Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра інформатики факультету інформатики

Розробка корпоративного QR-файлообмінника

Текстова частина до курсової роботи
за спеціальністю „Інженерія програмного забезпечення” 6.050103

Керівник курсової роботи
к.ф.-м.н. Ющенко Ю.О
_____
(підпис)
“____” _____ 2020 р.
Виконав студент
Приймак О.Я.
“____” _____ 2020 р.

Київ 2020

Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра інформатики факультету інформатики

ЗАТВЕРДЖУЮ

Зав.кафедри інформатики,

Доцент., к. ф.-м. н. С.С.Гороховський

(підпис)

„____”_____2019 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на курсову роботу

студенту Приймаку Олегу Ярославовичу факультету інформатики 4-го курсу

ТЕМА Розробка корпоративного QR-файлообмінника

Зміст ТЧ до курсової роботи:

1. Індивідуальне завдання
2. Календарний план
3. Анотація
4. Вступ
5. Обґрунтування корисності та доцільності реалізації файлообмінника
6. Розробка програми
7. Огляд реалізації та особливостей використаних технологій
8. Висновки
9. Список використаних джерел

Дата видачі „___” _____ 2019 р. Керівник _____

(підпис)

Завдання отримав _____

(підпис)

Тема: <u>Розробка корпоративного QR-файлообмінника</u>

Календарний план виконання роботи:

№ п/п	Назва етапу курсового проекту (роботи)	Термін виконання етапу	Примітка
1.	Отримання завдання на курсову роботу.	17.10.2019	
2.	Огляд існуючих методів(додатків).	17.11.2019	
3.	Написання першого розділу.	15.01.2020	
4.	Розробка застосунку.	10.03.2020	
5.	Написання останнього розділу.	02.04.2020	
6.	Створення слайдів для доповіді та написання доповіді.	05.04.2020	
7.	Корегування роботи згідно із зауваженнями керівника.	15.04.2020	

Студенту Приймаку О.Я.

Керівник Ющенко Ю.О.

“_____”_____

ЗМІСТ

Анотація

У роботі розглянуто принципи роботи принципи роботи Web, Native та Progressive застосунків, їх популярність, переваги і недоліки. Розроблено застосунок QR – файлообмінник для швидкого та зручного обміну інформацією.

ВСТУП

Кожного дня більшість осіб стикаються з величезними обсягами різноманітної інформації як на роботі, так і в побуті та дозвіллі. Людям потрібна інформація задля прийняття правильних, точних, найкращих, найефективніших та найдоцільніших рішень. Для обміну інформацією між особами створюються різноманітні файлообмінники та месенджери. Передача, зберігання гігабайтів пам'яті, листування, перегляд медіа-файлів для багатьох є звичною справою. Питання покращення зручності використання безпечного, надійного збереження та передачі інформації наразі набуло актуальності.

Виникає потреба у зручних та надійних сервісах обміну та збереження інформації. Широкого використання для підтвердження прав доступу та прав на вчинення певних дій набув код швидкого відгуку (QR-код). Простота та швидкість сканування кодів(навіть за допомогою мобільних пристроїв) широко використовується для кодування URL, веб-сторінок, рекламних оголошень, email повідомлень, персональних даних, географічних координат, інформації про залізничні квитки, підтверджень виконання банківських транзакцій, аутентифікації при вході в акаунти Інтернет сервісів тощо. Найбільш широкого використання ці коди набули в Японії та поступово їх використання поширюється в інших країнах.

Метою роботи є реалізація зручного, надійного та швидкого файлообмінника обміну даними між пристроями з різними операційними системами з високим захистом інформації від несанкціонованого використання. До функціональних вимог вказаного файлообмінника належить також захищений інструмент надання різних рівнів прав доступу окремим особам та чи групам осіб. Тобто файлообмінник має забезпечити потреби довільних груп користувачів: . політичні партії, громадські об`єднання, колективи працівників організацій та установ, груп спільних інтересів тощо. Саме тому при проектуванні можливостей файлообміннику враховано корпоративні потреби. За допомогою спеціалізованої технології на платформі розробки мобільних та веб-

застосунків - firebase, передбачено надання можливості користувачам зберігати з’єднання сервером, на якому зберігаються дані, які одночасно використовуються у режимі коригування.

Текстова частина курсової роботи складається з двох розділів. У першому розглядаються особливості застосування QR-кодів, проведено аналіз переваг та недоліків існуючих методів транспортування даних, web, native та progressive застосунків. У другій частині детальний опис розробленого додатку та описано особливості використання при реалізації інструментів та технологій.

РОЗДІЛ 1: Обґрунтування корисності та доцільності реалізації файлообмінника

Сьогодні в усьому світі смартфонами та іншими мобільними пристроями користуються приблизно 3 млрд осіб та кількість користувачів постійно та стрімко збільшується. Більшість з них проводять перед екранами смартфонів понад 3 години в день. Компанії намагаються надати користувачам більше корисних можливостей, вдосконалюють додатки для зручного, захищеного та надійного сумісного використання даних пов'язаними між собою користувачами. При цьому проблема розробки застосунків спільного використання даних для різних платформ залишається актуальною. Функціональні можливості додатків для різних систем Android, iOS та інших мають збігатися. Вельми успішно вказану проблему дозволяють вирішити крос-платформні веб-застосунки, які мають працювати однаково під різними операційними системами та у браузерах, які підтримують – гібрид між «нативним», та веб застосунком

1.1 Застосування та можливості QR (Quick Response) кодів

1.1.1 Опис QR коду

Штриховий код – це спосіб запису даних, який є зручним для зчитування сканерами (або електронними пристроями). Найчастіше використовуються штрихові коди UPC. Які складаються з смуг різної товщини. Вони містять ідентифікатори товарів.

