гарантувати, що сканери QR-кодів можуть правильно ідентифікувати та орієнтувати код для декодування.

Існує 4 типи функціональних шаблонів: шаблон пошуку, роздільник, часові шаблони та шаблони вирівнювання. Область кодування містить дані, які представляють інформацію про версію, інформацію про формат, дані та кодові слова для виправлення помилок. Рис. 1.19 ілюструє структуру символу QR-коду.

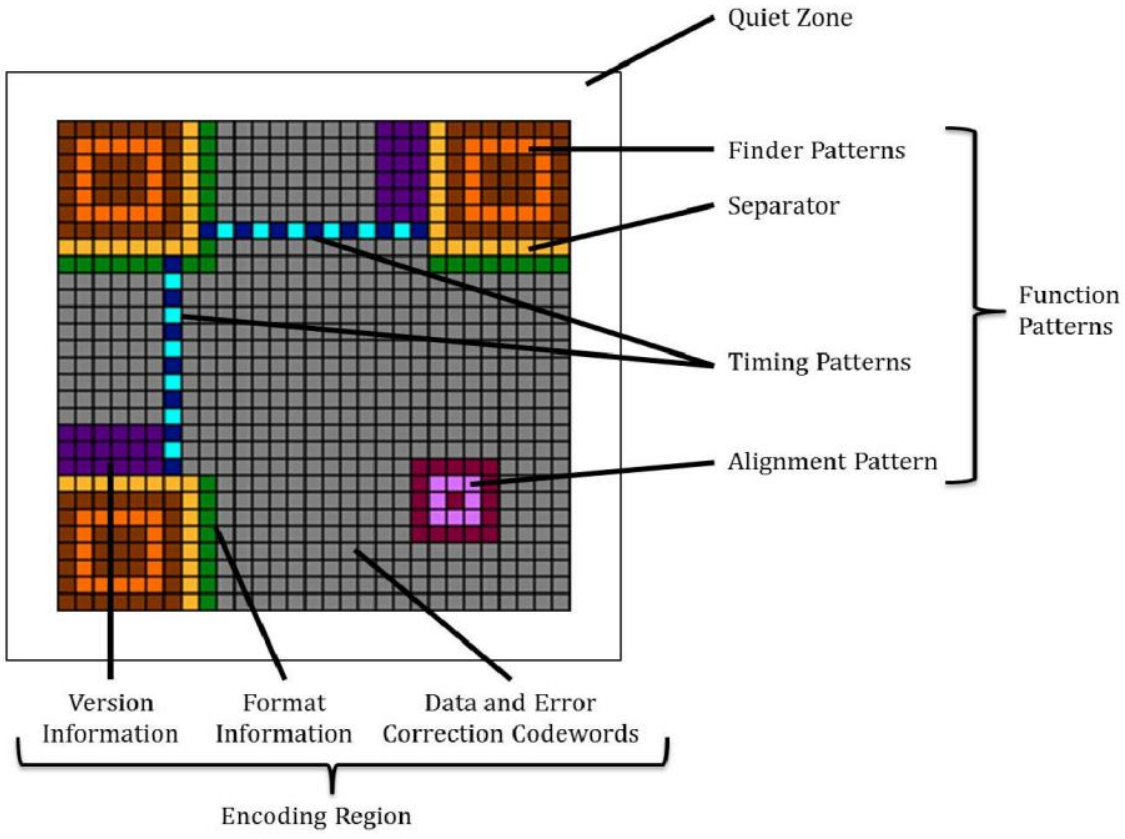

Рисунок 1.19 – Структура QR-коду

Шаблон пошуку: Візерунки пошуку – це спеціальні візерунки визначення положення, розташовані в трьох кутах (верхній лівий, правому

верхньому та лівому нижньому) кожного символу. Він складається із зовнішнього темного квадрата розміром 7 × 7 модулів, внутрішнього світлого квадрата розміром 5 × 5 модулів і суцільного темного квадрата в центрі розміром 3 × 3 модулі. Співвідношення ширини модулів у кожному шаблоні виявлення положення становить 1:1:3:1:1, як показано на Рис. 1.20. Шаблон пошуку розроблено таким чином, щоб він був шаблоном, який малоймовірно з'явиться в інших секціях, щоб сканери QR-кодів могли шукати це співвідношення світлих і темних модулів, щоб виявити шукач шаблони і правильно зорієнтувати QR-код для декодування [7].

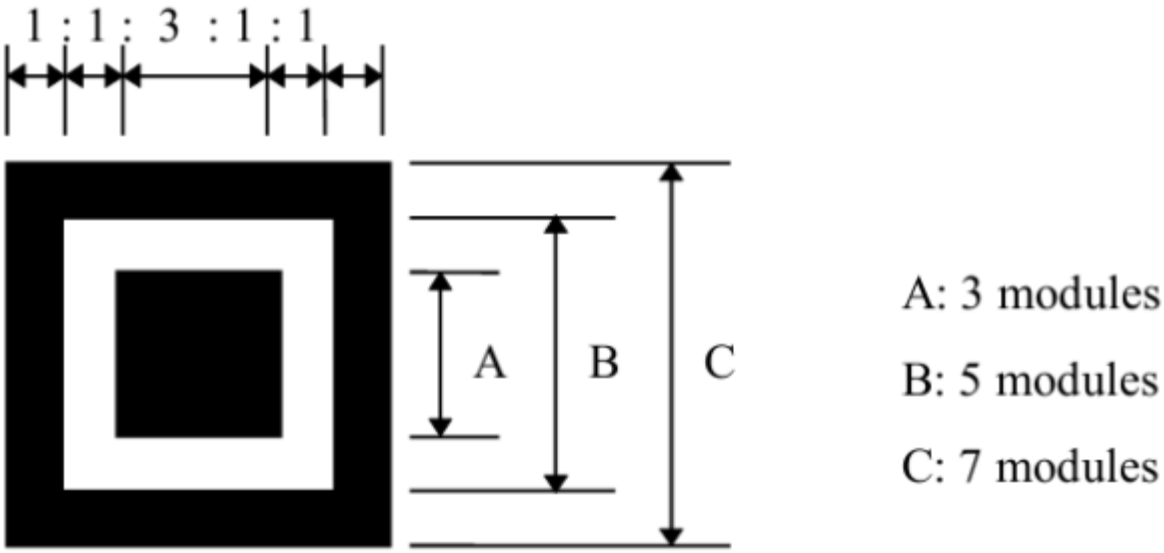

Рисунок 1.20 – Шаблон QR-коду

Розділювачі – це одномодульні широкі області пробілів між кожним шаблоном пошуку та областю кодування. регіоном кодування.

Шаблони синхронізації: існує 2 шаблони синхронізації, а саме горизонтальна і вертикальна часові діаграми. Вони складаються з чергування темних і світлих модулів, що чергуються. Горизонтальний шаблон хронометражу розміщується в 6-му рядку QR-коду між роздільниками. Вертикальний шаблон вертикальна хронометражна схема знаходиться в 6-му стовпчику QR-коду між роздільниками. Ці шаблони допомагають визначити щільність символів, модуль координат і області інформації про версію.

Шаблон вирівнювання будується з 5 × 5 темних модулів, 3 × 3 світлих модулів і одного темного модуля в центрі. QR-коди версії 2 і вище, повинні мати шаблони вирівнювання, а кількість шаблонів вирівнювання залежить від версії символу [17].

Область кодування містить інформацію про формат інформацію про формат, інформацію про версію, дані та помилки коди виправлення помилок.

Для інформації про формат слід зарезервувати одномодульний масив має бути зарезервовано біля лівого верхнього кута, правого верхнього кута, лівого верхнього, правого верхнього, лівого нижнього шаблону шукача та інформації про версію, а також блок розміром 6 × 3 над лівим нижнім шаблоном та блок розміром 3 × 6 ліворуч від верхнього правого зарезервовано.

Тиха зона – це область шириною у 4 модулі, яка не містить жодних даних, і використовується для того, щоб гарантувати, що навколишній текст або маркування не повинні вводити в оману щодо даних QR-коду [7, 19].

Кожен QR-код має три шаблони пошуку, які складаються з зовнішнього темного квадрата, середнього білого квадрата і внутрішнього темного квадрата. Зовнішній квадрат має розмір 7 модулів завширшки і 7 модулів заввишки, тоді як середній і внутрішній квадрати мають розміри 5 на 5 і 3 на 3 відповідно (див. Рис. 1.21) [16].

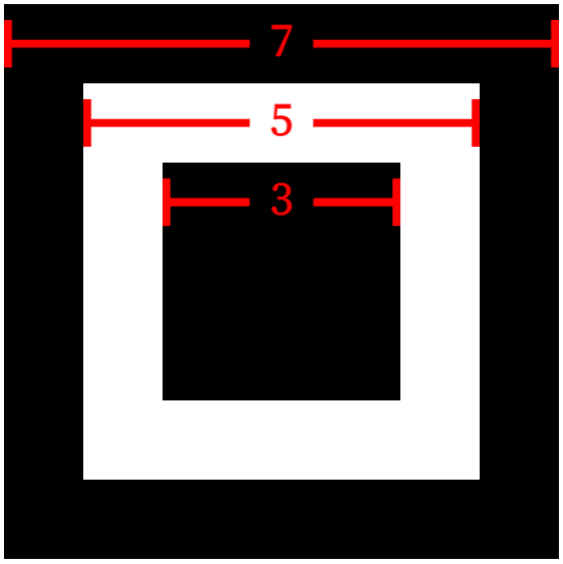

Рисунок 1.21 – Шаблон пошуку з відповідною шириною для кожного квадратного компонента

Шаблони пошуку завжди розміщуються у верхньому лівому, нижньому лівому та верхньому правому кутах QR-коду. У нашому прикладі HELLO WORLD шаблони пошуку будуть розміщені таким чином, як показано на Рис. 1.22.

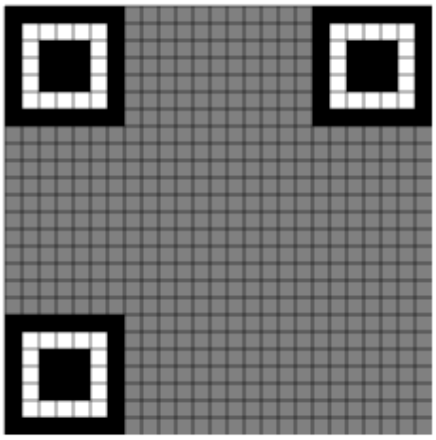

Рисунок 1.22 – Розміщення шаблону пошуку QR-коду для прикладу HELLO WORLD

На зовнішній стороні шаблонів пошуку є лінії світлових модулів, які називаються розділювальними шаблонами. Мета цих візерунків – відокремити візерунки пошуку від решти QR-коду, звідси і назва. Розділювачі мають ширину в 1 модуль і розміщені на сторонах зон пошуку, які звернені до внутрішньої сторони QR-коду, як показано на Рис. 1.23.

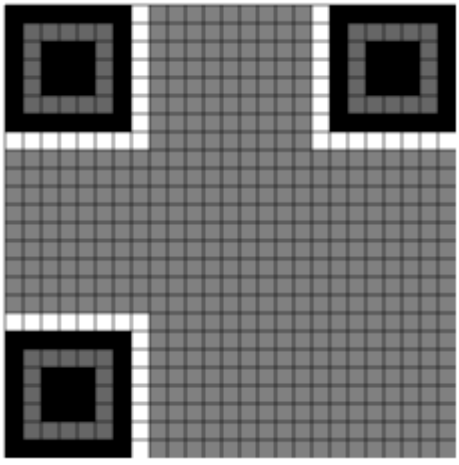

Рисунок 1.23 – Розміщення візерунків-розділювачів для прикладу HELLO WORLD.

Для версій QR-кодів, більших за 1, потрібно мати шаблони вирівнювання. Шаблон вирівнювання схожий на шаблон пошуку. Він складається з темного квадрата 5 на 5 модулів, внутрішнього світлого квадрата 3 на 3 модулі і одного темного модуля в центрі, як показано на Рис. 1.24 [16].

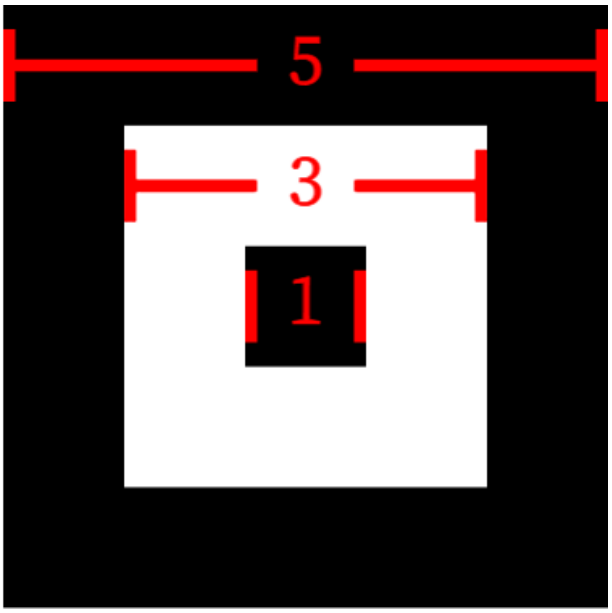

Рисунок 1.24 – Шаблони вирівнювання

Таблиця розташування шаблонів вирівнювання містить розташування вирівнювань, які залежать від версії QR-коду. Значення в таблиці представляють значення рядків і стовпців центру кожного модуля. Кожне вирівнювання має координату рядка і стовпця, яка є одним із значень. Наприклад, версія QR-коду для нашого прикладу має значення 6 і 18. Це відповідає центральним координатам (6, 6), (6, 18), (18, 6) і (18, 18).

Однак, жодна деталь вирівнювання не розміщена в місці, яке перекривається з деталями шукача і розділювача. Це означає, що в точках (6, 6), (6, 18) і (18, 6) не буде розміщено жодного вирівнювача. Це продемонстровано за допомогою Рис. 1.25.

