Міністерство освіти і науки України

Національний університет України

“Київський політехнічний інститут імені Ігоря Сікорського”

Теплоенергетичний факультет

Кафедра автоматизації проектування енергетичних процесів і систем

**КУРСОВИЙ ПРОЕКТ**

з дисципліни

«Програмування у комп’ютерних мережах та прикладні мережеві протоколи-1»

(назви дисципліни)

на тему: ‘Міський путівник для київського метрополітену’

Студента 3 курсу групи ТВ-61

Калашникова-Травіна В. В.

Керівник доц. канд. екон. наук Гусєва І. І.

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_\_\_ Оцінка: ECTS\_\_\_\_\_\_

Члени комісії \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

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

(підпис) (прізвище та ініціали)

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

(підпис) (прізвище та ініціали

Київ - 2018Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Кафедра автоматизації проектування енергетичних процесів і систем | | | | | |
| (Повна назва кафедри) | | | | | |
| **Дисципліна:** Програмування у комп’ютерних мережах та прикладні мережеві протоколи-**1** | | | | | |
| **Спеціальність:** | | Інженерія програмного забезпечення | | | |
| **Курс** | 3 | **Група** | ТВ-61 | **Семестр** | 1 |

**ЗАВДАННЯ**

**на курсовий проект студента**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Калашников-Травін Владислав Володимирович | | | | |
| (прізвищ, ім'я, по батькові) | | | | |
| **1. Тема роботи:** | ‘Міський путівник для київського метрополітену’ | | | |
| **2. Термін здачі закінченої КП:** | | | 27.12.2018 | |
| **3. Вихідні дані до проекту:** | | середовище розробки для Java Intellij Idea, | | |
| система керування базами даних MySQL, мова програмування Java | | | | |
|  | | | | |
|  | | | | |
| **4. Зміст розрахунково-пояснювальної записки:** | | | | Постановка задачі, |
| Опис засобів та технологій реалізації програмної системи, | | | | |
| Опис програмної реалізації системи, | | | | |
| Методика роботи користувача з програмною системою | | | | |
|  | | | | |
| **5. Перелік графічного матеріалу**  Рисунок 3.1 - Концептуальна модель бази даних, Рисунок 4.1 - Екран | | | | |
| очікування, Рисунок 4.2 - Перелік станцій | | | | |
|  | | | | |
| **6. Дата видачі завдання:** 30 жовтня 2018 | | | | |

1

**Календарний план**

|  |  |  |  |
| --- | --- | --- | --- |
| **№**  **з/п** | **Назва етапів курсового проекту** | **Строк виконання**  **(тижні семестру)** | **Примітки** |
|  |  |  |  |
| 1. | Узгодження та затвердження теми | 2 |  |
|  |  |  |  |
| 2. | Технічне завдання та його аналіз | 3 |  |
|  |  |  |  |
| 3. | Проектування системи | 4 |  |
|  |  |  |  |
| 4. | Вибір засобів розробки | 5 |  |
|  |  |  |  |
| 5. | Розробка системи | 6 |  |
|  |  |  |  |
| 6. | Оформлення пояснювальної записки | 6-7 |  |
|  |  |  |  |
| 8. | Надання проекту на перевірку | 7 |  |
|  |  |  |  |
| 9. | Захист проекту | 7-8 |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Студент |  |  |  |
|  | (підпис) |  | (прізвище, ім'я, по батькові) |
| Керівник |  |  |  |
|  | (підпис) |  | (прізвище, ім'я, по батькові) |
| «\_\_\_\_» | \_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_ р. |  |  |

**ЗМІСТ**

ВСТУП……………………………………………………………………………………3

1. Постановка задачі……………………………………………………………………..4

2. Опис засобів та технологій реалізації програмної системи………………………...5

2.1. Обґрунтування вибору мови програмування……………………………….5

2.2. Обґрунтування вибору системи керування базою даних………………….7

2.3. Технології, використані під час розробки програмного продукту………..9

3. Опис програмної реалізації системи…………………………………………………11

3.1. Архітектура системи…………………………………………………………11

3.2. Опис бази даних……………………………………………………………...13

3.3. Алгоритм роботи системи…………………………………………………...16

4. Методика роботи користувача з програмною системою…………………………...17

ВИСНОВКИ.......................................................................................................................18