Рисунок 1.1.1 Приклад UPC Штрихкод

Вперше штрих-код був розроблений інженером на Пенсільванській залізниці. У 50-роках 20 століття перші штрих-коди були використані Давидом Коллінзом для перерахунку та сортування вагонів залізниці. Першим прототипом штрихового коду був прямокутник півметра довжиною з червоними і синіми смугами. Пристрій для зчитування міг правильно сканувати код навіть при великій швидкості вагона. З часом розмір пристрою для сканування зменшився, як і розмір самого коду. Тоді ж виникла і ідея маркувати товари за допомогою цього коду. В 1973 році була створена організація Universal Product Code (UPC). Почалася ера застосування кодів у торгівлі, виробництві.

QR-код, або ж quick response code (швидкий відгук) – це є матричний (штриховий) код розроблений японською компанією Denso-Wave в 1994 році.

Переваги QR-штрихування:

1) цей код є двовимірним тому може нести в собі до чотирьох тисяч символів. Користувачі можуть зашифрувати не лише якесь коротке повідомлення, як номер товару або продукту на виробництві, а й довгі повідомлення, тексти, посилання, рекламні оголошення. Також є можливість зашифрувати й зображення. Правда є обмеження в 4КБ, цього для зображень недостатньо

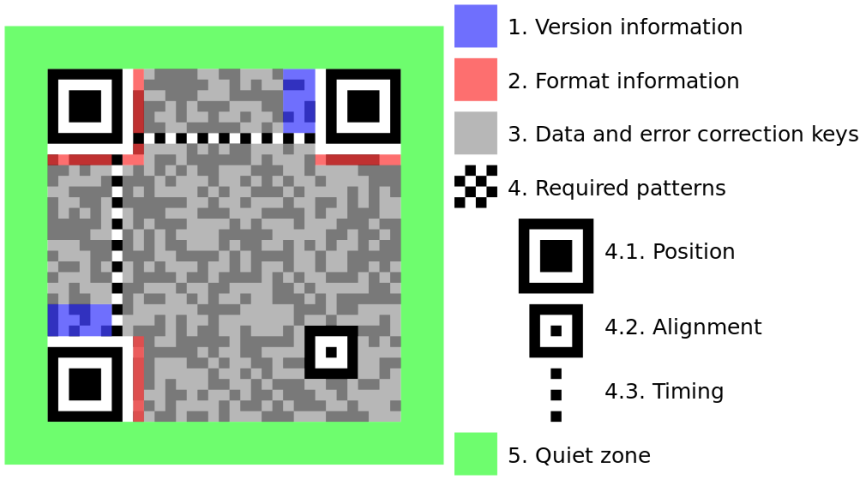

Рисунок 2.1.2 Структура QR-коду з позначеним функціональними елементами

2) основною перевагою QR-кодів є легке розшифровування за допомогою сканувального обладнання. Просканувати код можна і за допомогою фотокамери мобільного пристрою. Також існує багато застосунків для сканування коду з веб-камери персонального комп'ютера, інших пристроїв. Старий штрих-код (UPC) сканують променем
3) величезною перевагою цього кодування над попереднім є його широке використання і перспективність. QR-коди з кожним днем з'являються навколо нас все частіше й частіше. Великі японські компанії започаткували впровадження сканерів для розшифрування кодів (саме в Японії вони найпопулярніші). Тому не дивно зараз побачити цей штрих-код в оголошеннях на стовпах, стінах. Багато рекламних компаній переведені на використання цього коду

Рисунок 3.1.2 Структура QR-коду з позначеним функціональними елементами

1.1.2 Застосування QR-кодів

Найчастіше штрих-код використовують для розміщення в рекламі, пошуку інформації, обміну контактами, реєстрації в сервісах, підключення до Wi-Fi, переходу за посиланнями. Також коди використовуються в туризмі (наприклад для позначення туристичних об’єктів, географічних координат).

Є й більш унікальні приклади використання цього штрихування. В Литві з’явився сайт «Віртуальне кладовище», де для ідентифікації використовується код. QR коди наносять на продукти, архітектурні споруди. В країнах Азії (у нас теж, у магазинах мережі ВелМарт) оплата здійснюється за допомогою QR, люди презентують подарунки, перераховують милостиню жебракам. Нещодавно в Києві в метро перейшли на оплату за допомогою цього штрихування.

1.2 Аналіз методів транспортування даних

Файлообмінники існують спеціально для обміну файлами. Файлообмінником називається сервіс, що надає користувачеві місце для зберігання файлів та цілодобовий доступ до них через Web. Користувач один раз завантажує файл і

він доступний для всіх користувачів за посиланням. Його можна публікувати на форумах, пересилати поштою або поширювати в соціальних мережах. Але у файлообмінників є ряд недоліків. Основними з яких являються максимальний розмір файлу, швидкість завантажування, необхідність реєстрації. Розглянемо основні методи транспортування даних.

1) додатки Wi-Fi

   Існують різні мобільні програми, наприклад WebSharing. Застосунок запускає на пристрої сервер і дає парольний доступ до носія із локальної мережі через веб-інтерфейс. Запуск сервера відбувається всього в два етапи – запуск застосунку і уже після натиску кнопки «Start» сервер готовий за IP-адресою і портом для доступу, що відобразиться на екрані. Інтерфейс програми не підтримує багатомовність, але цього цілком достатньо, так як застосунок потрібен лише для запуску сервера. З цієї причини налаштувань не багато, але велика частина доступна у платній версії: вказати постійний пароль для з’єднання, блокування відключення екрану під час роботи програми, можливість вказати e-mail для відправлення інформації для доступу, налаштування мережі. Основним недоліком є необхідність того, щоб обидва пристрої були в одній мережі Wi-Fi