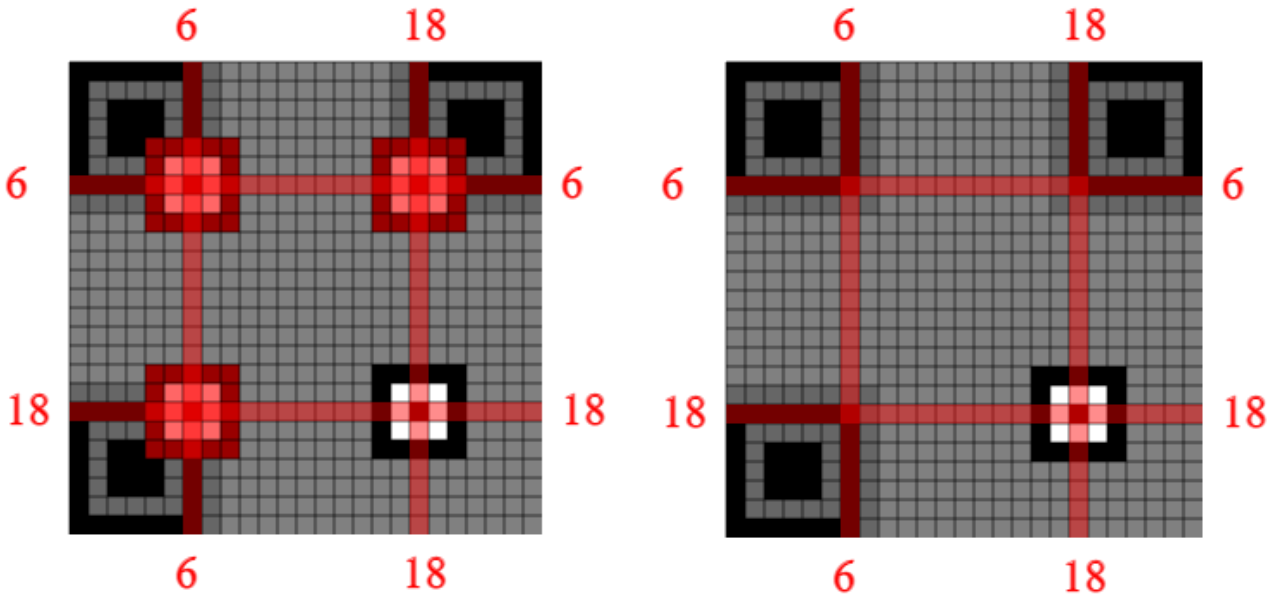

Рисунок 1.25 – Неправильний спосіб розміщення модулів вирівнювання (ліворуч) проти правильного способу (праворуч)

Кожен QR-код має один горизонтальний і один вертикальний часовий шаблон. Ці часові шаблони складаються з чергування темних і світлих модулів, які завжди починаються і закінчуються темним модулем. Вони розміщені між роздільниками в сьомому рядку і сьомому стовпчику QR-коду (див. Рис. 1.26).

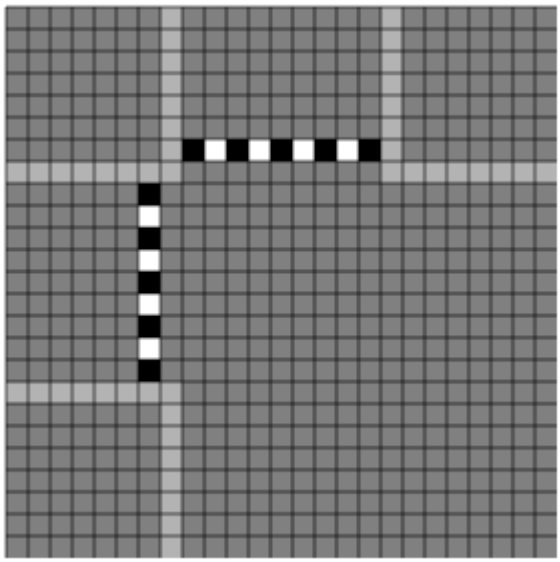

Рисунок 1.26 – Приклад розміщення шаблонів синхронізації. Розділювачі заштриховані для довідки

Кожен QR-код має один темний модуль праворуч від нижнього лівого роздільника (див. Рис. 1.27). Зокрема, модуль знаходиться в колонці 8 і рядку 4*(v-1)+13, де v – версія QR-коду. Для нашого прикладу це рядок номер 17, оскільки ми використовуємо QR-код версії 2 [21].

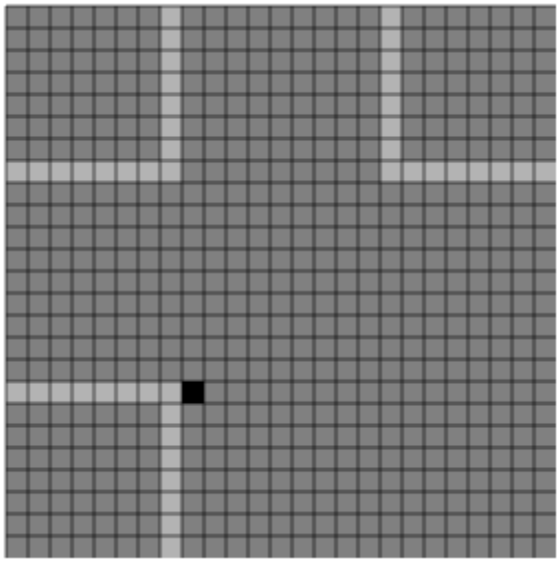

Рисунок 1.27 – Темне розміщення модулів для прикладу HELLO WORLD. Розділювачі заштриховані для довідки

QR-коди з версією 7 і вище містять дві області, які зберігають інформацію про версію. Одна область являє собою блок розміром 6 на 3 модулі над лівим нижнім роздільником (див. Рис. 1.28).

Інша область – це блок розміром 3 на 6 модулів зліва від верхнього правого роздільника. 18-бітний рядок версії складається з 6 бітів для номера версії і 12 бітів корекції помилок [21].

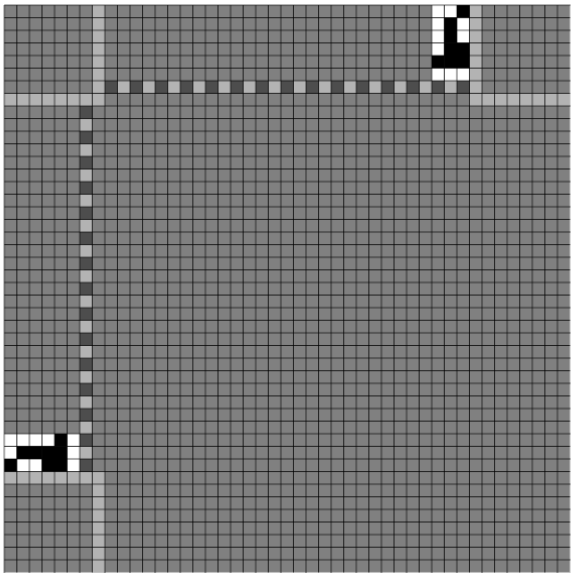

Рисунок 1.28 –Розміщення інформації про версію в QR-коді

Нижче детально описано розташування рядка з інформацією про версію для кожної області. Генерування бітів виправлення помилок для рядка інформації про версію дещо відрізняється від генерування кодових слів помилок. Біти так само створюються шляхом ділення полінома повідомлення на поліном генератора, але деталі дещо відрізняються [22].

Верхній правий блок	Нижній лівий блок
| 17 | 16 | 15 |	| 17 | 16 | 15 | 14 | 13 | 12 |
| 14 | 13 | 12 |	| 11 | 10 | 9 | 8 | 7 | 6 |
| 11 | 10 | 9 |	| 5 | 4 | 3 | 2 | 1 | 0 |
| 8 | 7 | 6 |	
| 5 | 4 | 3 |	
| 2 | 1 | 0 |	

Індекс 0 – старший лівий біт, а індекс 17 – старший правий біт [8].

1.4. Ризики використання QR-кодів

Єдиним контекстом, в якому звичайні QR-коди можуть нести виконувані дані, є тип даних URL. Ці URL-адреси можуть містити код JavaScript, який може бути використаний для використання вразливостей у програмах на хост-системі, таких як зчитувач, веб-браузер або програма для перегляду зображень, оскільки зчитувач зазвичай надсилає дані до програми, пов'язаної з типом даних, що використовується в QR-коді.

У разі відсутності програмних експлойтів (див. Рис. 1.7.1), шкідливі QR-коди в поєднанні з дозволеним зчитувачем все одно можуть поставити під загрозу вміст комп'ютера і конфіденційність користувача. Ця практика відома як «прикріплення», портфоліо «атакуючих тегів». Вони легко створюються і можуть бути прикріплені поверх законних QR-кодів. На смартфоні дозволи зчитувача можуть дозволити використання камери, повний доступ до Інтернету, читання/запис контактних даних, GPS, читання історії браузера, читання/запис локальної пам'яті та глобальні системні зміни (див. Рис. 1.29).

Рисунок 1.29 – QR-код, що містить експлойт

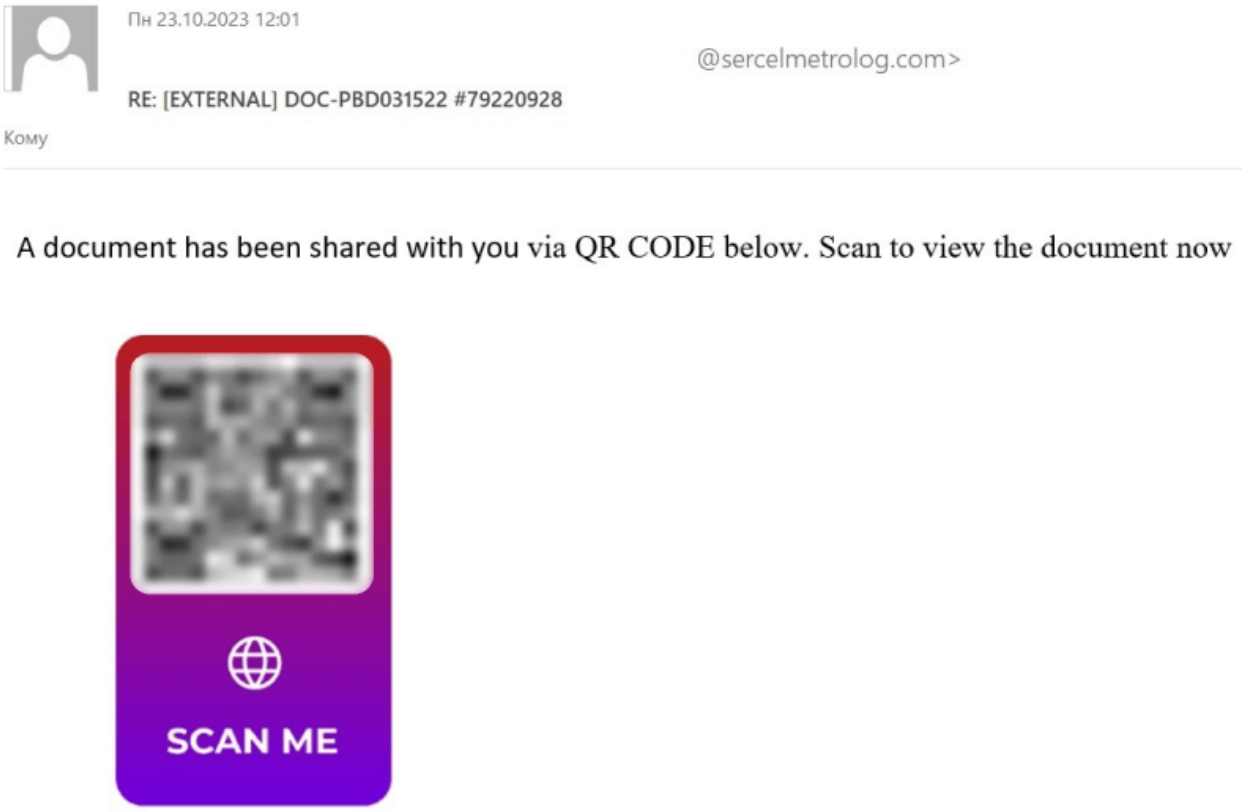

Рисунок 1.30 – Фішинговий QR-код, надісланий поштою

Ризики включають посилання на небезпечні веб-сайти за допомогою уразливостей браузера, увімкнення мікрофона/камери/ GPS, а потім передачу цих каналів на віддалений сервер, аналіз конфіденційних даних (паролів, файлів, контактів, транзакцій), а також надсилання email/SMS/IM-повідомлень або пакетів для DDoS-атак у складі бот-мережі, пошкодження налаштувань конфіденційності, крадіжки особистих даних, і навіть самі дії, що містять шкідливу логіку, таку як JavaScript або вірус (див. Рис. 1.30).

Ці дії можуть відбуватися у фоновому режимі, в той час як користувач бачить лише зчитувач, що відкриває на перший погляд нешкідливу веб-сторінку. У Росії шкідливий QR-код змусив телефони, які його відсканували, надсилати преміум-повідомлення вартістю $6 за кожне (див. Рис. 1.31). QR-коди також були пов'язані з шахрайством, коли на паркомати (див. Рис. 1.32-1.33). наклеювали наклейки, які видавали за можливість швидкої оплати, як це було в Остіні, Сан-Антоніо та Бостоні, серед інших міст США та Австралії.

Рисунок 1.31 – QR-код, що містить шкідливі скрипти

Рисунок 1.32 – Шкідливий QR-код

Рисунок 1.33 – QR-код на паркувальному майданчику

Підсумовуючи, QR-коди є надзвичайно зручними в користуванні, однак разом з їхніми перевагами, існує і ризик. У руках шахраїв QR-коди можуть стати потужним засобом для поширення шкідливих програм, фішингу або спрямування користувачів на небезпечні веб-сайти. Зловмисники можуть замаскувати шкідливі посилання під виглядом легітимних QR-кодів, що може призвести до серйозних наслідків для тих, хто їх сканує.

Тому важливо бути обережними під час використання QR-кодів, скануючи їх лише з надійних джерел і уникати підозрілих або незнайомих кодів, щоб захистити себе від можливих загроз.