СПИСОК ЛIТЕРАТУРИ...................................................................................................19

**ВСТУП**

У сучасному світі мобільні електронні пристрої поширюються дуже швидко. Потужності збільшуються в геометричній прогресії. Зявляються нові технології, які дозволяють розмістити ще більші потужності на меншій площі. Компанії зі сфери розробки програмного забезпечення пропонують безліч засобів для розвязання тих чи інших проблем. Проте часто виникає потреба розвязати нетривіальну задачу або просто створити новий продукт. У такому разі єдине, що залишається - це розробка унікального продукту. Для того, щоб він виявився успішним, він повинен відповідати найвищим стандартам своєї галузі. Тому кожний створений продукт повинен бути потрібний користувачам, добре продуманий, спроектований та реалізований. Якщо потрібно, щоб продукт став поширений серед користувачів, він повинен бути також професійно представлений.

Продукт, розроблений в результаті курсового проекту можна назвати унікальним. Технологія BLE-маячків хоч і добре відома і поширюється, але не використовується в тій області, яку представляє даний проект. Отже, темою проекту є розробка додатку з цiльовою платформою Android з використанням технології "Beacon", згiдно з темою курсового проекту.

Такий програмний продукт є зручним у користуванні і потрібним як киянам, так і гостям столиці. Додаток допоможе у пошуку станції метро, вчасно сповістить користувача про знаходження станції найшвидшого громадського транспорту зовсім поблизу. Туристи почуватимуться спокійно гуляючи по місту, якщо будуть знати, що десь поруч є метро.

3

**1. Постановка задачі**

Для розробки на курсовому проекті з дисципліни "Програмування у комп’ютерних мережах та прикладні мережеві протоколи-1" була запропонована тема ‘Міський путівник для Київського метро’. До системи були поставлені наступні вимоги:

* програмне забезпечення має бути розроблене для операційних систем Android або iOS;
* допустимий вільний вибір мови програмування, якщо можливості вибраної мови гарантують виконання усіх вимог курсового проекту і використання необхідних технологій;
* застосування як основної технології - Bluetooth маячків;
* вибір, створення і використання бази даних для зберігання даних про станції метро і дані про маячки .

Програмна система в кінцевій реалізації проекту повинна давати можливість користувачеві побачити усю необхідну інформацію про станцію метро, біля якої він знаходиться в даний момент. При цьому має бути забезпечена конфіденційність даних користувача і ніяк не повинне використовуватися в обробці системою інформація про місцеположення користувача.

Вхідними даними для даної системи є місцезнаходження користувача, а вихідними даними, які отримує користувач, є інформація про станцію метро, поруч з якою користувач знаходиться.

4

4

**2. Опис засобів та технологій реалізації програмної системи**

**2.1 Обґрунтування вибору мови програмування**

На початку 90-х, Java, яка спочатку називалася "Oak", була створена командою на чолі з Джеймсом Гослінгом для Sun Microsystems (вiдтепер компанія належить Oracle).

Java була спочатку розроблена для використання на цифрових мобільних пристроях, таких як мобільні телефони. Однак, коли Java 1,0 була випущена для загального користування в 1996, її основна ціль була зміщена для використання в Інтернеті, забезпечуючи інтерактивність з користувачами, надаючи розробникам спосіб отримання анімованих веб-сторінок.

Тим не менш, було багато оновлень з версії 1,0, як J2SE 1,3 в 2000, J2SE 5,0 в 2004, Java SE 8 в 2014, і Java SE 10 в 2018.

Програми, які написані мовою Java, запускаються за допомогою віртуальної машини Java (вiд англ. JVM - Java Virtual Machine), яка переводить Java байт-код у власні інструкції процесора і дозволяє непрямі виконання програм операційною системою. JVM забезпечує більшість компонентів, необхідних для запуску байт-коду, який зазвичай менше, ніж при виконувані програми, написаною іншою мовою програмування.

Для розробки програм на мові Java потрібен комплект розробки програмного забезпечення Java (SDK), який зазвичай включає компілятор, інтерпретатор, генератор документації та інші інструменти, що використовуються для отримання повного додатку.