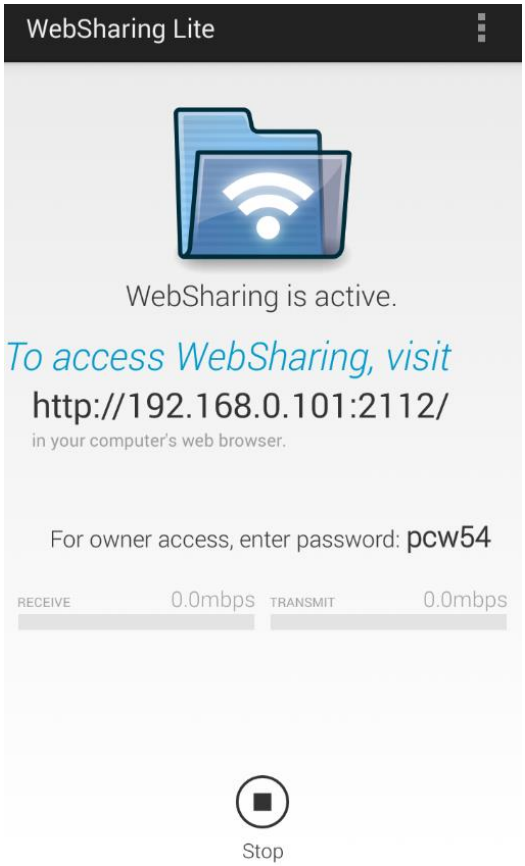

Рисунок 4.2.1 Приклад серверу WebSharing

2) хмарні послуги

   Google Drive, MS OneDrive та інші сервіси дозволяють завантажувати файли в хмару, після чого ви можете надіслати посилання на файл на інший пристрій. Недоліками цього методу є необхідність створення облікового запису, а також користувачу все одно потрібно буде якось перенести посилання з одного пристрою на інший

3) Bluetooth

   Це бездротова технологія передачі даних між двома пристроями, що знаходяться на близькій відстані. Переваги та недоліки технології добре відомі тим, хто її використовує. Мережі в яких працює Bluetooth відносно безпечні, непогано захищені від крадіжки даних. Перевага технології в тому, що для передачі даних не потрібен Інтернет і технологія є бездротовою, не вимагає наявності кабелів і проводів. Також в останніх версіях швидкість транспортування даних досягає 1 Мбіт/сек. Основним

ж недоліком є незручне підключення пристроїв. Пристрої повинні бути безпосередньо близько між собою. В відкритій місцевості і без наявних радіоперешкод технологія може ефективно використовуватися на відстані до 100 метрів. В основному ж радіус дії всього кілька метрів. Є також проблема з різноманітністю версій, виникає проблема сумісності версій. Найбільш поширена ця технологія серед користувачів мобільних телефонів. Майже всі сучасні смартфони оснащені Bluetooth. Також технологію використовують для планшетів, ноутбуків, плеєрів, тощо. Технологія не є особливо енергозатратною, тому використовується для прикладу для медичних приладів

4) USB-кабель

   Пристрій можна підключити до комп'ютера за допомогою кабелю USB. Перевагою є дуже висока швидкість передачі даних, не потрібен Інтернет. Для транспортування потрібен кабель, але він не завжди знаходить поруч. Неможливо підключити два мобільних пристрої

5) електронна пошта, месенджери

   Файли також можна передавати через месенджери. Метод є досить простим у використанні, але для транспортування даних користувачам необхідно пройти реєстрацію і щоб вони знаходилися в контактах один одного. Також месенджери з метою оптимізації та збільшення швидкості передачі можуть погіршити якість зображень

6) флеш-накопичувач

   Носій інформації, що використовує флеш-пам'ять для збереження даних та підключається до комп'ютера чи іншого пристрою через USB-порт. Флеш накопичувачі зазвичай підтримують перезапис. Основну популярність здобули у зв'язку з тим, що вони дуже компактні, легкі і мають великий об'єм пам'яті. Основне призначення – зберігання й перенесення файлів та обмін між ними, резервне копіювання, завантаження операційних систем тощо. Флеш-носій потрібно завжди носити із собою, не в усіх пристроях є USB-порт

Рисунок 5.2.2 Приклад дизайну флеш-накопичувача

1.3 Порівняння Web та Native застосунків

Web-застосунок – клієнт-серверна прикладна програма, в якій клієнтом виступає браузер, а сервером – web-сервер. Вся логіка веб-застосунку розміщена на сервері, а браузер виконує функцію відображення інформації клієнтові і передачі даних від клієнта до сервера. Основною перевагою такого підходу є те, що клієнти не залежать від конкретної операційної системи користувача, тому застосунок є багатоплатформовим. Як наслідок веб-застосунки набули значної популярність на початку 20 століття.

Замість того, щоб писати різні версії для Android, Windows та інших операційних систем застосунок створюється для платформи один раз і розгортається на ній. Застосунок отримує від клієнта запит, на сервері виконує обчислення, звертається до бази даних і таке інше, і повертає результат по протоколу HTTP/HTTPS – веб-сторінку. Застосунок може бути і сам клієнтом інших сервісів. Швидкість розробки таких застосунків висока. Розробники можуть пропонувати застосунок без підтвердження магазинів застосунків (Play Market).

Проте є декілька недоліків:

1) можливість користувача змінювати налаштування браузера, такі як зміна розміру шрифтів, кольорів та інших може перешкоджати правильній роботі веб-застосунку. Розробники реалізують клієнтський HTML, CSS по-різному, тому виникають проблеми при можливих доопрацюваннях та підтримці
2) у застосунку немає доступу до пристрою користувача. Незважаючи на те, що іноді це було б зручно. Це обмежує деякі функції, які використовуються в «нативних» застосунках для більш персонального використання
3) виникає проблема з пошуком застосунку, тому що немає магазинів застосунків, як у випадку з «нативними» Apple App Store та Play Store.
4) доступ до мережі Інтернет є не завжди доступним. Деякі застосунки не є оптимізованими під погане з'єднання, розмір екрану користувача. Поруч з тим використовування Інтернет-з'єднання значно зменшує контроль за безпекою

«Нативний» застосунок – це мобільна прикладна програма, що створюється для окремої платформи і безпосередньо встановлюється на пристрій користувача, займаючи при цьому певний об'єм пам'яті. Термін «нативний застосунок» часто використовується в контексті мобільної розробки, тому що мобільне програмне забезпечення часто пишеться так, щоб програма могла працювати на певній операційній платформі. Такі застосунки користувачі завантажують через магазини застосунків своєї операційної системи. Apple App Store для iOS та Play Store для Android.