1.5. Висновки до розділу I

У розділі було розглянуто основи QR-кодів, їх структуру, а також важливі аспекти, пов'язані з їх використанням у сучасному світі. Було встановлено, що QR-коди представляють собою потужний інструмент для зберігання та передачі даних, який широко застосовується в різних сферах, зокрема в маркетингу, бізнесі та системах безпеки.

Особлива увага була приділена захисту даних при використанні QR-кодів. Розглянуто методи шифрування та аутентифікації, які дозволяють забезпечити конфіденційність інформації, що передається через QR-коди, а також важливість застосування стандартів безпеки для запобігання зловживанням.

Вивчено також обробку помилок, пов'язаних з QR-кодами, що дозволяє підвищити їх надійність у реальних умовах використання. Встановлено, що правильно налаштовані алгоритми корекції помилок є ключовими для забезпечення точності сканування, навіть за наявності пошкоджень або дефектів у коді. Розкрито вміст QR-кодів, який може включати текстову, числову та іншу інформацію, і як це впливає на ефективність та функціональність QR-кодів у різних сферах. Важливим аспектом є оптимізація структури QR-кодів для забезпечення швидкого доступу до збереженої інформації.

Шаблони для QR-кодів також було охарактеризовано як ефективний інструмент для стандартизації кодування та спрощення процесів генерації QR-кодів для конкретних застосувань. Окремо варто зазначити ризики, пов'язані з використанням QR-кодів, зокрема можливість підробки та перенаправлення користувачів на шкідливі ресурси. У зв'язку з цим необхідно впроваджувати додаткові заходи безпеки для мінімізації цих ризиків.

Отже, QR-коди є ефективним і зручним засобом для зберігання та передачі інформації, однак їх використання повинно супроводжуватися дотриманням високих стандартів безпеки та контролю для забезпечення надійності і захисту даних.

РОЗДІЛ ІІ
ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Середовище розробки

Visual Studio Code (VS Code) – дуже популярний і потужний редактор коду, який пропонує численні переваги для розробки Python (див. Рис. 2.1.1). Завдяки широкому набору функцій, розширюваності та простоті використання він є популярним серед розробників. У цій відповіді ми розглянемо переваги використання Visual Studio Code як редактора коду для розробки Python.

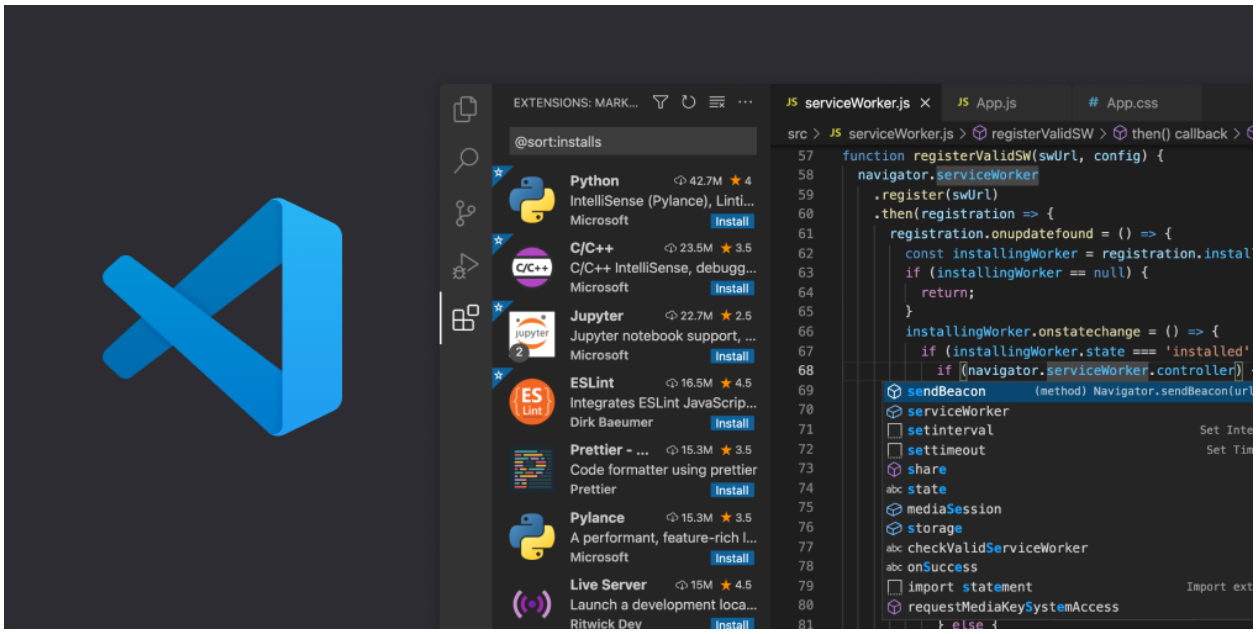

Рисунок 2.1 – Вигляд програмного забезпечення Visual Studio Code

Однією з головних переваг використання Visual Studio Code є його чудова підтримка Python. VS Code забезпечує інтелектуальне завершення коду, підсвічування синтаксису та перевірку помилок для коду Python. Це допомагає розробникам писати чистий і безпомилковий код, виявляючи потенційні помилки в режимі реального часу. Крім того, VS Code пропонує інтегровані можливості налагодження, що дозволяють розробникам встановлювати точки зупинки, перевіряти змінні та покроково виконувати код, що полегшує виявлення та виправлення проблем.

Ще однією перевагою використання Visual Studio Code є широка екосистема розширень (див. Рис. 2.2). VS Code підтримує широкий спектр розширень, які розширюють його функціональність і задовольняють конкретні потреби. Для розробки на Python доступні розширення для популярних фреймворків, що надають такі функції, як фрагменти компонентів, форматування коду та проектні риштування. Ці розширення можуть значно підвищити продуктивність, автоматизуючи повторювані завдання та надаючи ярлики для поширених шаблонів коду.

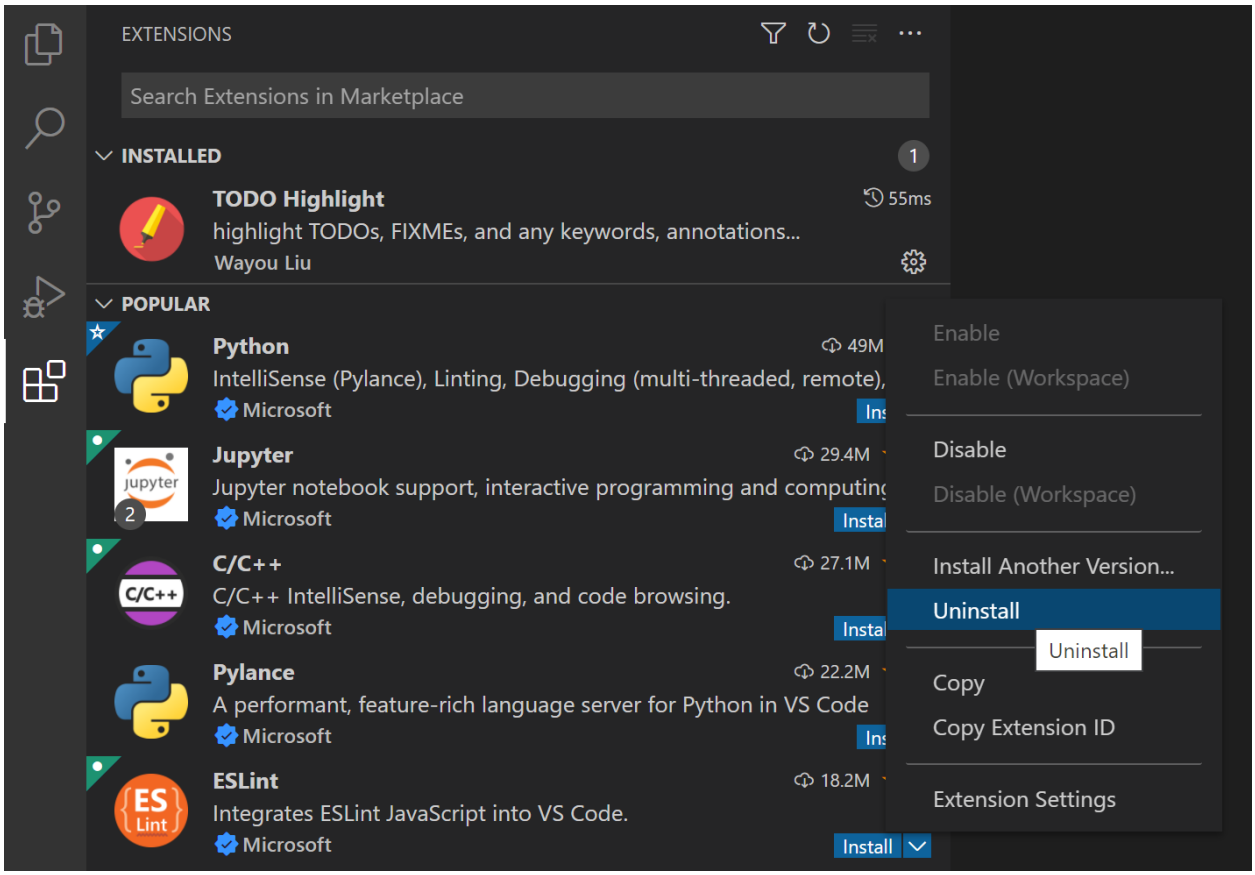

Рисунок 2.2 – Розширення для програмного забезпечення Visual Studio Code

VS Code також пропонує бездоганну інтеграцію з системами контролю версій, такими як Git (див. Рис. 2.3). Він надає вбудоване вікно контролю вихідного коду, яке дозволяє розробникам вносити, фіксувати і проштовхувати зміни безпосередньо з редактора. Ця інтеграція спрощує спільну роботу і полегшує відстеження та управління змінами коду.

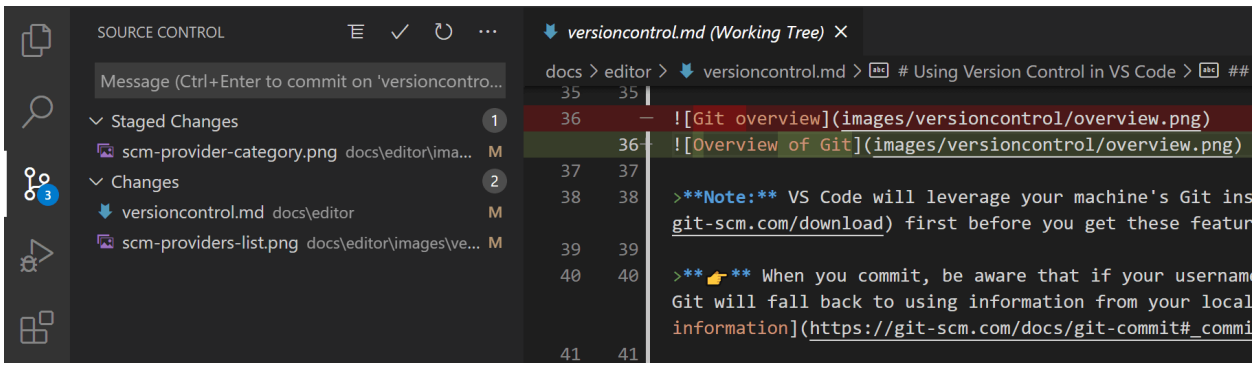

Рисунок 2.3 – Git для програмного забезпечення Visual Studio Code

Крім того, Visual Studio Code надає середовище, яке легко налаштовується та адаптується. Воно дозволяє розробникам персоналізувати свій робочий простір, обираючи теми, налаштовуючи сполучення клавіш та різні параметри. Така гнучкість дозволяє розробникам створювати середовище, яке відповідає їхнім уподобанням і підвищує продуктивність.

На додаток до цих переваг, Visual Studio Code має чудову підтримку інших веб-технологій, які зазвичай використовуються при розробці Python. Вона забезпечує потужну підтримку HTML і CSS з такими функціями, як автозавершення, форматування і попередній перегляд в реальному часі. Він також пропонує інтегровану підтримку терміналу, що дозволяє розробникам запускати інструменти командного рядка і скрипти, не виходячи з редактора.

Підсумовуючи, переваги використання Visual Studio Code як редактора коду для розробки Python включають в себе відмінну підтримку Python, широку екосистему розширень, безшовну інтеграцію з системами контролю версій, налаштовуване середовище та потужну підтримку інших веб-технологій. Ці особливості роблять Visual Studio Code потужним інструментом для розробки Python, що підвищує продуктивність і дозволяє розробникам ефективно писати високоякісний код [9].

Завдяки активній спільноті, VS Code розвивається надзвичайно швидко, нові функції впроваджуються щодня, всі проблеми вирішуються швидко, а користувацький досвід постійно покращується. Тому не дивно, що цей інструмент вже багато років є улюбленим для багатьох розробників [10].

2.2 Мова програмування та бібліотека qrcode

Ідея Python виникла у 1989 році, коли її творець Гвідо ван Россум зіткнувся з недоліками мови ABC (а саме, з проблемою розширюваності). Россам розпочав роботу над створенням нової мови, яка об'єднала в собі всі хороші риси мови ABC та нові бажані можливості, такі як розширюваність та обробка винятків. Python 1.0 був випущений в 1994 році; він запозичив систему модулів з Modula-3, мав можливість взаємодіяти з операційною системою Amoeba і включав функціональні інструменти програмування.

У 2000 році основна команда розробників Python переїхала на Beopen.com, і в жовтні 2000 року вийшла версія Python 2.0 з багатьма поліпшеннями, включаючи збирач сміття і підтримку Unicode.

У грудні 2008 року вийшов Python 3.0, який відмовився від зворотної сумісності та отримав новий дизайн, щоб уникнути дублюючих конструкцій та модулів. Це все ще багатопарадигмальна мова, що пропонує розробникам можливості об'єктної орієнтації, структурного програмування та функціонального програмування.