Час розробки може бути прискорено за рахунок використання інтегрованих середовищ розробки (IDE)-таких як JBuilder, Netbeans, Eclipse або INTELLIJ IDEA. IDE сприяють розвитку GUI, що включає в себе кнопки, текстові поля, панелі, рами, смуги прокручування та інші об'єкти.

Протягом багатьох років, Java зарекомендувала себе успішною мовою для використання в Інтернеті.

Java була розроблена з декількома ключовими принципами:

* простота у використанні: основи Java прийшли з мови програмування C++. Хоча C++ є потужною мовою, вона є складною у своєму синтаксисі і неадекватною вiдносно деяких вимог Java. Java побудована на удосконаленій ідеї C++.

5

* надійність: Java необхідна для зменшення ймовірності фатальних помилок від програміста. Маючи це на увазі, було введено об'єктно-орієнтовне програмування.
* безпека: оскільки Java спочатку була націлена на мобільні пристрої, які будуть обмінювати дані через мережі, вона була побудова так, щоб включити високий рівень безпеки. Java, ймовірно, є найбезпечнішою мовою програмування на сьогоднішній день.
* незалежність від платформи: програми повинні працювати незалежно від машини, на яких вони виконуються. Java була написана як портативна і крос-платформенна мова.

Команда Sun Microsystems була єдиною в поєднанні з цими ключовими принципами, і завдяки цьому популярність Java тільки зростає.

Сьогодні мова Java є широко використовуваним фондом для розробки та доставки контенту в Інтернеті. За даними Oracle, існує більш ніж 9 000 000 Java-розробників по всьому світу і більше, ніж 3 000 000 000 мобільних телефонів запущених на Java.

У 2014 році один з найбільш значних змін було те, що вийшла у загальний доступ Java SE 8. Зміни включали в себе додаткові функціональні можливості програмування, паралельну обробку з використанням потоків і покращена інтеграція з JavaScript.[2]

5

6

6

**2.2. Обґрунтування вибору системи керування базою даних**

Реляційна база даних — це тип бази даних, що зберігає інформацію в електронних таблицях і здійснює пошук даних в одній таблиці на підставі визначених ключових полів іншої таблиці. Реляційна база даних є сукупністю елементів даних, організованих у вигляді набору формально описаних таблиць, з яких дані можуть бути доступними або повторно зібрані багатьма різними способами без необхідності реорганізації таблиць бази даних.[6]

Такий спосіб збереження даних дозволяє логічно скомпонувати їх, що при правильному проектуванні БД дозволяє досить зручно вибирати дані. Є також і недоліки, а саме при роботі з великими об’ємами даних через зв’язки між таблицями це уповільнює роботу.

Існує багато різних реляційних СУБД і кожна використовується для своїх потреб. При реалізації проекту була обрана база даних MySQL.

MySQL — це компактна система керування базою даних, що характеризується високою швидкістю, стійкістю і простотою використання. Дана СУБД вважається гарним рішенням для малих і середніх задач. Найповніші можливості сервера виявляються в UNIX-системах, де є підтримка багатопоточності, що підвищує продуктивність системи в цілому.[6]

Тому саме через наступні можливості була обрана ця СУБД:

* простота у встановленні та використанні
* підтримується необмежена кількість користувачів, що одночасно працюють із БД
* висока швидкість виконання команд
* наявність простої і ефективної системи безпеки

Цілісність баз даних являє собою властивість даних, що визначає повноту і правильність інформації, яка вміщується в БД. Підтримка цілісності даних включає такі складові:

− структурна цілісність;

7

− обмеження реальних значень даних;

7

− посилкова цілісність.

Структурна цілісності передбачає виконання таких умов:

− наявність тільки однорідних структур даних типу "реляційне відношення";

− відсутність дублікатів кортежів;

− обов'язкова наявність у кожному відношенні первинного ключа;

− обмеження доменів, яке передбачає визначення кожного атрибуту на своєму домені;

− можливість застосування невизначених значень NULL (позначає відсутність будь-якого значення атрибуту).