Основними перевагами таких застосунків є доступ до файлової системи та самого пристрою, тому вони мають доступ до апаратної частини пристрою, тобто можуть використовувати в своєму функціоналі камеру, мікрофон, геолокацію, плеєр, книгу контактів та інше. Такі застосунки оптимізовані під конкретні операційні системи, тому можуть працювати правильно і швидко.

1.4 Прогресивні веб-застосунки (PWA)

Поняття Progressive Web Application давно на слуху, з початку 2018 року застосунки цього типу підтримуються всіма основними браузерами, але технологія не є особливо популярною, незважаючи на їх очевидні переваги. Progressive Web Application (PWA) – це веб-застосунок, створений с використанням певних технологій для досягнення заданих цільових показників.

Цільові показники це:

1) надійність (Reliable) – застосунок завантажується і показується зразу, незважаючи на статус і якість мережевого з'єднання.
2) швидкість (Fast) – обмін даними мережею відбувається швидко, інтерфейс користувача плавний і чутливий.
3) привабливість (Engaging) – робота з застосунком для користувача має бути комфортною і приємною.

За думкою Google самі ці показники відрізняють веб та «нативні» застосунки. Тобто розробнику даються інструменти (Push Notifications, Service Worker і таке інше) і йому потрібно досягти цілей: сайт повинен працювати на слабкому з'єднанні, бути швидкими в завантаженні, працювати без мережі за необхідності. Внутрішньо PWA схожі на «нативні» застосунки. Всі основні ресурси застосунку можна зберігати на клієнтській частині, по мережі передається лише змінний контент.

Розглянемо основні частини PWA:

1) Web App manifest

   Файл JSON, що описує для браузера назву застосунку, іконку в різних розширеннях, те як застосунок буде виглядати і таке інше. Це дозволяє встановити застосунок на домашній екран застосунку.
2) HTTPS

Для PWA необхідно, щоб всі ресурси сайту передавалися через протокол HTTPS.

3) Service Worker

Основним у роботі PWA є Service Worker (SW). Це прокладка між фронт і бек (клієнтською і серверною) частинами застосунку. Всі запити браузера користувача проходять через Service Worker. Розділення сайту на дві частини дозволило зробити перехід з web до прогресивного застосунку максимально простим. Такий застосунок не залежить від якихось фреймворків, це чистий JavaScript. Хоча є можливість використовувати уже готові бібліотеки для генерації коду. З пам'яті сховища в SW є доступ до Cache Storage для веб-ресурсів та IndexDB для даних. JavaScript файл для SW легко пишеться і без використання готових бібліотек, що дуже важливо для розуміння і контролю написаного розробником функціоналу. Для підключення недостатньо просто додати файл в проект. Потрібно вказати браузеру де цей сценарій шукати: зареєструвати SW за допомогою JavaScript на кожній сторінці сайту. Головне те, що розробник має повну свободу в реалізації логіки на SW. Наприклад можна отримати запит від браузера, перевірити стан Інтернет з'єднання, якщо з'єднання задовільне – отримати результат із сервера і повернути його користувачу, додавши при цьому його в кеш. Але якщо якість з'єднання низька, або взагалі інтернету немає – повернути результат з кешу. Цей підхід називається NetworkFirst.

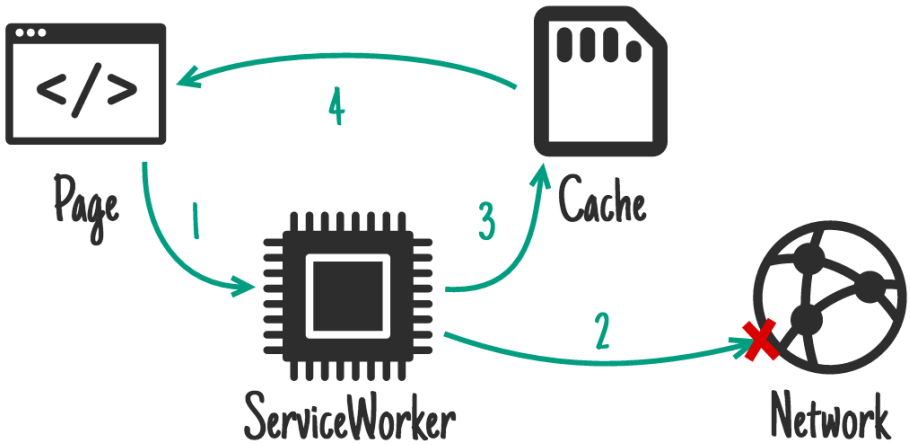

Рисунок 6.4.1 Стратегія Network First для Service Worker

Або ж реалізувати CacheFirst стратегію – якщо дані є в кеші користувача, будуть повернена інформація саме з кешу, інакше через мережу. Можна реалізувати і власну логіку. Наприклад користувач отримує дані з кеш-пам’яті тільки протягом двох днів після їх завантаження. Саме завдяки Service Worker застосунки цього типу можуть працювати без мережі. Під час першої роботи користувача з сайтом – дані завантажуються в кеш і вже при наступних відвідуваннях сайт може працювати без Інтернету.