Сьогодні Python має декілька реалізацій, включаючи Python, написаний на мові Java для віртуальної машини Java; IronPython, написаний на C# для Common Language Infrastructure, та версію PyPy, написану на RPython і перекладену на C. Слід зазначити, що CPython, написаний на C і розроблений Python Software Foundation, є стандартною і найпопулярнішою реалізацією Python. Хоча ці реалізації працюють рідною мовою, якою вони написані, вони також здатні взаємодіяти з іншими мовами за допомогою модулів. Більшість цих модулів працюють за моделлю розробки спільноти, мають відкритий вихідний код і є безкоштовними [11].

Випущений у лютому 2015 року, Python 3.4.3 пропонує значне покращення підтримки Unicode, серед інших нових можливостей.

Різноманітне застосування мови Python є результатом поєднання функцій, які надають цій мові перевагу над іншими. Деякі з переваг програмування на Python включають:

1. Наявність сторонніх модулів: індекс пакетів Python (PyPI) містить численні модулі сторонніх розробників, які роблять Python здатним взаємодіяти з більшістю інших мов та платформ.

2. Широка бібліотека підтримки: Python надає велику стандартну бібліотеку, яка включає такі області, як інтернет-протоколи, операції з рядками, інструменти веб-сервісів та інтерфейси операційної системи. Багато завдань програмування з високим рівнем використання вже вписано в стандартну бібліотеку, що значно скорочує довжину коду, який потрібно написати.

3. Відкритий вихідний код та розвиток спільноти: мова Python розробляється за ліцензією з відкритим вихідним кодом, схваленою OSI, що робить її вільною для використання та розповсюдження, в тому числі в комерційних цілях.

Крім того, її розвиток стимулюється спільнотою, яка співпрацює над її кодом шляхом проведення конференцій та списків розсилки, а також надає численні модулі.

4. Легкість у вивченні та доступна підтримка: Python пропонує чудову читабельність і простий у вивченні синтаксис, що допомагає початківцям використовувати цю мову програмування. Рекомендації щодо стилю коду, PEP 8, надають набір правил для полегшення форматування коду. Крім того, широка база користувачів та активних розробників призвела до створення багатого банку інтернет-ресурсів для заохочення розвитку та подальшого впровадження мови.

5. Зручні структури даних: Python має вбудовані структури даних у вигляді списків і словників, які можна використовувати для побудови швидких структур даних під час виконання. Крім того, Python також надає

можливість динамічної високорівневої типізації даних, що зменшує довжину необхідного допоміжного коду.

6. Продуктивність та швидкість: Python має чистий об'єктно-орієнтований дизайн, надає розширені можливості управління процесами, має потужні можливості інтеграції та обробки тексту, а також власний фреймворк для модульного тестування, що сприяє підвищенню швидкості та продуктивності. Python вважається життєздатним варіантом для створення складних багатопротокольних мережевих додатків [12].

Застосування Python (див. Рис. 2.4):

- Десктопні програми на основі графічного інтерфейсу;
- Програми для обробки зображень та графічного дизайну;
- Наукові та обчислювальні програми;
- Ігри;
- Веб-фреймворки та веб-додатки;
- Корпоративні та бізнес-додатки;
- Операційні системи;
- Розробка мов;
- Прототипування.

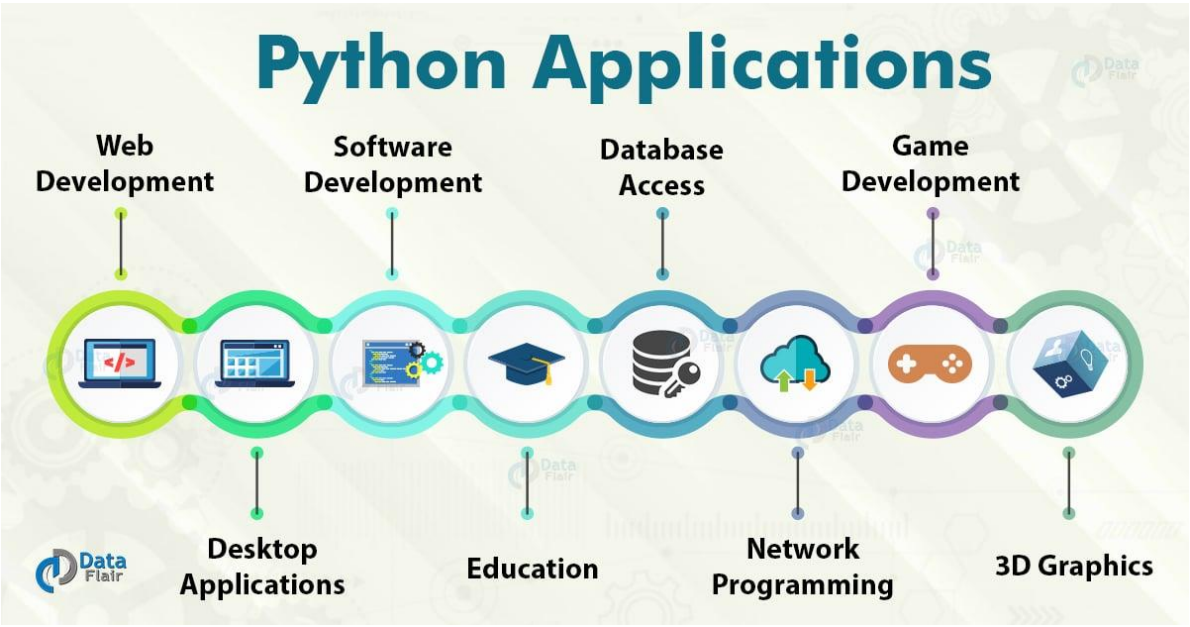

Рисунок 2.4 – Приклади застосування мови Python

Python – це потужна та універсальна мова програмування, яка може бути використана для створення широкого спектру додатків, в тому числі тих, що генерують QR-коди. Однією з найпопулярніших бібліотек для створення QR-кодів у Python є бібліотека qrcode. Ця бібліотека дозволяє розробникам легко генерувати QR-коди, надаючи простий та інтуїтивно зрозумілий інтерфейс для створення та налаштування кодів.

Щоб створити QR-код за допомогою Python, спочатку потрібно встановити бібліотеку qrcode за допомогою команди pip install qrcode. Після встановлення бібліотеки ви можете імпортувати її у свій скрипт Python і використовувати для створення QR-коду, надавши інформацію, яку ви хочете закодувати, наприклад, текст або URL-адресу. Ви також можете налаштувати зовнішній вигляд QR-коду, змінивши розмір, колір і рівень корекції помилок.

Крім того, ви можете використовувати Python для створення QR-кодів з більш розширеними можливостями, такими як включення логотипу або зображення в QR-код або створення QR-кодів з більш складними типами даних. Бібліотеки Python, такі як Pillow і qrcodegen, можна використовувати в поєднанні з бібліотекою qrcode для додавання таких функцій.

Таким чином, Python забезпечує простий та ефективний спосіб створення QR-кодів, що робить його популярним серед розробників. За допомогою його простих у використанні бібліотек розробники можуть легко генерувати широкий спектр QR-кодів, налаштовувати їх зовнішній вигляд і додавати розширені функції [17].

Встановлення бібліотеки qrcode у Python – це простий процес, який можна виконати за допомогою менеджера пакетів pip:

1. Відкрийте командний рядок або термінал на вашому комп'ютері.
2. Введіть команду «pip install qrcode» і натисніть enter. Почнеться процес встановлення бібліотеки qrcode.
3. Дочекайтеся завершення інсталяції. Ви повинні побачити повідомлення про успішне встановлення бібліотеки.

Крім того, можна встановити бібліотеку за допомогою команди «!pip install qrcode», якщо ви використовуєте скрипт python в Jupyter notebook або аналогічну програму [18].

Можна перевірити, чи бібліотека була успішно встановлена, набравши «import qrcode» в python-скрипті і виконавши його. Якщо не з'явиться жодної помилки, це означає, що бібліотека успішно встановлена.

Варто зазначити, що якщо ви працюєте у віртуальному середовищі або на вашому комп'ютері встановлено декілька версій python, можливо, вам доведеться використовувати pip3 або pip3.x (залежно від версії python) замість pip для встановлення бібліотеки [13].

2.3 Архітектура проекту

Спрощена архітектура розроблюваного проекту для генерації та розпізнавання QR-кодів має такі елементи:

Архітектура цього проєкту складається з декількох основних компонентів:

1. Інтерфейс користувача (gui):

- tkinter: бібліотека для створення графічного інтерфейсу, що використовується для побудови віконного інтерфейсу, додавання кнопок, полів введення та інших елементів.
- root: головне вікно програми, яке містить всі елементи gui.
- label_data, label_folder: мітки для позначення полів введення даних і вибору папки.
- data_entry: поле введення тексту для даних, які потрібно закодувати в qr-код.
- folder_entry: поле введення шляху до папки, де буде збережено згенерований qr-код.

- button_browse: кнопка для відкриття діалогового вікна вибору папки.
- button_generate: кнопка для генерації qr-коду.
- button_decode: кнопка для вибору файлу і розшифрування qr-коду.
- button_language: кнопка для перемикання мови інтерфейсу між англійською та українською.

2. Функції обробки даних:

- generate_qr_code(data, folder): функція для створення qr-коду з введених даних і збереження його у вказаній папці. Вона використовує бібліотеку `qrcode`.
- decode_qr_code(image_path): функція для розшифрування qr-коду з зображення. Вона використовує бібліотеку `pyzbar` для декодування даних із зображення.

3. Функції для роботи з файловою системою:

- select_folder(): функція для відкриття діалогового вікна вибору папки та отримання шляху до неї.
- select_and_decode_qr(): функція для відкриття діалогового вікна вибору файлу із зображенням qr-коду та розшифрування даних.

4. Функції для керування інтерфейсом:

- handle_generate(): функція, що перевіряє наявність даних і папки, після чого викликає функцію генерації qr-коду.
- update_language(language): функція для оновлення тексту елементів інтерфейсу залежно від вибраної мови (англійської або української).
- center_window(window): функція для центрування вікна програми на екрані.

5. Налаштування стилю інтерфейсу:

- всі елементи інтерфейсу оформлені в єдиному стилі за допомогою параметрів кольорів фону та тексту, що налаштовуються через змінні в словнику `style_options`.

6. Переключення мов інтерфейсу:

- в проекті реалізована функціональність багатомовності, що дозволяє переключати інтерфейс між двома мовами (англійською та українською) за допомогою функції `update_language()` і кнопки `button_language`.

Зовнішні бібліотеки:

- qrcode: для генерації qr-кодів.
- pyzbar: для р озшифрування qr-кодів з зображень.
- pil (pillow): для роботи із зображеннями, використовується для відкриття файлів зображень.

Основний потік: програма починається з налаштування графічного інтерфейсу, після чого ініціалізується обрана мова і викликається головний цикл `root.mainloop()`, що відповідає за відображення і підтримку взаємодії з користувачем [19].

Схематично архітектуру програмного забезпечення з графічним інтерфейсом для створення й обробки QR-кодів показано за допомогою схеми на Рис. 2.5.

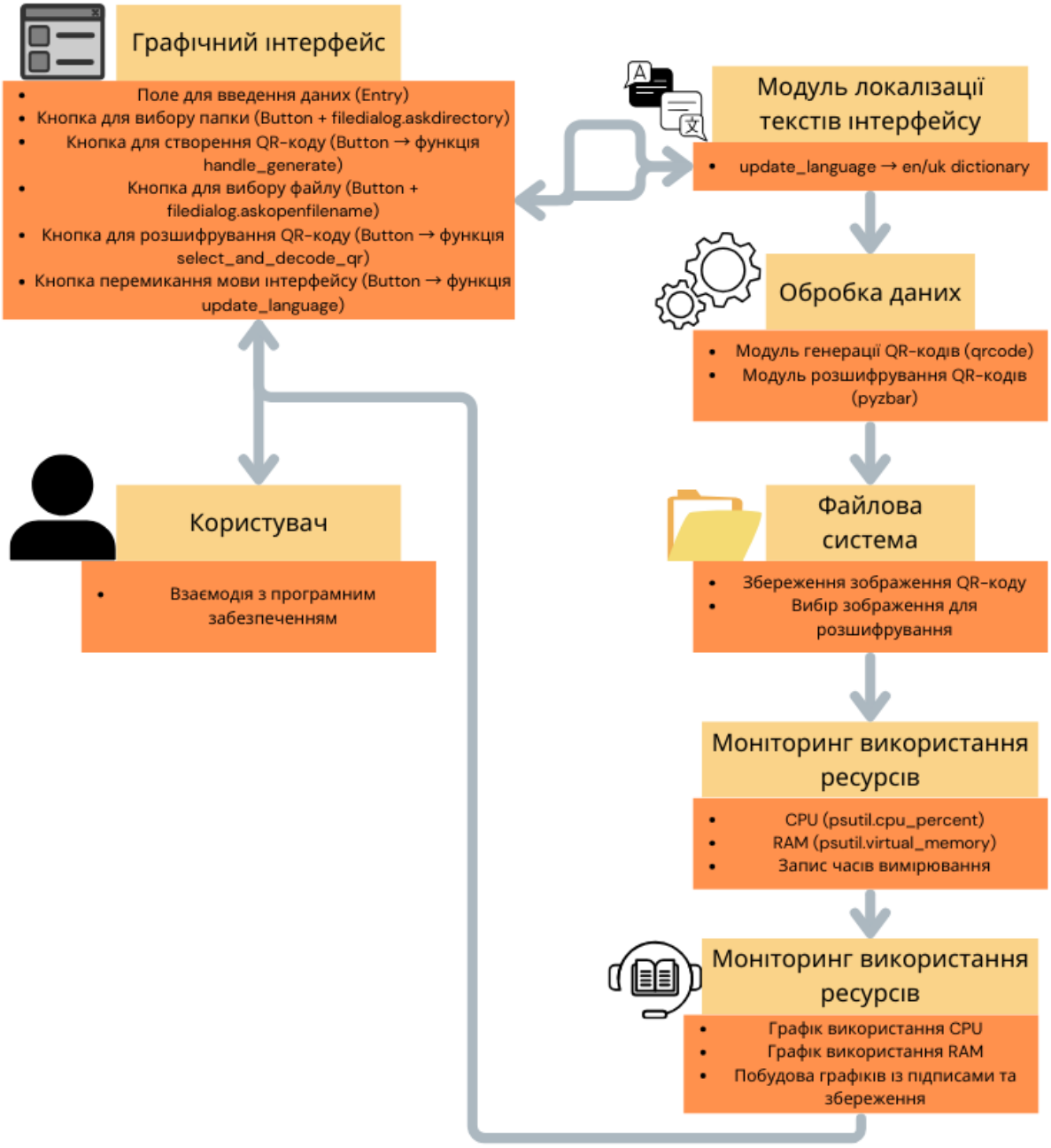

Рисунок 2.5 – Архітектура програмного забезпечення

2.4 Висновки до розділу II

У розділі були розглянуті основні аспекти, пов'язані з розробкою програмного забезпечення для створення та обробки QR-кодів, включаючи вибір середовища розробки, мови програмування, використані бібліотеки та архітектуру проекту.