Невизначене NULL значення розглядається, як значення невідоме на даний момент часу. Це значення при появі додаткової інформації може бути замінено на деяке конкретне значення. Введення NULL викликало необхідність застосування замість двозначної логіки тризначної логіки. У цьому випадку передбачаються реляційні операції з невизначеними значеннями.

Обмеження реальних значень даних вимагають, щоби значення поля належали деякому діапазону значень, або задовольняли певному арифметичному співвідношенню між значеннями різних полів.

Посилкова цілісність означає, що зміни в таблицях повинні виконуватися синхронно, а зміст двох пов'язаних таблиць має відповідати таким правилам:

* кожному запису основної таблиці відповідає нуль або більше записів підлеглої таблиці;
* в підлеглій таблиці немає записів, які не мають батьківських записів в основній таблиці;
* кожний запис підлеглої таблиці має тільки один батьківський запис основної таблиці.

Умови цілісності даних визначають, які дані можуть бути записані в БД у результаті додавання або оновлення даних.

8

**2.2 Технології, використані під час**

**розробки програмного продукту**

BLE-маяк неодноразово передає один сигнал, що інші пристрої можуть бачити. Замість того, щоб випромінювати видиме світло, він передає радіосигнал, який складається з комбінації букв і цифр інтервалом який складає приблизно 1/10 секунди. Bluetooth-обладнаний пристрій, так само як і смартфон може "бачити" маячок як тільки той заходить в діапазон.

Як вони виглядають? Маяки дуже маленькі, прості пристрої. Якщо Ви відкриєте один, Ви не знайдете складного устрою чи апаратного забезпечення. Ви знайдете ЦП, радіо-модуль та акумулятори. Маяки часто використовують невеликі літієві акумулятори (менші і потужніші батарейки АА). Вони бувають різних форм і кольорів, можуть включати акселерометри, датчики температури, або унікальні додаткові компоненти, але всі вони мають одну спільну річ-вони передають сигнал.

Це не просто кидати старе послання в повітря. Маяк передає унікальний ідентифікаційний номер, який повідомляє пристрій, що маяк поруч.

Як можна взаємодіяти з маяками?

Наприклад, коли торговий центр встановлює маяки в їхній будівлі, всі маяки будуть мати певні ідентифікатори, зареєстровані в спеціальному додатку. Це означає, що смартфон може відразу визнати, що вхідний ID має важливе значення і що це з конкретного торгового центру. ID, однак, має мало сенсу самостійно.

Що відбувається далі? Це залежить від того, що власник запрограмував його робити. Один код може викликати програму для відправки купон. Інші можуть запропонувати навігаційні послуги. Можливості майже безмежні. Маяки не відслідковують вас. Вони не зацікавлені в цьому. Вони просто транслюють сигнал. Ось чому цей сигнал може викликати багато різних речей.

Інтернет-платформа, дозволяє керувати, настроювати і оновлювати всі ваші маяки. Звідти ви можете розробити власну програму або скористатися додатковою програмою, яка називається системою керування контентом. Ці програми дозволяють зв'язувати посилання, зображення, відеозаписи та тексти з окремими маяками. Багато з цих платформ зроблені так, щоб бути дуже зручними. Наприклад, програма може дозволити власникові музею додати нові можливості для їх програми-галереї (наприклад, вікторини або аудіо-гідів), просто ввівши питання або текст. Програма потім робить всю роботу автоматично і зберігає все в хмарі так, що ваш додаток може легко отримати доступ до нього.

9

9

Bluetooth забезпечує інфраструктуру для всієї екосистеми маячка. Це стандарт для відправки даних на коротких відстанях. Ось чому обладнання маяка може бути простим. Навколо нас існує вже веб-Bluetooth, що може з'єднати маяки і смартфони, і майже все інше.

BLE − це енергоефективна версія Bluetooth. Спочатку була представлена у 2010. Малі потреби енергії BLE є життєво важливими для маяків, так як це дозволяє їм працювати протягом багатьох років на крихітних акумуляторах. Він також споживає набагато менше енергії, ніж старий і незграбний Bluetooth. Насправді, BLE є основною силою в IoT.