4) Push Notifications

Понад 90% користувачів використовують в своїх застосунках PWA технології лише для надсилання сповіщень. Для того щоб зареєструвати Push Notification потрібно дозволити надсилання сповіщень на сайті і вже пізніше користувач зможе отримувати сповіщення. Звичайні користувачі мережі інтернет часто бачать на сайтах запит на підтвердження надсилання сповіщень, але це вже виглядає швидше як спам і використовується не зовсім доцільно. Для прикладу сповіщення можна використовувати як підписку.

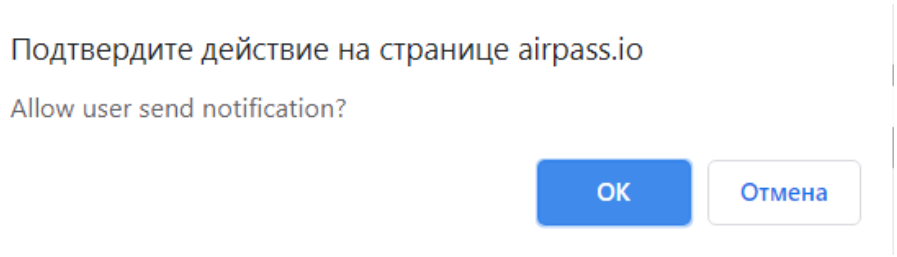

Рисунок 7.4.2 Запит на надсилання Push Notification

До сповіщення розробник може прикріпити короткий текст-опис, URL-посилання, на яке перейде користувач, що натиснув на панель сповіщення та зображення.

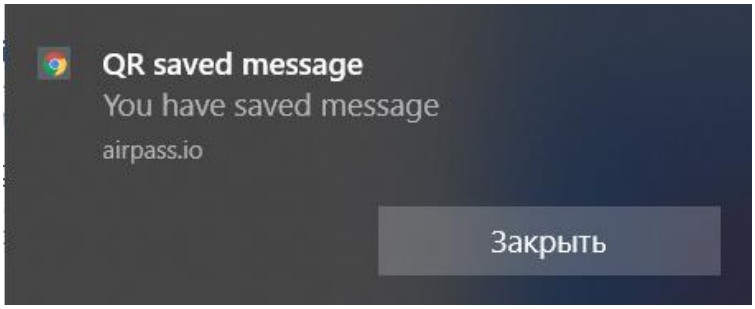

Рисунок 8.4.3 Приклад Push Notification

Застосунки даного типу є в основному заслугою корпорації Google та їх розробників. Технологія є досить новою і постійно доповнюється, в цей момент підтримується усіма сучасними браузерами. Деякий функціонал, який використовується у курсовій роботі був розроблений протягом минулого року і вже в найближчому майбутньому буде доповнюватися. Зараз прогресивні застосунки не дуже популярні, в основному їх використовують для сповіщень користувачам. А компанії досі розробляють для свого бізнесу окрім веб ще і «нативні» застосунки. Розвитку технології перешкоджає компанія Apple, якій не вигідно розвивати конкуруючу технологію, що може зменшити кількість покупок в Apple Store, та й поширити застосунки, що не будуть проходити через їх магазин застосунків.

РОЗДІЛ 2: Огляд реалізації та особливостей використаних технологій

У цьому розділі розповідається про розроблений застосунок для швидкого обміну повідомленнями, файлами. Використані технології, такі як:

1) Vue, Vue cli та Vue router для правильної роботи клієнтської частини;
2) Long polling – для встановлення постійного з’єднання між сервером і клієнтом;
3) Service Worker для коректної роботи з режимом без мережі;
4) MySQL система управління базами даних для збереження актуальної інформації;
5) Сервер на Nginx;
6) Firebase Push Notification для сповіщень користувачам;
7) PWA та Share Target;
8) I18n – технологія для багатомовності

2.1 Огляд шаблону програмування MVVM

Застосунок створено з врахуванням шаблону програмування MVVM (Model – View - ViewModel). Шаблон був представлений у 2005 році Джоном Госманом, як модифікація шаблону Presentation Model. Цей патерн використовують, якщо потрібно реалізувати зв’язування даних, тобто зміна інтерфейсу користувача змінить і дані, що лежать в основі. В класичних патернах такий як MVC та MVP зміна в інтерфейсі користувача не впливає безпосередньо на Model, а попередньо опрацьовується через Controller або Presenter.

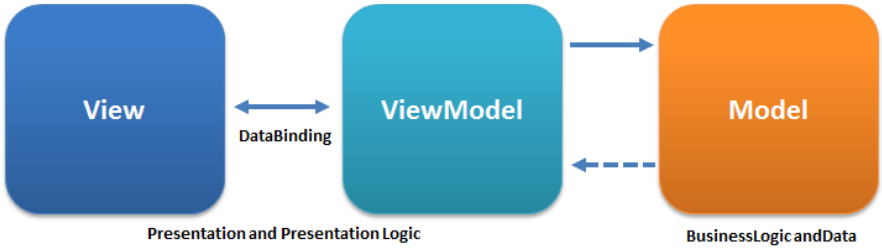

Рисунок 2.1.1 MVVM патерн

Вся логіка застосунку розділяється на три компоненти. Компонент Модель відповідає за логіку роботи із даними. У курсовій роботі вся логіка представлена у директорії Model. Логіка роботи з даними розміщена у відповідних класах, що наслідують спільного батька – клас Model. Для роботи з з'єднаннями, файлами, повідомленнями та іншим – клас QR, робота з сповіщеннями у класі WebPush, функціонал для багатомовності – клас I18n.