Було визначено, яке програмне середовище є найбільш підходящим для розробки даного програмного забезпечення. Розглянуті переваги та недоліки різних IDE, а також можливості для інтеграції з іншими інструментами, що значно полегшують процес розробки та тестування програм.

Обрано мову програмування, що найкраще підходить для вирішення завдань, пов'язаних із створенням і обробкою QR-кодів. Проаналізовано основні можливості мови, її ефективність для швидкої обробки даних і підтримки необхідних бібліотек.

Бібліотека qrcode, використана для роботи з QR-кодами, має значний функціонал, що дозволяє ефективно генерувати та зчитувати ці коди. Розглянуто її основні переваги та приклади використання в контексті вирішення завдань, пов'язаних із QR-кодами.

Архітектура проекту була побудована таким чином, щоб забезпечити максимальну ефективність, зручність використання та масштабованість програмного забезпечення. Описано ключові модулі та їх взаємодія, що дозволяє ефективно реалізувати всі необхідні функціонали.

Таким чином, розділ висвітлює ключові технологічні рішення, які впливають на розробку програмного забезпечення для роботи з QR-кодами, підкреслюючи важливість правильно підібраного середовища розробки, мови програмування, бібліотек і архітектурних рішень для досягнення ефективності та надійності розробленого продукту.

РОЗДІЛ ІІІ
РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1. Реалізація програми

Програма складається з кількох основних блоків, що забезпечують її функціональність.

Перший блок відповідає за інтерфейс користувача (GUI), створений за допомогою бібліотеки `tkinter` (див. Рис. 3.1). Він включає головне вікно, в якому розташовані різні елементи, такі як поля введення, кнопки, і мітки. Користувач може вводити дані для генерації QR-коду або вибирати зображення для декодування QR-коду через цей інтерфейс.

```
# Налаштування інтерфейсу
root = tk.Tk()
root.title("QR Code Generator and Decoder")

# Встановлення ширини вікна
root.geometry("550x300")

# Налаштування фону вікна та стилів
root.configure(bg="#2E4053")

style_options = {
    "bg": "#2E4053",
    "fg": "white",
    "font": ("Helvetica", 12)
}

# Поле для введення даних
label_data = tk.Label(root, text="", **style_options)
label_data.grid(row=0, column=0, padx=10, pady=10, sticky="e")
data_entry = tk.Entry(root, width=50)  # Збільшення ширини за необхідністю
data_entry.grid(row=0, column=1, padx=10, pady=10)

# Вибір папки
label_folder = tk.Label(root, text="", **style_options)
label_folder.grid(row=1, column=0, padx=10, pady=10, sticky="e")
folder_entry = tk.Entry(root, width=50)  # Збільшення ширини за необхідністю
folder_entry.grid(row=1, column=1, padx=10, pady=10)
button_browse = tk.Button(root, text="", command=select_folder, bg="#1ABC9C", fg="white")
button_browse.grid(row=1, column=2, padx=10, pady=10)
```

Рисунок 3.1 – Блок інтерфейсу користувача (GUI)

Другий блок складається з функцій для обробки даних. Зокрема, це функція генерації QR-коду, яка перетворює введені користувачем дані на QR-код і зберігає його у вибраній папці, та функція для декодування QR-коду, яка зчитує дані з вибраного зображення (див. Рис. 3.2).

```python
# Функція для генерації QR-коду
def generate_qr_code(data, folder):
    qr = qrcode.QRCode(
        version=1,
        error_correction=qrcode.constants.ERROR_CORRECT_L,
        box_size=10,
        border=4,
    )

    qr.add_data(data)
    qr.make(fit=True)
    img = qr.make_image(fill_color="black", back_color="white")

    filename = f"{folder}/example_qr.png"
    img.save(filename)

    messagebox.showinfo(language_strings['success'], f"{language_strings['qr_generated']} {filename}")
```

Рисунок 3.2 – Блок функцій для обробки даних

Третій блок включає функції для роботи з файловою системою, що дозволяють користувачеві вибирати папку для збереження файлу або вибирати файл із зображенням QR-коду для декодування (див. Рис. 3.3).

```python
# Функція для відкриття діалогового вікна та вибору шляху до папки
def select_folder():
    folder_path = filedialog.askdirectory(title=language_strings['select_folder'])
    folder_entry.delete(0, tk.END)
    folder_entry.insert(0, folder_path)

# Функція для вибору файлу та розшифрування QR-коду
def select_and_decode_qr():
    image_path = filedialog.askopenfilename(title=language_strings['select_qr_image'], filetypes=[("PNG files", "*.png")])
    if image_path:
        decoded_data = decode_qr_code(image_path)
        messagebox.showinfo(language_strings['decoded_data'], f"{language_strings['decoded_qr_data']}: {decoded_data}")
```

Рисунок 3.3 – Блок функцій для роботи з файловою системою

Четвертий блок складається з функцій для керування інтерфейсом. Вони забезпечують реакцію на дії користувача, такі як натискання кнопок для генерації або декодування QR-коду, а також зміну мови інтерфейсу (див. Рис. 3.4). Крім того, є функція, що центрує вікно програми на екрані, роблячи інтерфейс зручнішим.

```python
# Функція для центрування вікна на екрані
def center_window(window):
    window.update_idletasks()
    width = window.winfo_width()
    height = window.winfo_height()
    x = (window.winfo_screenwidth() // 2) - (width // 2)
    y = (window.winfo_screenheight() // 2) - (height // 2)
    window.geometry(f'{width}x{height}+{x}+{y}')
```

Рисунок 3.4 – Блок керування інтерфейсом

Нарешті, п'ятий блок присвячений налаштуванню стилю інтерфейсу, де визначені кольори фону і тексту для всіх елементів, що дозволяє зробити програму візуально приємною і зручною у використанні (див. Рис. 3.5).

```python
# Встановлення ширини вікна
root.geometry("550x300")

# Налаштування фону вікна та стилів
root.configure(bg="#2E4053")

style_options = {
    "bg": "#2E4053",
    "fg": "white",
    "font": ("Helvetica", 12)
}
```

Рисунок 3.5 – Блок налаштування стилю інтерфейсу

Ці блоки працюють разом для створення програми, яка дозволяє користувачеві легко генерувати і декодувати QR-коди через зручний графічний інтерфейс на двох мовах на вибір: українська або англійська (див. Рис. 3.6-3.7).

```
language_strings = {
    "enter_data": "Enter Data:",
    "save_to_folder": "Save to Folder:",
    "browse": "Browse",
    "generate_qr": "Generate QR Code",
    "decode_qr": "Decode QR Code",
    "success": "Success",
    "qr_generated": "QR code generated and saved as",
    "select_folder": "Select Folder",
    "select_qr_image": "Select QR Code Image",
    "decoded_data": "Decoded Data",
    "decoded_qr_data": "Decoded QR code data",
    "input_error": "Input Error",
    "enter_data_folder": "Please enter data and select a folder.",
    "no_qr_found": "No QR code found",
    "language": "Language: English",
    "switch_language": "Switch to Ukrainian"
}
```

Рисунок 3.6 – Блок налаштування мови інтерфейсу

```
else:
    language_strings = {
        "enter_data": "Введіть дані:",
        "save_to_folder": "Зберегти в папку:",
        "browse": "Огляд",
        "generate_qr": "Створити QR код",
        "decode_qr": "Розшифрувати QR код",
        "success": "Успіх",
        "qr_generated": "QR код створено та збережено як",
        "select_folder": "Виберіть папку",
        "select_qr_image": "Виберіть зображення QR коду",
        "decoded_data": "Розшифровані дані",
        "decoded_qr_data": "Розшифровані дані QR коду",
        "input_error": "Помилка введення",
        "enter_data_folder": "Будь ласка, введіть дані та виберіть папку.",
        "no_qr_found": "QR код не знайдено",
        "language": "Мова: Українська",
        "switch_language": "Переключити на англійську"
    }
```

Рисунок 3.7 – Блок налаштування мови інтерфейсу

Повний код програмної реалізації проекту наведено в Додатку А.

Таким чином, після запуску програми, користувач отримує такий вигляд програми та етапи роботи з нею, як показано на Рис. 3.8-3.14.