Програмне забезпечення маяка є відносно простим, але порядок виклику дії може бути складнішим. Кожна система трохи відрізняється, але процес передачі можна описати приблизно так:

Маяк передає свої ідентифікаційні номери приблизно десять разів кожну секунду (іноді більше, іноді менше, в залежності від його налаштувань). Пристрій Bluetooth, (наприклад, Ваш телефон) ловить цей сигнал. Коли виділений застосунок розпізнає його, він зв'язує його з дією або елементом вмісту, що зберігається у хмарі, і відображає його користувачу. Ви можете "навчити" ваш додаток, як реагувати на маяк сигналу шляхом розробки за допомогою сторонніх інструментів.

10

10

**3. Опис програмної реалізації системи**

**3.1. Архітектура системи**

Система представляє собою Android додаток написаний на Java. В Android інструменти SDK компілюють всі коди програми з будь-якими даними та ресурсами в файли APK: Android-пакет, який є архівним файлом з .apk суфіксом. Один АПК файл містить весь Android додаток і файл для декларацій пристроїв, які потребує програма.

Після встановлення на пристрої, кожен додаток живе у своєму власному ізольованою програмному середовищі безпеки. Таким чином, Android система реалізує принципи найменших привілеїв. Тобто, кожен додаток, за замовчуванням, має доступ тільки до компонентів, які він вимагає, але не більше того. Це створює дуже безпечне середовище, в якій додаток не може отримати доступ до частини системи, до якої не дозволено торкатись.

Компоненти додатку є основними будівельними блоками Android програми. Кожен компонент є іншою точкою, через яку система може виконувати вашу програму. Не всі компоненти є фактичними точками входу для користувача, а деякі навіть залежать один від одного, але кожен з них існує в якості власного вікна і відіграє особливу роль - кожен з них є унікальним будівельним блоком, який допомагає визначити загальну поведінку вашої програми.

Є чотири різних типи компонентів програми. Кожен тип слугує задля визначеної мети і має яскраво виражений життєвий цикл, який визначає, як компонент створюється й завершується.

*Activity(діяльність*) являє собою один екран з призначеним для користувача інтерфейсом. Наприклад, електронна пошта може мати один вид Activity , який показує список нових повідомлень електронної пошти, іншу Activity, щоб скласти лист, та іншу Activity для читання електронної пошти. Хоча Activity працюють разом, формуючи цілісний користувацький інтерфейс в поштовому додатку, кожен з них працює незалежно від інших. Таким чином, інша програма може почати будь-який з цих видів Activity. Наприклад, додаток камери може почати свою діяльність в поштовому додатку, що складає нову пошту, для того, щоб поділитися зображенням.

11

Діяльність реалізована як підклас Activity.

*Служба (Service)* являє собою компонент, який працює у фоновому режимі, щоб виконати тривалі операції або роботи для віддалених процесів. Служби не забезпечують користувацький інтерфейс. Наприклад, служба може відтворювати музику у фоновому режимі, поки користувач знаходиться в іншому додатку, або він може отримати дані по мережі, не блокуючи взаємодію користувача з активністю. Ще один компонент, наприклад, Activity, може запустити службу і працювати в з нею тандемом.

Сервіс реалізований як підклас служби, і ви можете дізнатися більше про це в керівництві Послуги розробників.

*Контент-провайдер (Content-provider)* керує загальним набором даних додатку. Ви можете зберігати дані у файловій системі, базі даних MYSQL, в Інтернеті, або будь-якому місці, до якого додаток може отримати доступ. Користуючись контент-провайдером, інші додатки можуть запитувати або навіть змінювати дані (якщо контент-провайдер це дозволяє). Наприклад, Android система забезпечує контент-провайдер, який керує інформацією про контакти користувача. Таким чином, будь-яка програма з відповідними правами може запитувати частину контент-провайдера (наприклад, ContactsContract.Data) читати і писати інформацію про конкретну людину.

Контент-провайдери також корисні для читання і запису даних, яка є приватною для вашої програми, а не розділяє. Наприклад, додаток Примітка зразку Pad використовує постачальника контенту для збереження запису.

Контент-провайдер реалізований як підклас ContentProvider і повинен реалізувати стандартні API-інтерфейси, які дозволяють іншим програмам виконувати транзакції.