Компонент Представлення (View) – це графічний інтерфейс (таблиці, кнопки, зображення, індикатори і таке інше). Цей компонент наче підписник для компонента ViewModel. Тобто якщо у ViewModel змінилося якесь значення в об'єкті, то він оповіщає підписника про таку зміну. У випадку, якщо користувач активує якийсь елемент графічного інтерфейсу – компонент Представлення зробить виклик відповідної команди представленої Моделлю Представлення. В роботі це представлено за допомогою директив фреймворку Vue.js. Ці директиви й забезпечують так звану реактивність (в роботі представлено за допомогою текстової інтерполяції з використанням синтаксису Mustache(подвійні фігурні дужки), директив v-model, v-bind та інших).

Компонент Модель Представлення (ViewModel) містить в собі дані з Моделі для зв'язування а також команди, які може викликати Представлення, щоб впливати на зміну Моделі. У розробленому застосунку це JavaScript сценарії та механізм long polling.

2.2 Життєвий цикл роботи застосунку

Життєвий цикл застосунку складається з 3 основних етапів. Після введення користувачем запиту на обрану сторінку браузер користувача спочатку звертається до файлу Service Worker, якщо такий для цього сайту існує. Якщо цього файлу не існує в пам'яті на пристрої користувача, то перший етап пропускається.

На першому етапі надсилається запит до Service Worker файлу. В сценарії перевіряється наявність Інтернету. Якщо Інтернет відсутній, інформація щодо сайту, а це JavaScript сценарії, бібліотеки, зображення – завантажуються з кеш-пам'яті користувача. Важливо, щоб ця інформація була наявна в пам'яті – процес заповнення кешу інформацією відбувається попередньо, паралельно з завантаженням сторінки сайту через активне з'єднання Інтернет.

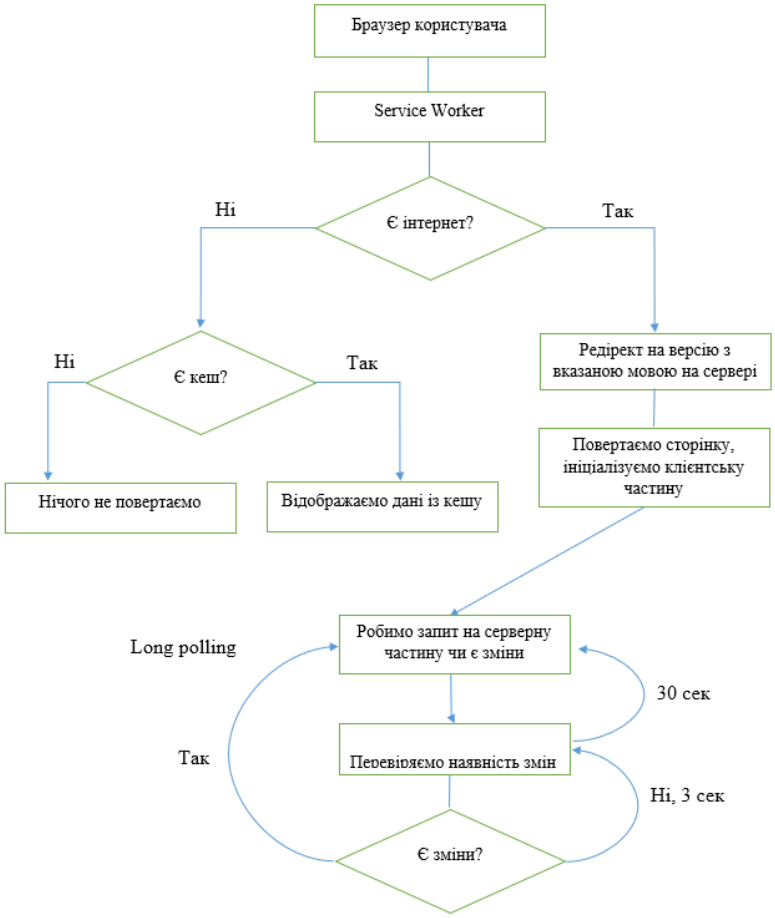

Рисунок 2.2.1 Життєвий цикл застосунку

На другому етапі розпочинається серверна робота. В основі серверної роботи лежить перенаправлення користувача на сторінку з обраною мовою, налаштування мета-тегів, запис відвідувань користувачів та інше.

Застосування підтримує багатомовність. Вона реалізована за допомогою Vue.js бібліотеки I18n. Бібліотека є зрозумілою та зручною у використанні. Для роботи з I18n потрібно встановити бібліотеку з використанням менеджера пакунків npm або yarn, завантажити повідомлення за ключем і мовою та налаштувати локаль застосунку. Щоб змінити мову застосунку – достатньо лише змінити локаль, бібліотека автоматично перемалює сайт. В застосунку поки підтримується лише 3 мови. Тобто якщо користувач зайшов на сайт вперше – на сервері перевіряються дані про з'єднання і відповідно до цих даних встановлюється мова для користувача. Проте нерідко буває, що користувач хоче обрати іншу мову. В верхній частині застосунку користувач може вибрати іншу мову і його вибір збережеться в кеш пам'яті. Під час наступної роботи з сайтом користувачу відобразиться сторінка з обраною ним мовою.

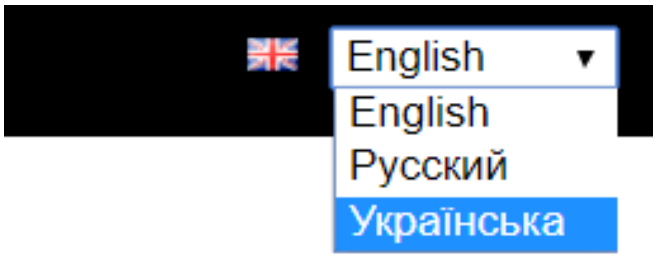

Рисунок 2.2.2 Вибір мови застосунку

Тексти, що представлені на клієнтській частині застосунку зберігаються в базі даних на сервері. Під час завантаження застосунку, завантажуються і тексти для багатомовності. Після завантаження ці тексти також зберігаються в сховищі браузера користувача, тому оффлайн версія сайту буде працювати, враховуючи обрану користувачем мову. Список доступних мов також зберігається на сервері,

що є зручно адже для того, щоб додати якусь мову не потрібно змінювати код застосунку.