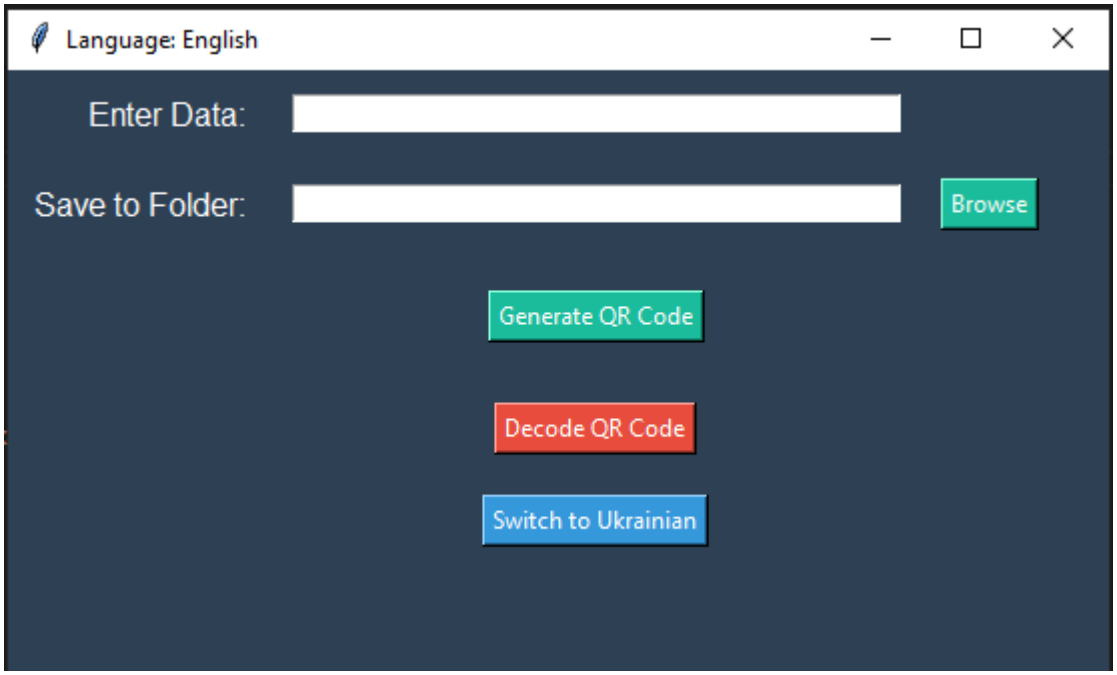

Рисунок 3.8 – Робота спроектованого програмного забезпечення

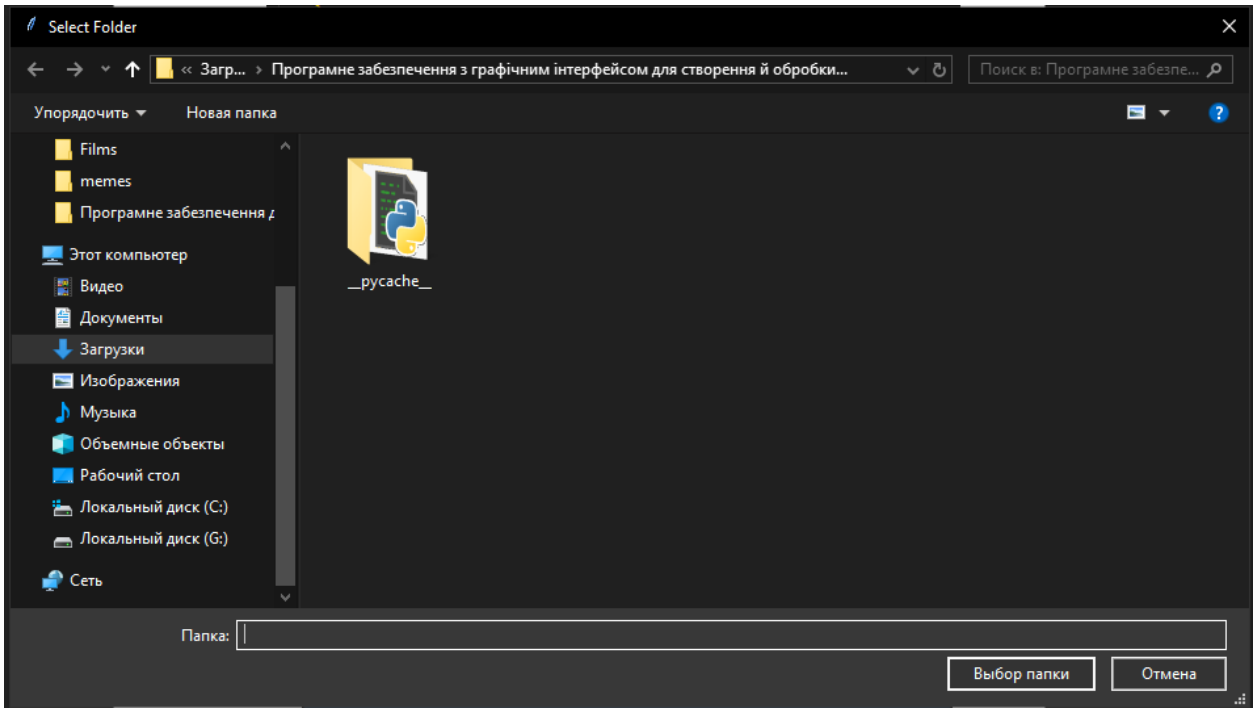

Рисунок 3.9 – Робота спроектованого програмного забезпечення

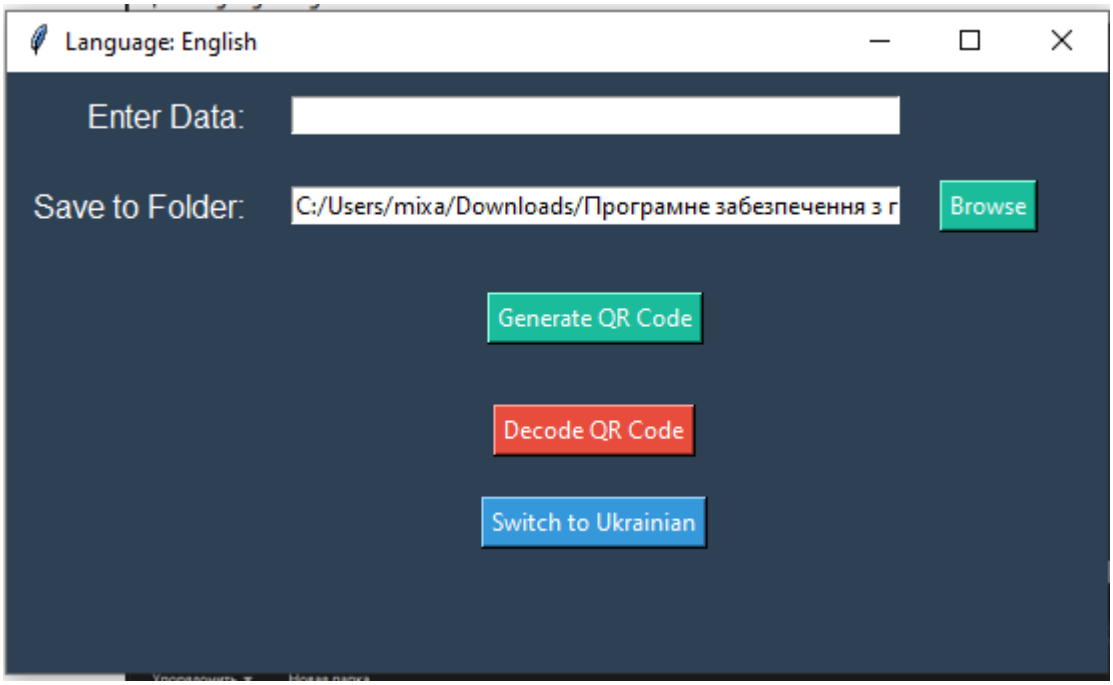

Рисунок 3.10 – Робота спроектованого програмного забезпечення

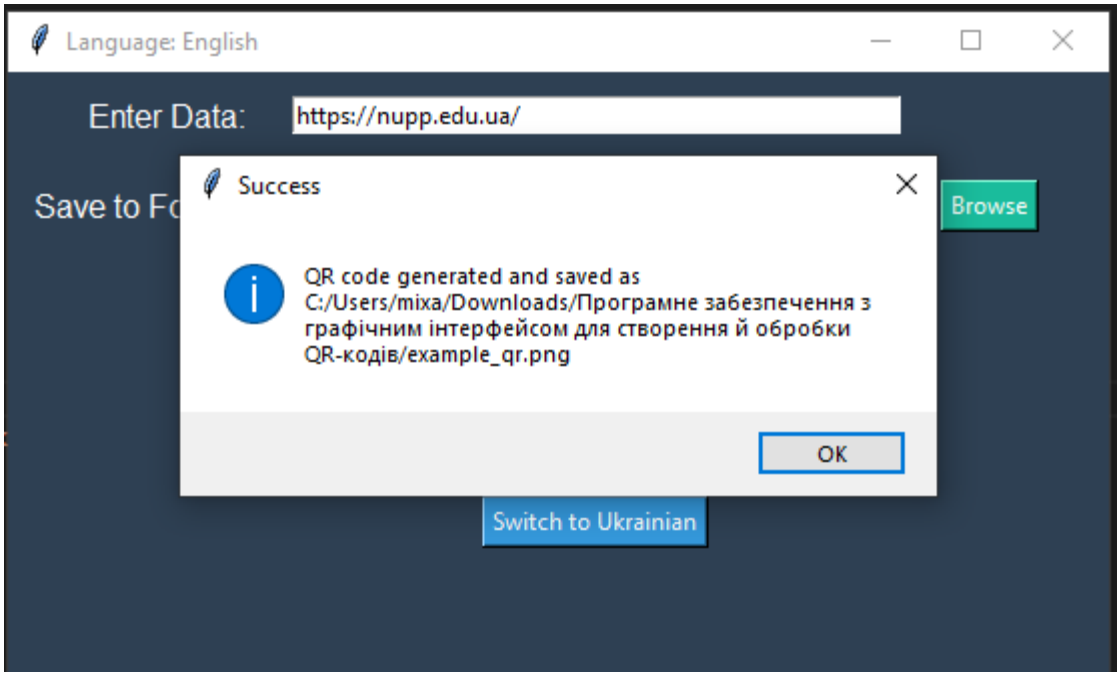

Рисунок 3.11 – Робота спроектованого програмного забезпечення

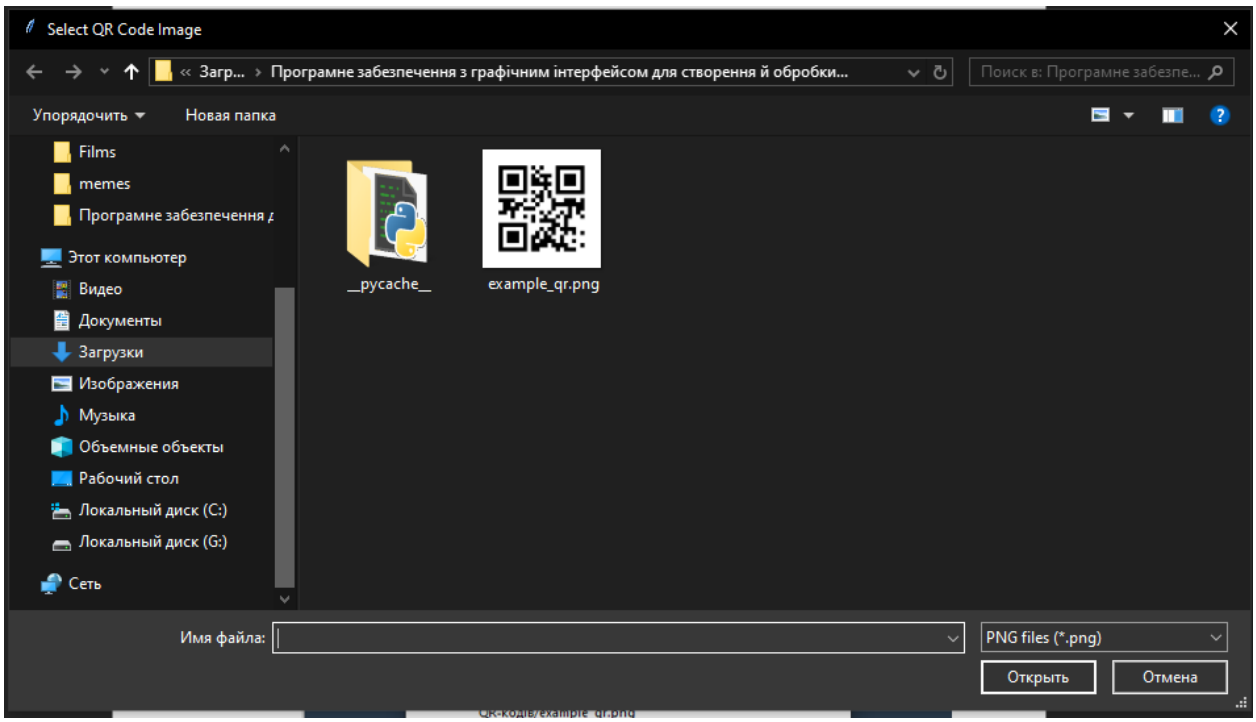

Рисунок 3.12 – Робота спроектованого програмного забезпечення

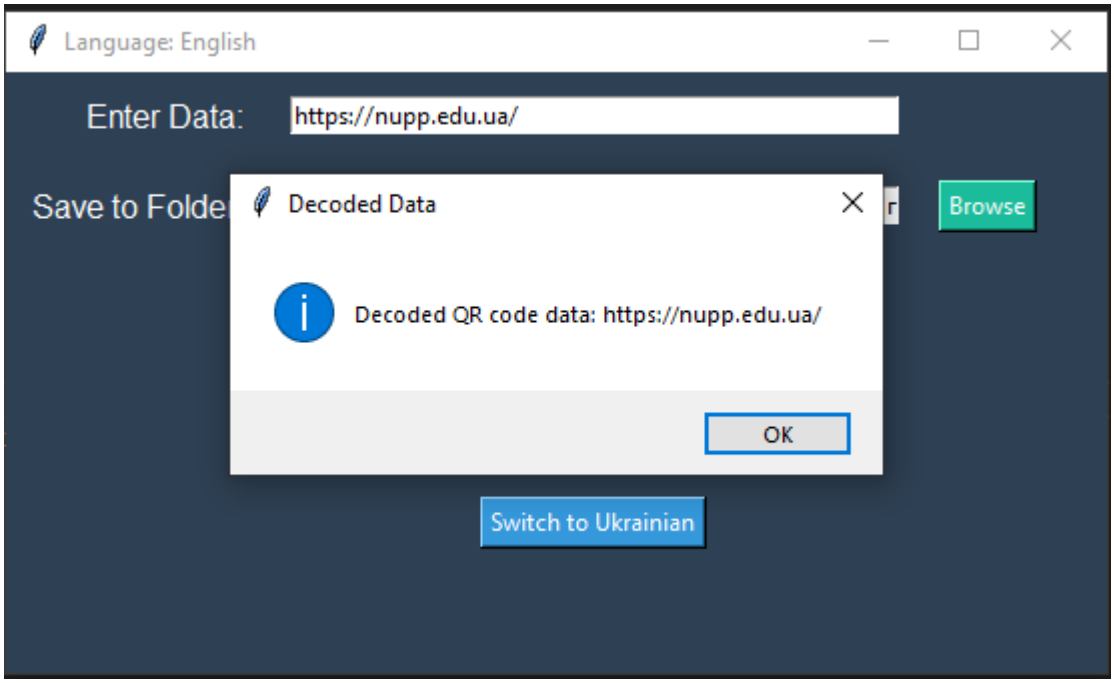

Рисунок 3.13 – Робота спроектованого програмного забезпечення

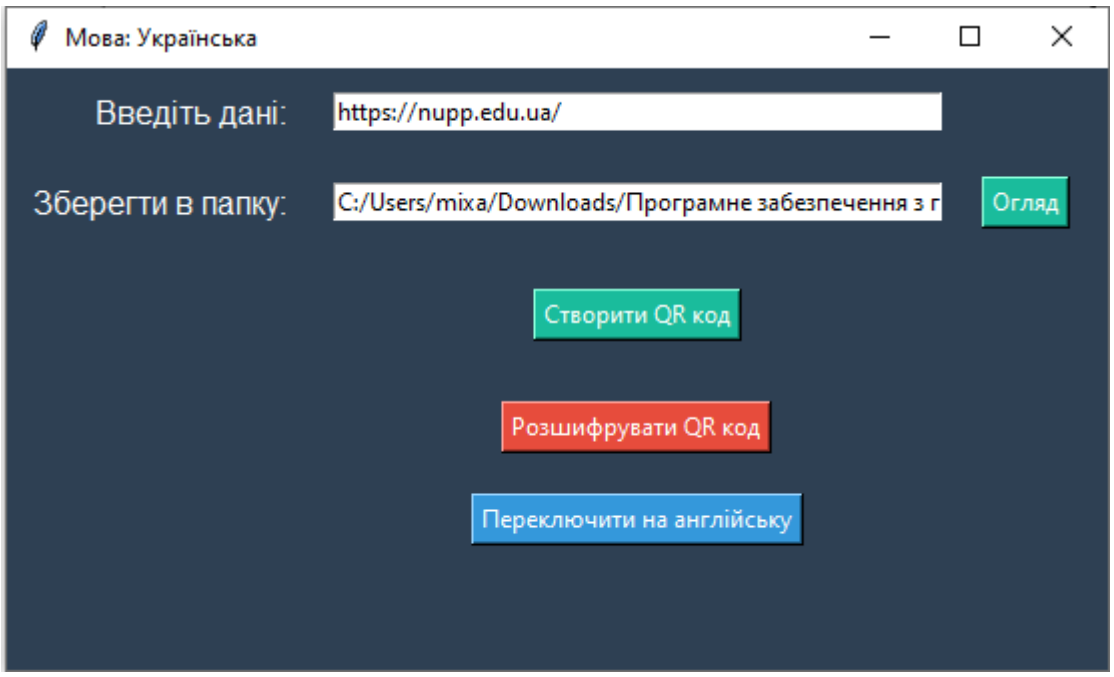

Рисунок 3.14 – Робота спроектованого програмного забезпечення

У результаті програмне забезпечення створює файл «example_qr.png», що містить у собі текст, введений у відвідне поле (у нашому випадку – посилання на сайт Національного університету «Полтавська політехніка імені Юрія Кондратюка»), як показано на Рис. 3.15.

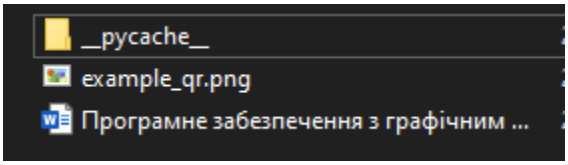

Рисунок 3.15 – Створений програмою файл «example_qr.png»

Рисунок 3.16 – Створений програмою файл «example_qr.png»

Отриманий QR-код можна зчитати з будь-якого пристрою, що підтримує функцію розпізнавання QR-кодів. Для перевірки отриманого результату зчитаємо згенерований QR-код за допомогою смартфону з операційною системою Android 14. Результати перевірки наведені у вигляді Рис. 3.17-3.18.