12

**3.2. Опис бази даних**

На рисунку 3.1 зображена концептуальна модель бази даних системи. Вона містить у собі 4 таблиці. Дві з них мають прямий зв’язок з предметною областю, відповідають основним сутностям системи, дві інші таблиці є словниками для коректного формату зберігання даних.

Отже перша таблиця ‑ "beacons", містить достатню інформацію про усі бікони, що пов’язані з системою путівника. Ідентифікатор маячка складається з буквених і цифрових символів, а також тире, тож необхідно використати тип char. Необхідна кількість символів для зберігання ідентифікаційного номеру - 36, він має вигляд 2F234454-CF6D-4A0F-ADF2-F4911BA9FFA6. Також маячок може мати назву. На одній станції може знаходитись більше одного маячка, адже розміри станції може опинитись значними. Тому в таблиці "beacons" є зовнішній ключ ‑ ідентифікаційний номер станції з наступної таблиці ‑ stations.

13

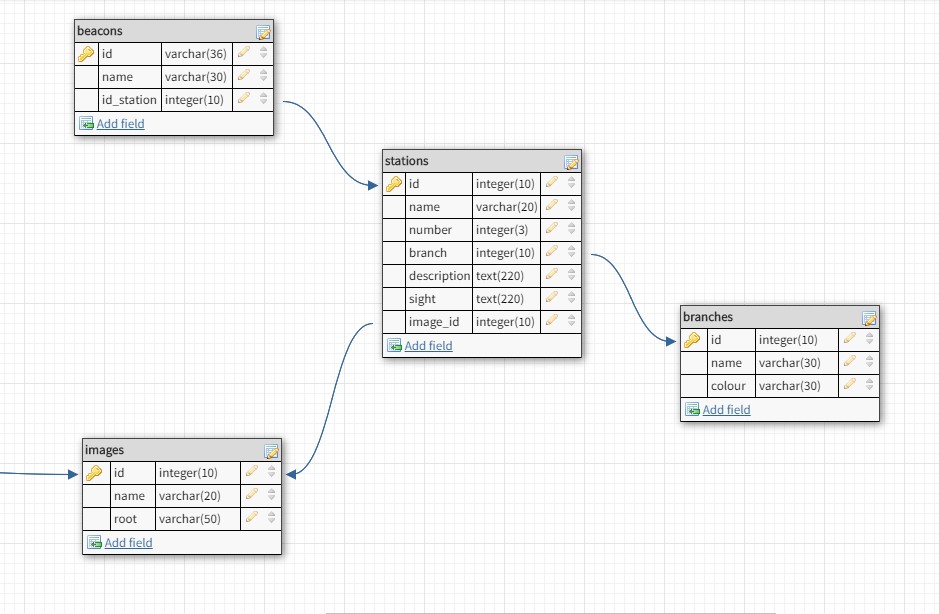


Рисунок 3.1 ‑ Концептуальна модель бази даних

13

Для кожної станції існує такий набір полів: name, number, branch, description, sight, image\_id. Детальніше про ці поля:

* name ‑ назва станції, має тип varchar, максимальна допустима кількість символів ‑ 20
* number ‑ номер станції, адже усі станції метро в києві пронумеровані (наприклад, Дорогожичі 311, Академмістечко 110 і т д). Тип поля ‑ integer, кількість символів 3, адже номер являється тризначним числом.
* branch ‑ гілка, до якої належить станція метро. В даному випадку варто винести перелік гілок з назвами в окрему таблицю, про неї пізніше. А це поле являє собою ідентифікаційний номер з таблиці branches і є до неї зовнішнім ключем
* description ‑ опис кожної станції, у який входять основні відомості, необхідні користувачу, і можливо, певна історична інформація тощо. Максимальна допустима кількість символів ‑ 220

14

* sight ‑ опціональне для заповнення поле, може містити основні відомості про пам’ятку що знаходиться поруч зі станцією метро. Максимальна допустима кількість символів ‑ 220
* image\_id ‑ зовнішній ключ до таблиці images, що містить дані про ілюстрації.

Таблиця images має наступні поля: id ‑ ідентифікаційний номер запису, первинний ключ таблиці, name ‑ назва ілюстрації для візуального представлення запису, root ‑ шлях до зображення.