Саме на третьому етапі роботи розміщена основна логіка. Застосунок повинен працювати в реальному часі, тобто якщо користувач завантажує файл, пише текстове повідомлення – все це одразу має відобразитися на девайсах інших користувачів цього з'єднання. Актуальну інформацію потрібно зберігати на сервері за ключем. На початку роботи системою генерується унікальний ключ, він прив'язується до цього сеансу. На серверній частині в базі даних за цим ключем зберігається інформація про завантажені файли, стан файлів, користувачів, збережені з'єднання, текстове повідомлення. Для того, щоб підтримувати інформацію на сервері в актуальному стані використовується технологія Long polling(з англ. довге опитування). Ця технологія походить від іншої, що має назву polling.

Реалізація технології polling досить проста – з клієнтської частини по черзі відправляється Ajax запит на сервер (так званий гачок), де перевіряється чи дані є в актуальному стані, якщо ні – запит повертає актуальні дані з бази даних.

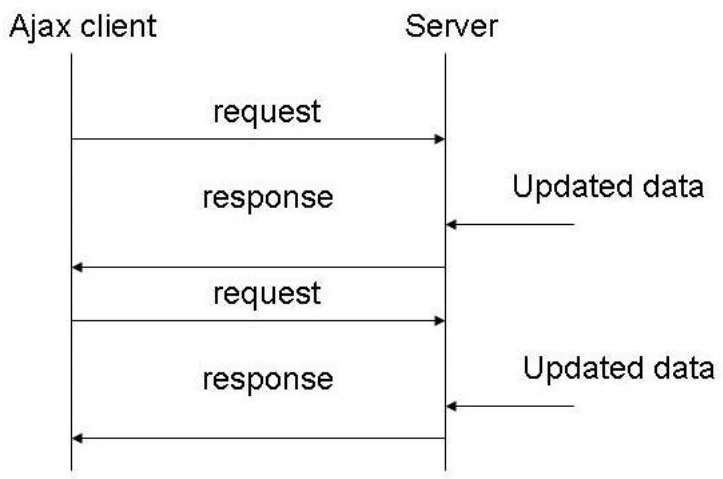

Рисунок 2.2.3 Long polling

Ці технології відрізняються на етапі перевірки інформації. Якщо ми використовуємо long polling і на серверній частині змін немає (тобто дані користувача актуальні), то ми не відправляємо відповідь на клієнтську частину, а продовжуємо серверний сценарій і через деякий час робимо перевірку на актуальність ще раз. Ця процедура перевірки в моєму застосунку триває протягом 30 секунд, після завершення повертаємо відповідь. Таким чином ми не робимо зайвих Ajax запитів на серверну частину і інформація залишається актуальною. Будь-яка зміна за певним ключем – змінює інформацію на девайсах всіх користувачів цього з'єднання.

2.3 Можливості застосунку

У цьому підрозділі розповідається про можливості застосунку. Це генерація і сканування QR-кодів, створення з'єднання, надсилання повідомлень, URL адрес, файлів, збереження з'єднання і його подальше використання.

Застосунок є Single-page application. Це означає, що немає потреби в перезавантаженні сторінки під час роботи. Зміна інтерфейсу здійснюється під час переходу між вкладками. Під час переходу введена користувачем інформація залишається. Всього є три вкладки.

На головній вкладці користувач бачить автоматично згенерований QR-код з посиланням на вкладку Передача в форматі - /transfer/generatedCode. Якщо на іншому пристрою перейти за цим посиланням, автоматично створиться з'єднання, користувача, що згенерував код перенаправить також. Також на сторінці можна відкрити поле для вводу і згенерувати свій власний код за введеним текстом. Внизу сторінки є відповіді на запитання щодо застосунку, для чого він потрібен та як ним користуватися.

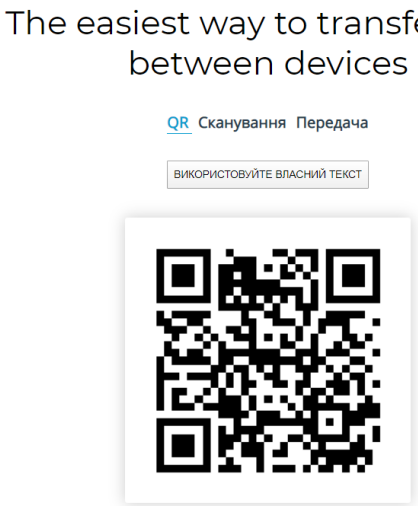

Рисунок 2.3.1 Головна вкладка застосунку

На вкладці сканування користувач може просканувати QR-код. Для цього потрібно надати доступ до камери на пристрої. Одразу після сканування з'явиться текст з результатом сканування, нижче тексту окремо винесено URL-посилання на ресурси в тексті. Сканування з мобільного пристрою супроводжується вібрацією.

На вкладці Передача розміщено поле для вводу повідомлення. Будь-яка зміна тексту відображається одразу на всіх пристроях з'єднання. Також кнопка для завантаження файлів з можливістю завантажити багато файлів одночасно. Існує обмеження на розмір файлів в 150Мб. Під час завантаження на всіх пристроях цього з'єднання буде відображатися індикатор прогресу завантаження. Усі користувачі можуть завантажити файли одразу. Під час завантаження генерується хеш код, за яким файли зберігаються на сервері. Видалення файлів відбувається кожної години. Також файли можна додати до з'єднання просто перетягнувши їх на екран браузера на своєму пристрої.