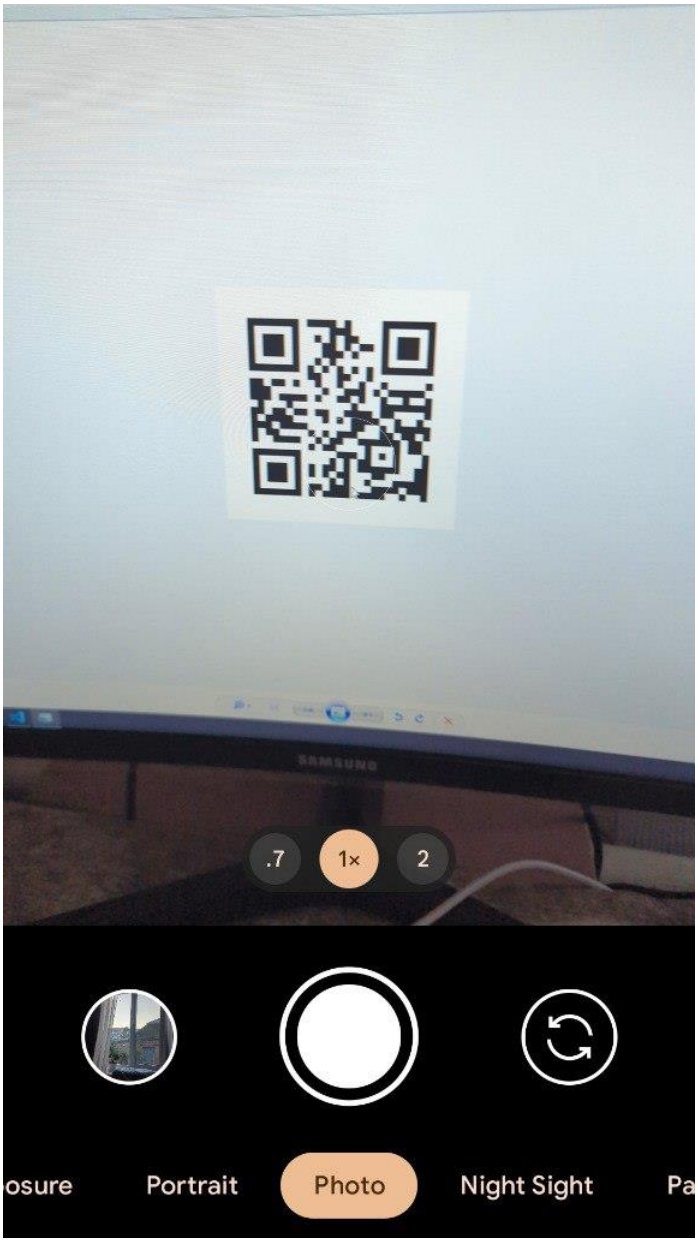

Рисунок 3.17 – Перевірка створеного QR-коду

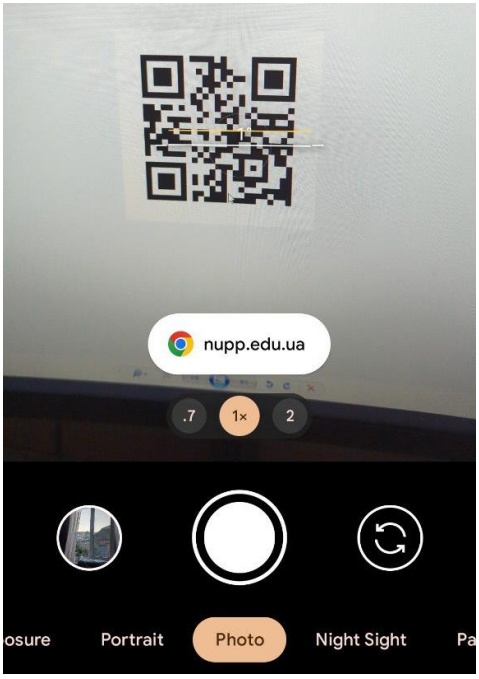

Рисунок 3.18 – Перевірка створеного QR-коду

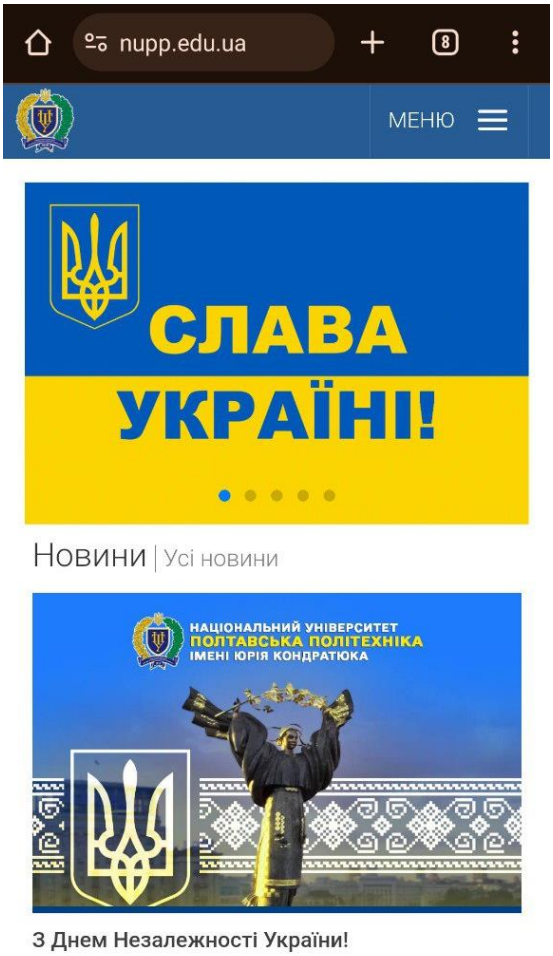

Рисунок 3.19 – Перевірка створеного QR-коду

3.2. Тестування програмного забезпечення

Тестування програмного забезпечення вирішено проводити за допомогою чек-листа та тест-кейсів, оскільки ці два методи тестування є найбільш зрозумілими та інформативними, особливо для розробленого у ході виконання даної дипломної роботи програмного забезпечення з графічним інтерфейсом для створення й обробки QR-кодів.

Чек-лист – це список тестів, які повинні бути виконані в певному порядку. Допомагає зрозуміти, чи повністю виконано тестування і скільки тестів не пройшло. Він також допомагає формалізувати тестування окремо взятого функціоналу, розмістивши тести у вигляді списку. Порядок тестів у чек-листі може бути як суворим, так і випадковим [14].

Тест-кейс – це набір дій, які виконуються над системою, щоб визначити, чи задовольняє вона програмним вимогам і чи функціонує правильно. Мета тестового кейсу – визначити, чи працюють різні функції в системі так, як очікується, і підтвердити, що система задовольняє всім відповідним стандартам, інструкціям і вимогам замовника.

Процес написання тестового кейсу також може допомогти виявити помилки або дефекти в системі. Тестові кейси зазвичай пишуться членами команди забезпечення якості (QA) або команди тестування і можуть використовуватися як покрокові інструкції для кожного тесту системи.

Тестування починається після того, як команда розробників завершила роботу над функцією або набором функцій системи. Послідовність або набір тестових кейсів називається тестовим набором [15].

Чек-лист подано у вигляді Таблиці 3.1.

Таблиця 3.1 – Чек-лист для розробленого ПЗ

№	Перевірка	Деталі	Статус
1	Генерація QR-коду	Ввести валідний URL у поле для даних.	Перевірено
2	Генерація QR-коду	Ввести валідний текст у поле для даних.	Перевірено
3	Генерація QR-коду	Вибрати існуючу папку для збереження QR-коду.	Перевірено
4	Генерація QR-коду	Згенерувати QR-код та перевірити, чи файл успішно зберігається.	Перевірено
5	Генерація QR-коду	Перевірити, чи QR-код правильно кодує введені дані.	Перевірено
6	Валідація вводу	Залишити поле вводу даних порожнім і перевірити, чи програма відображає повідомлення про помилку.	Перевірено
7	Валідація вводу	Перевірити, чи програма приймає та коректно обробляє спеціальні символи.	Перевірено
8	Вибір папки для збереження	Вибрати існуючу папку та перевірити, чи файл успішно зберігається.	Перевірено
9	Декодування QR-коду	Вибрати валідне зображення з QR-кодом і перевірити, чи дані декодуються правильно.	Перевірено
10	Декодування QR-коду	Спробувати декодувати зображення, що не містить QR-коду	Перевірено
11	Інтерфейс користувача	Перевірити, чи інтерфейс програми є зрозумілим та інтуїтивно зрозумілим.	Перевірено
12	Інтерфейс користувача	Переконатися, що всі кнопки та елементи управління працюють відповідно до їх призначення.	Перевірено
13	Інтерфейс користувача	Перевірити, чи програма правильно обробляє скасування дій.	Перевірено
14	Продуктивність	Перевірити час генерації QR-коду при введенні великого обсягу даних.	Перевірено
15	Продуктивність	Оцінити швидкість декодування QR-коду.	Перевірено
16	Продуктивність	Переконатися, що програма не зависає при виконанні запитів.	Перевірено
17	Логування та обробка помилок	Переконатися, що програма коректно обробляє неочікувані ситуації.	Перевірено

Тест-кейси подано у вигляді Таблиць 3.2-3.10.

Таблиця 3.2 – Тест-кейс для розробленого ПЗ

Номер	**1**
Назва	Генерація QR-коду з валідним URL
Опис	Перевірити, чи програма коректно генерує QR-код для валідного URL.
Передумови	Введено валідний URL.
Кроки	1. Ввести валідний URL у поле для даних.
	2. Натиснути кнопку "Генерувати".
Очікуваний результат	QR-код генерується та зберігається у вибраній папці.
Фактичний результат	QR-код успішно згенеровано та збережено в обраній папці.

Таблиця 3.3 – Тест-кейс для розробленого ПЗ

Номер	**2**
Назва	Генерація QR-коду з валідним текстом
Опис	Перевірити, чи програма коректно генерує QR-код для валідного тексту.
Передумови	Введено валідний текст.
Кроки	1. Ввести валідний текст у поле для даних.
	2. Натиснути кнопку "Генерувати".
Очікуваний результат	QR-код генерується та зберігається у вибраній папці.
Фактичний результат	QR-код успішно згенеровано та збережено в обраній папці.

Таблиця 3.4 – Тест-кейс для розробленого ПЗ

Номер	**3**
Назва	Вибір існуючої папки для збереження QR-коду
Опис	Перевірити, чи програма дозволяє вибрати існуючу папку для збереження.
Передумови	Вибрана існуюча папка.
Кроки	1. Натиснути кнопку для вибору папки.
	2. Вибрати існуючу папку.
Очікуваний результат	Програма дозволяє вибрати папку без помилок.
Фактичний результат	Програма успішно дозволила вибрати існуючу папку.

Таблиця 3.5 – Тест-кейс для розробленого ПЗ

Номер	**4**
Назва	Збереження QR-коду
Опис	Перевірити, чи QR-код успішно зберігається у вибраній папці.
Передумови	QR-код успішно згенеровано.
Кроки	1. Згенерувати QR-код.
	2. Перейти до вибраної папки.
Очікуваний результат	Файл з QR-кодом знаходиться у вибраній папці.
Фактичний результат	Файл з QR-кодом успішно збережено у вибраній папці.

Таблиця 3.6 – Тест-кейс для розробленого ПЗ

Номер	**5**
Назва	Виведення повідомлення про помилку при пустому полі вводу
Опис	Перевірити, чи програма виводить повідомлення про помилку при залишенні поля вводу порожнім.
Передумови	Поле вводу даних порожнє.
Кроки	1. Натиснути кнопку "Генерувати".
Очікуваний результат	Виводиться повідомлення про помилку, що поле вводу не може бути порожнім.
Фактичний результат	Виводиться коректне повідомлення про помилку.

Таблиця 3.7 – Тест-кейс для розробленого ПЗ

Номер	**6**
Назва	Валідація невалідних даних
Опис	Перевірити, чи програма коректно обробляє невалідні дані.
Передумови	Введено невалідні дані.
Кроки	1. Ввести невалідні дані у поле для даних.
	2. Натиснути кнопку "Генерувати".
Очікуваний результат	Виводиться повідомлення про помилку, що дані невалідні.
Фактичний результат	Програма вивела повідомлення про невалідні дані.

Таблиця 3.8 – Тест-кейс для розробленого ПЗ

Номер	**7**
Назва	Декодування валідного QR-коду
Опис	Перевірити, чи програма коректно декодує валідний QR-код.
Передумови	Збережено валідне зображення з QR-кодом.
Кроки	1. Вибрати зображення з QR-кодом.
	2. Натиснути кнопку "Декодувати".
Очікуваний результат	Програма коректно декодує дані з QR-коду.
Фактичний результат	Дані з QR-коду успішно декодовано.

Таблиця 3.9 – Тест-кейс для розробленого ПЗ

Номер	**8**
Назва	Обробка зображення без QR-коду
Опис	Перевірити, як поводиться програма при спробі декодування зображення без QR-коду.
Передумови	Вибрано зображення, що не містить QR-коду.
Кроки	1. Вибрати зображення без QR-коду.
	2. Натиснути кнопку "Декодувати".
Очікуваний результат	Виводиться повідомлення про помилку, що QR-код не знайдено.
Фактичний результат	Коректне повідомлення про помилку виведено.

Таблиця 3.10 – Тест-кейс для розробленого ПЗ

Номер	**9**
Назва	Вибір неіснуючої папки для збереження
Опис	Перевірити, чи програма виводить повідомлення про помилку при спробі вибрати неіснуючу папку.
Передумови	Вибрана неіснуюча папка.
Кроки	1. Натиснути кнопку для вибору папки.
	2. Спробувати вибрати неіснуючу папку.
Очікуваний результат	Виводиться повідомлення про помилку.
Фактичний результат	Виводиться коректне повідомлення про помилку.