**3.3. Алгоритм роботи системи**

15

Як вже було зазначено в частині 3.1 даного документа, система має такі основні компоненти ‑ Activity (активності) та Service (сервіс). Кожна активність працює з конкретним інтерфейсом, наприклад екран очікування біконів, екран перегляду знайдених станцій. Серед них має бути головна активність MainActivity. Усі активності разом формують єдине ціле ‑ користувацький інтерфейс. Тому активності напряму пов’язані з частиною архітектури проекта ‑ виглядом, і працюють з сторінками, обробляючи інформацію і відправляючи на екран.

Наступний компонент ‑ служба або сервіс ‑ працює у фоновому режимі. Служба чекає на сигнал від біконів. З певною періодичністю вона ловить сигнали і надсилає користувачу сповіщення. Так що не має потреби постійно працювати з додатком щоб отримати інформацію від системи, вона потурбується щоб користувачу були відправлені дані отримані з сервера як тільки бікон з’явиться в полі зору. Служба може запускатися і запускати інші компоненти системи навіть коли користувач відкрив інший додаток.

Частина роботи створеної в рамках курсового проекту системи знаходиться на сервері, де зберігаються дані. Додаток надсилає на сервер get-запит з ідентифікатором маячка, який отримала служба. Тоді сервер надсилає свій sql-запит до бази даних системи і відсилає отримані дані назад. Коректність даних постійно перевіряється і користувачу приходять сформована сторінка з обробленим текстом і ілюстраціями.

16

**4. Методика роботи користувача з програмною системою**

Щоб мати можливість користуватися усіма можливостями мобільного додатку, створеного в рамках курсового проекту, необхідно дотримуватися певних системних вимог щодо користування додатком. Ці вимоги пов’язані з використанням технології beacon-маячків. Після встановлення додатку необхідно в зайти в його настройки і ввімкнути дозвіл на використання місцезнаходження. Ще однією важливою умовою є увімкнення Bluetooth на смартфоні. Це усі незначні системні вимоги. Далі можна впевнено користуватись зручним і надійним ресурсом ‘City Walking Tour’.

Відкривши додаток, користувач побачить головний екран, що зображений на рисунку 4.1. На ньому видно, що додаток знаходиться в постійному пошуку і без відпочинку працює для свого користувача.