Над полем для вводу є дві таблички. В другій таблиці показано всі інші пристрої в цьому з'єднанні. На рисунку це пристрій Sony Xperia на операційній системі Android (див. рисунок 2.3.2).

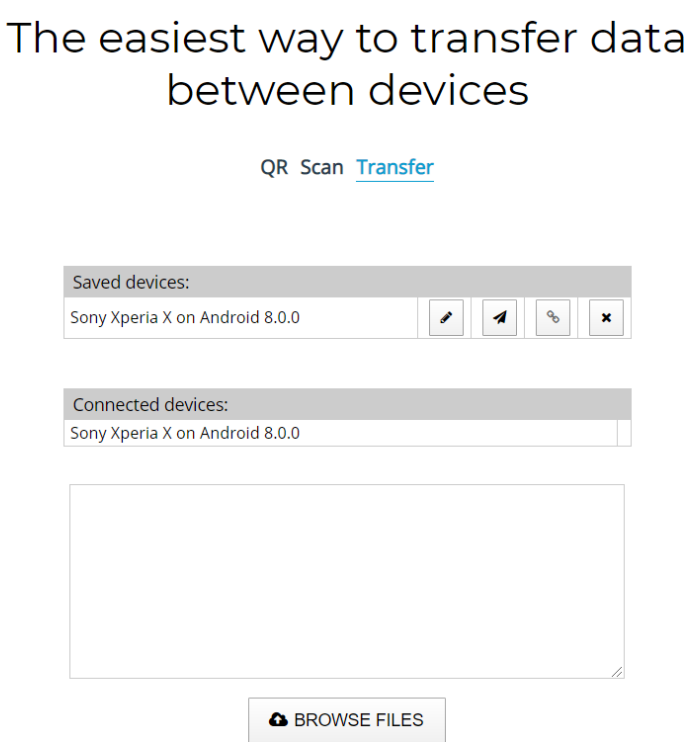

Рисунок 2.3.2 Вкладка Передача

Користувачі можуть зберігати з’єднання. Для цього конкретному користувачу надсилається запит на збереження з’єднання та дозвіл для сповіщень. Після отримання дозволу користувач може надсилати своїм збереженим контактам сповіщення. Всі активні збережені з’єднання зберігаються в окремій таблиці. Є можливість відредагувати свої з’єднання.

Застосунок можна додати собі на робочий екран смартфону. Це відкриває нові можливості, такі як пересилання файлів, тексту, зображень, посилань прямо з файлової системи, інших застосунків. Ця технологія називається Share Target API.

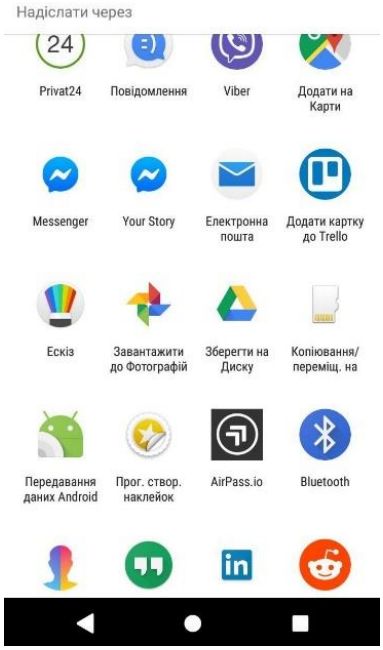

Рисунок 2.3.2 Меню Share Target

Для того, щоб скористатися цією можливістю необхідно на своєму пристрої обрати файл, натиснути на значок «поділитися» та вибрати застосунок із доступних для поширення. Файл, текст, зображення або посилання автоматично завантажиться на сервер, а користувача перенаправить на наш застосунок, де можна вибрати адресанта для надсилання контенту. Є можливість обрати і декілька адресантів, таким чином легко реалізовано мультипоширення.

Висновки

У курсовій роботі проведено аналіз та порівняно Web, Native та Progressive застосунки, зроблено висновки щодо їх переваг та недоліків. Зокрема обґрунтовано перспективи Progressive застосунків, оскільки вони поєднують в собі можливості native та web. Основним результатом курсової роботи є створений застосунок, за допомогою якого можна встановлювати швидкі Інтернет зв’язки між користувачами і групами користувачів, обмінюватися повідомленнями, файлами, посиланнями. В застосунку є можливість зберігати та редагувати власні контакти. Збереженим контактам можна надсилати web push сповіщення. Застосунок можна додати на головний екран смартфону і використовувати як стандартне рішення для поширення контенту з файлової системи. Для початкових вкладок працює оффлайн версія, що легко інтегрується до повної, як тільки з’явиться мережа в користувача. Застосунок підтримує багатомовність. Функціонал застосунку дозволяє широкому колу користувачів у різних сферах покращити процес обміну інформацією. У майбутньому планується впровадити необов’язкову реєстрацію користувачів, надання преміум аккаунтів з додатковими можливостями, збільшити кількість доступних мов, розширити можливості щодо поширення файлів безпосередньо з файлової системи пристрою. За подібними застосунками з використанням QR майбутнє.

Список використаної літератури

1. Документація Vue.js [Електронний ресурс]

https://vuejs.org/v2/guide/

2. Документація workbox [Електронний ресурс]

https://developers.google.com/web/tools/workbox

3. Документація Progressive Web Apps [Електронний ресурс]

https://developers.google.com/web/ilt/pwa

4. Стаття на Вікіпедії про кодування [Електронний ресурс]

https://en.wikipedia.org/wiki/Universal_Product_Code

5. Google developers console документація [Електронний ресурс]

https://console.developers.google.com/apis/

6. Composer документація [Електронний ресурс]

https://getcomposer.org/doc/

7. I18n документація [Електронний ресурс]

https://kazupon.github.io/vue-i18n/

8. Firebase документація [Електронний ресурс]

https://firebase.google.com/docs