Таким чином, розроблена програма успішно виконує свої основні

функції, дозволяючи користувачам генерувати QR-коди з введених даних та зчитувати інформацію з уже існуючих QR-кодів. Завдяки зручному графічному інтерфейсу, користувачі можуть легко взаємодіяти з програмою, виконуючи всі необхідні операції. Створені QR-коди відповідають стандартам і можуть бути зчитані будь-яким пристроєм, що підтримує сканування QR-кодів, забезпечуючи надійність і універсальність використання цього рішення.

3.3. Висновки до розділу III

У розділі були розглянуті важливі етапи реалізації програмного забезпечення та його тестування, що дозволили забезпечити працездатність і надійність розробленого продукту.

На етапі реалізації програми були детально описані всі кроки, пов'язані з програмною реалізацією функціоналу, включаючи генерацію QR-кодів, зчитування їх, а також взаємодію з користувачем через графічний інтерфейс. Окрему увагу було приділено використанню бібліотек, що забезпечують обробку даних і взаємодію з QR-кодами, таких як бібліотека qrcode. Описано, як побудовані основні функції програми, а також розглянуті принципи організації коду для забезпечення зручності та ефективності роботи.

Тестування програмного забезпечення є важливим етапом для перевірки коректності виконання всіх функцій і можливих помилок у програмі. У процесі тестування було виявлено та виправлено деякі помилки, що дозволило поліпшити стабільність і надійність програми. Було протестовано як окремі модулі, так і програму в цілому, що дозволило забезпечити її відповідність вимогам до функціональності та безпеки. Також проводились тести на сумісність і продуктивність, що дозволило визначити можливі ризики і оптимізувати роботу програми.

Таким чином, даний розділ охоплює основні етапи розробки та тестування програмного забезпечення для створення й обробки QR-кодів. Під час реалізації програми було застосовано сучасні технології та бібліотеки, а

етап тестування дозволив забезпечити її стабільність і безпеку, що є важливими аспектами для успішного використання продукту в реальних умовах.

ВИСНОВКИ

У ході дипломної роботи була розроблена програма з графічним інтерфейсом для кодування й зчитування QR-кодів, що успішно виконує свої основні функції, дозволяючи користувачам генерувати QR-коди з введених даних та зчитувати інформацію з уже існуючих QR-кодів.

Завдяки зручному графічному інтерфейсу, користувачі можуть легко взаємодіяти з програмою, виконуючи всі необхідні операції. Створені QR-коди відповідають стандартам і можуть бути зчитані будь-яким пристроєм, що підтримує сканування QR-кодів, забезпечуючи надійність і універсальність використання цього рішення.

Крім цього, були розглянуті теоретичні питання, що безпосередньо стосуються обраної теми, серед них:

1. Основи QR-кодів;
2. Захист даних при використанні QR-кодів;
3. Обробка помилок QR-кодів;
4. Вміст QR-кодів;
5. Структура QR-коду;
6. Шаблони для QR-кодів;
7. Ризики використання QR-кодів.

Мета дипломної роботи, яка полягала у створенні програмного продукту, що об'єднує функції генерації та декодування QR-кодів у зручній і зрозумілій формі для користувачів різного рівня підготовки, досягнута.

У ході роботи було реалізовано програмне забезпечення, яке вирішує технічні завдання, пов’язані з генерацією та обробкою QR-кодів. Крім того, створено сучасний інтерфейс, що забезпечує простоту використання та доступність функціоналу для широкого кола користувачів. Отриманий результат дозволяє ефективно використовувати розроблене програмне забезпечення у різних сферах, які потребують роботи з QR-кодами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Features and Applications of QR Codes [Електронний ресурс] – Режим доступу до ресурсу: https://www.researchgate.net/publication/360360156_Features_and_Applications_of_QR_Codes

2. Methods for data validation using QR codes [Електронний ресурс] – Режим доступу до ресурсу: https://www.researchgate.net/publication/376669586_Methods_for_data_validation_using_QR_codes

3. QR Code Error Correction [Електронний ресурс] – Режим доступу до ресурсу: https://www.qrstuff.com/blog/general/qr-code-error-correction

4. Фуцзянь Чжонган Електронні технології. «Application of two-dimensional code technology in the field of anti-counterfeiting and standard proposal». China Standard Word, 2001.

5. Different Types of QR Codes for Your Marketing Goal [Електронний ресурс] – Режим доступу до ресурсу: https://www.qr-code-generator.com/qr-code-marketing/different-types-of-qr-codes/

6. How Does a QR Code Encode Data? [Електронний ресурс] – Режим доступу до ресурсу: https://www.qr-code-generator.com/blog/qr-code-encoding

7. An Introduction to QR Code Technology [Електронний ресурс] – Режим доступу до ресурсу: https://www.researchgate.net/publication/318125149_An_Introduction_to_QR_Code_Technology

8. An Introductory Tutorial For Encoding QR Codes [Електронний ресурс] – Режим доступу до ресурсу: https://zavier-henry.medium.com/an-introductory-walkthrough-for-encoding-qr-codes-5a33e1e882b5

9. What are the advantages of using Visual Studio Code [Електронний ресурс] – Режим доступу до ресурсу: https://eitca.org/web-development/eitc-wd-jsf-javascript-fundamentals/introduction-eitc-wd-jsf-javascript-

fundamentals/history-of-javascript/examination-review-history-of-javascript/what-are-the-advantages-of-using-visual-studio-code-as-a-code-editor-for-javascript-development

10. Why VS Code remains a developer favorite, year after year [Електронний ресурс] – Режим доступу до ресурсу: https://shiftmag.dev/vs-code-171

11. Benefits of Python over Other Programming Languages [Електронний ресурс] – Режим доступу до ресурсу: https://www.invensis.net/blog/benefits-of-python-over-other-programming-languages

12. Python Applications – Unleash the power of Python [Електронний ресурс] – Режим доступу до ресурсу: https://data-flair.training/blogs/python-applications

13. Generate QR Codes with Python: An Easy Guide [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/@rahulmallah785671/create-qr-code-by-using-python-2370d7bd9b8d

14. Testing Docs Templates – Checklist [Електронний ресурс] – Режим доступу до ресурсу: https://strongqa.com/qa-portal/testing-docs-templates/checklist

15. Definition test case [Електронний ресурс] – Режим доступу до ресурсу: https://www.techtarget.com/searchsoftwarequality/definition/test-case

16. T. Sriram, V.K. Rao, «Застосування технології штрихкодів в автоматизованих системах зберігання та пошуку», Протоколи конференції з промислової електроніки, Тайбей, 1996.

17. ISO/IEC 18004:2000. «Інформаційні технології – Автоматична ідентифікація та технології збору даних – Символіка штрихкоду – QR Code», 2000.

18. Phaisarn Sutheebanjard, Wichian Premchaiswadi, «Генератор QR-кодів», IEEE, 8-ма Міжнародна конференція з ІКТ та інженерії знань, 2019.

19. Y. Yan, H.W. Liu, «Дослідження та застосування технологій кодування і декодування QR-кодів», Університет науки і технологій, Пекін.

20. R. Dorado, E. Torress, C. Rus, «Мобільне навчання: Використання QR-кодів для створення навчальних матеріалів», IEEE - Технології, застосовані в навчанні електроніки (TAEE), Севілья, Іспанія, 2016.

21. Dong-Hee Shin, Jaemin Jung, Byeng-Hee Chang, «Психологія QR-кодів: Перспектива користувацького досвіду», Science Direct, Computers in Human Behavior, том 28, 2012.

22. Sumit Tiwari, Sandeep Sahu, «Новий підхід до виявлення фальсифікації OMR-аркушів за допомогою зашифрованого QR-коду», IEEE - Міжнародна конференція з обчислювального інтелекту та досліджень обчислень, Коїмбатор, Індія, 2014.

ДОДАТОК А
ВИХІДНИЙ КОД ОСНОВНИХ КОМПОНЕНТІВ

Файл project.py

```
import qrcode
from pyzbar.pyzbar import decode
from PIL import Image
import tkinter as tk
from tkinter import filedialog, messagebox

# Функція для генерації QR-коду
def generate_qr_code(data, folder):
    qr = qrcode.QRCode(
        version=1,
        error_correction=qrcode.constants.ERROR_CORRECT_L,
        box_size=10,
        border=4,
    )

    qr.add_data(data)
    qr.make(fit=True)
    img = qr.make_image(fill_color="black", back_color="white")

    filename = f"{folder}/example_qr.png"
    img.save(filename)

    messagebox.showinfo(language_strings['success'],
f"{language_strings['qr_generated']} {filename}")

# Функція для розшифрування QR-коду з зображення
def decode_qr_code(image_path):
    img = Image.open(image_path)
    decoded_objects = decode(img)

    for obj in decoded_objects:
        return obj.data.decode('utf-8')

    return language_strings['no_qr_found']

# Функція для відкриття діалогового вікна та вибору шляху до папки
def select_folder():
    folder_path = filedialog.askdirectory(title=language_strings['select_folder'])
    folder_entry.delete(0, tk.END)
    folder_entry.insert(0, folder_path)

# Функція для вибору файлу та розшифрування QR-коду
def select_and_decode_qr():
    image_path =
filedialog.askopenfilename(title=language_strings['select_qr_image'],
filetypes=[("PNG files", "*.png")])
    if image_path:
        decoded_data = decode_qr_code(image_path)
        messagebox.showinfo(language_strings['decoded_data'],
f"{language_strings['decoded_qr_data']}: {decoded_data}")

# Функція для обробки генерації QR-коду
def handle_generate():
```

```
    data = data_entry.get()
    folder = folder_entry.get()
    if data and folder:
        generate_qr_code(data, folder)
    else:
        messagebox.showwarning(language_strings['input_error'],
language_strings['enter_data_folder'])

# Функція для центрування вікна на екрані
def center_window(window):
    window.update_idletasks()
    width = window.winfo_width()
    height = window.winfo_height()
    x = (window.winfo_screenwidth() // 2) - (width // 2)
    y = (window.winfo_screenheight() // 2) - (height // 2)
    window.geometry(f'{width}x{height}+{x}+{y}')

# Функція для оновлення тексту на основі обраної мови
def update_language(language):
    global language_strings
    if language == "en":
        language_strings = {
            "enter_data": "Enter Data:",
            "save_to_folder": "Save to Folder:",
            "browse": "Browse",
            "generate_qr": "Generate QR Code",
            "decode_qr": "Decode QR Code",
            "success": "Success",
            "qr_generated": "QR code generated and saved as",
            "select_folder": "Select Folder",
            "select_qr_image": "Select QR Code Image",
            "decoded_data": "Decoded Data",
            "decoded_qr_data": "Decoded QR code data",
            "input_error": "Input Error",
            "enter_data_folder": "Please enter data and select a folder.",
            "no_qr_found": "No QR code found",
            "language": "Language: English",
            "switch_language": "Switch to Ukrainian"
        }
    else:
        language_strings = {
            "enter_data": "Введіть дані:",
            "save_to_folder": "Зберегти в папку:",
            "browse": "Огляд",
            "generate_qr": "Створити QR код",
            "decode_qr": "Розшифрувати QR код",
            "success": "Успіх",
            "qr_generated": "QR код створено та збережено як",
            "select_folder": "Виберіть папку",
            "select_qr_image": "Виберіть зображення QR коду",
            "decoded_data": "Розшифровані дані",
            "decoded_qr_data": "Розшифровані дані QR коду",
            "input_error": "Помилка введення",
            "enter_data_folder": "Будь ласка, введіть дані та виберіть папку.",
            "no_qr_found": "QR код не знайдено",
            "language": "Мова: Українська",
            "switch_language": "Переключити на англійську"
        }

    # Оновлення тексту всіх елементів
    label_data.config(text=language_strings['enter_data'])
    label_folder.config(text=language_strings['save_to_folder'])
```

```
    button_browse.config(text=language_strings['browse'])
    button_generate.config(text=language_strings['generate_qr'])
    button_decode.config(text=language_strings['decode_qr'])
    button_language.config(text=language_strings['switch_language'])
    root.title(f"{language_strings['language']}")

# Налаштування інтерфейсу
root = tk.Tk()
root.title("QR Code Generator and Decoder")

# Встановлення ширини вікна
root.geometry("550x300")

# Налаштування фону вікна та стилів
root.configure(bg="#2E4053")

style_options = {
    "bg": "#2E4053",
    "fg": "white",
    "font": ("Helvetica", 12)
}

# Поле для введення даних
label_data = tk.Label(root, text="", **style_options)
label_data.grid(row=0, column=0, padx=10, pady=10, sticky="e")
data_entry = tk.Entry(root, width=50)  # Збільшення ширини за необхідністю
data_entry.grid(row=0, column=1, padx=10, pady=10)

# Вибір папки
label_folder = tk.Label(root, text="", **style_options)
label_folder.grid(row=1, column=0, padx=10, pady=10, sticky="e")
folder_entry = tk.Entry(root, width=50)  # Збільшення ширини за необхідністю
folder_entry.grid(row=1, column=1, padx=10, pady=10)
button_browse = tk.Button(root, text="", command=select_folder, bg="#1ABC9C",
fg="white")
button_browse.grid(row=1, column=2, padx=10, pady=10)

# Кнопки для генерації та розшифрування
button_generate = tk.Button(root, text="", command=handle_generate, bg="#1ABC9C",
fg="white")
button_generate.grid(row=2, column=1, padx=10, pady=20)
button_decode = tk.Button(root, text="", command=select_and_decode_qr, bg="#E74C3C",
fg="white")
button_decode.grid(row=3, column=1, padx=10, pady=10)

# Кнопка для перемикання мови
button_language = tk.Button(root, text="", command=lambda: update_language("uk" if
language_strings['language'] == "Language: English" else "en"), bg="#3498DB",
fg="white")
button_language.grid(row=4, column=1, padx=10, pady=10)

# Ініціалізація мови до англійської та оновлення тексту інтерфейсу
update_language("en")

# Центрування вікна на екрані
center_window(root)

root.mainloop()
```