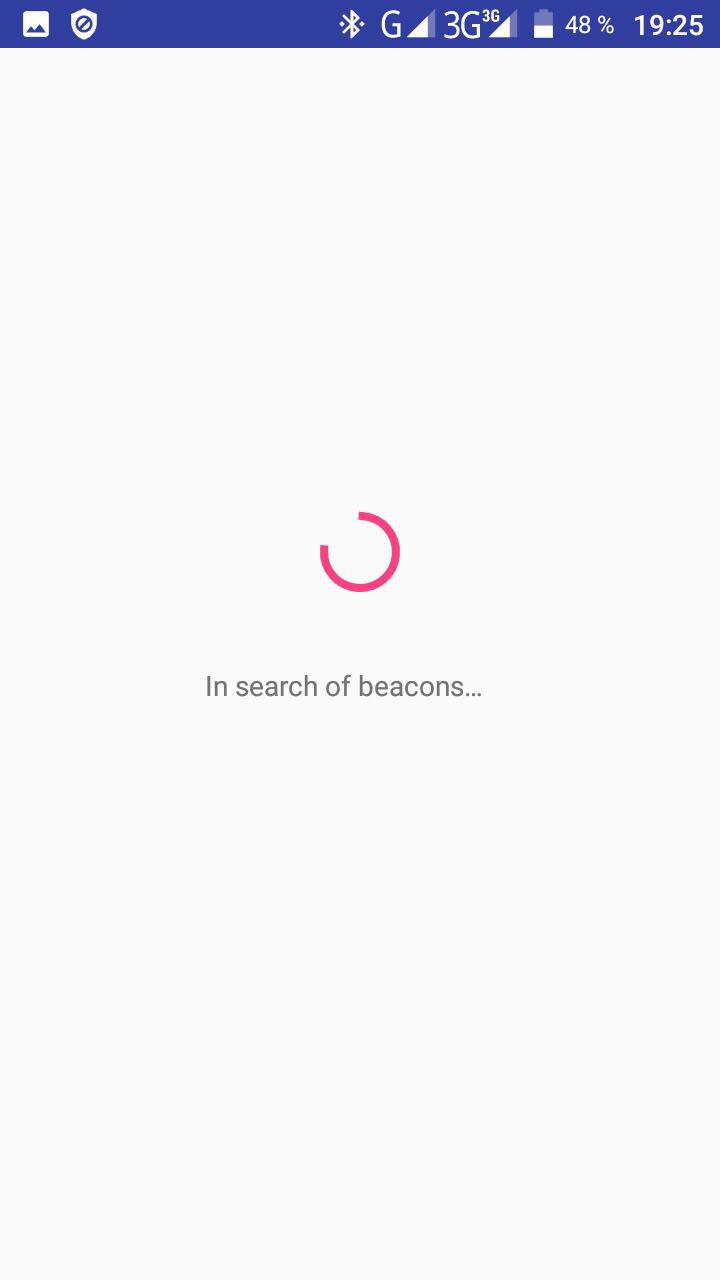


Рисунок 4.1 ‑ Екран очікування

17

Тепер додаток можна залишити і чекати сповіщення. Коли воно з’явиться і користувач перейде за ним у додаток, побачить наступний екран, представлений на рисунку 4.2.



Рисунок 4.2 ‑ Перелік станцій

На зображеному екрані користувач бачить перелік усієї знайденої і доступної в поточний момент часу інформації. В цілому тут лише ілюстрації та назви станцій метро. Далі користувач може перейти до потрібної йому станції для отримання детальної інформації про неї.

18

**ВИСНОВКИ**

Результатом розробки курсового проекту є мобiльний Android-додаток. Згiдно з темою, додаток являє собою фоновий сканер, що постiйно працює в фонi, навiть якщо телефон переведено в режим очiкування. Щойно сканер знайде поблизостi такий маячок, що зареєстрований у вiддаленiй базi даних, додаток завантажить iнформацiю про найближчу станцiю метро.

В процесi розробки було вдосконалено навички у мобiльному та асинхронному програмуваннi, проектуваннi баз даних. Крiм того, я ознайомився з технологiєю BLE-Bluetooth та принципом роботи маячкiв. Спроектований додаток потребував досконалої системи збору, тож було використано та вивчено систему збору "Gradle".

На жаль, Android SDK не має влаштованих iнструментiв для роботи з маячками, тож було використано сторонню бiблiотеку "AltBeacon". Вона може бути встановлена, використовуючи "Gradle", додавши посилання на репозиторiй до конфiгурацiйного файлу проекту.

Пiдбивши пiдсумки, можна сказати, що технологiя BLE-Bluetooth та побудований, згiдно з нею, маячок дуже корисна, особливо в сферах, де залучена велика кiлькiсть людей. Побудований додаток зможе завжди допомогти людинi знайти найближчу станцiю метро та найближчi до нього мiсцевi вiдзнаки.

**СПИСОК ЛІТЕРАТУРИ**

1. Android Beacon Library [Електронний ресурс] // 2015 ‑ Режим доступу до ресурсу: <https://altbeacon.github.io/android-beacon-library/>.
2. Java теорія [Електронний ресурс] ‑ Режим доступу до ресурсу: https://www.thoughtco.com/what-is-java-2034117.
3. Beacon теорія [Електронний ресурс] ‑ Режим доступу до ресурсу: https://kontakt.io/beacon-basics/what-is-a-beacon/.
4. Як працюють beacon'u? [Електронний ресурс] ‑ Режим доступу до ресурсу: https://www.intellectsoft.net/blog/what-are-beacons-and-how-do-they-work/.
5. Java.Android [Електронний ресурс] ‑ Режим доступу до ресурсу: https://www.dummies.com/programming/java/what-is-java-and-why-is-it-so-great/.
6. MySQL теоретичний зміст [Електронний ресурс] ‑ Режим доступу до ресурсу: https://searchoracle.techtarget.com/definition/MySQL.