Приватний заклад вищої освіти
Одеський технологічний університет «ШАГ»
Кафедра інформаційних технологій та фундаментальної підготовки

Випускна кваліфікаційна робота бакалавра

«СИСТЕМА УЗГОДЖЕННЯ ВЗАЄМОДІЇ КОРИСТУВАЧІВ З УРАХУВАННЯМ ГЕОПОЗИЦІОНУВАННЯ»

на здобуття бакалаврського рівня вищої освіти
зі спеціальності «122 Комп'ютерні науки»

Виконавці проекту

№	П.І.Б.	Група
1	Богдан Д. М.	КН-П-181
2	Боженович Д. С.	КН-Д-182
3	Демиров Д. А.	КН-П-181
4	Казанцев А. В.	КН-Д-182

Наукові керівники:

_____ к.ф.-м.н., доц. Самойленко Д. М.

_____ ст. викл. Судавная Т. М.

Автор звіту

_____ Демиров Денис Андрійович

Одеса – 2022

АНОТАЦІЯ

УДК 004.9
Д-30

Демиров Д. А. Система узгодження взаємодії користувачів з урахуванням геопозиціонування: Автореферат випускної кваліфікаційної роботи на здобуття бакалаврського рівня вищої освіти зі спеціальності «122 Комп’ютерні науки». - Одеса: ОТУШ, 2022 - 17 с.

Робота присвячена дослідженням, розвиненню та впровадженню інформаційної системи, головним призначенням якої є узгоджена взаємодія між користувачами з урахуванням їх географічної локації. Додаткові функції системи включають реєстрацію, авторизацію та автентифікацію, засоби обмеження доступу до контенту, ведення журналів активності користувачів системи.

Актуальність роботи визначається поширеністю задач, які ґрунтуються на оперативній фільтрації результатів за географічною ознакою, зокрема, за віддаленістю від поточного місцезнаходження об'єкта, а також необхідністю врахування у складі комплексної оцінки волатильності результату місцезнаходження суб'єкта пошуку. Особливу увагу звертає на себе той факт, що засоби встановлення локації можуть мати суттєві похибки або відмови спрацювання. Це дозволяє класифікувати задачі як такі, що формулюються в умовах невизначеності або неповної визначеності вихідних даних. Додатково, актуальність диктується потребою впровадження мобільних інформаційних систем та реалізації у них покращених засобів захисту інформації, зокрема, персональних даних

Об'єктом дослідження у рамках даної роботи виступає алгоритм складання комплексної оцінки результатів за багатьма критеріями в умовах неповної визначеності вихідних умов. Предметом дослідження обрано фактор географічної позиції (локації) користувача на момент формування запиту. У якості задач дослідження було встановлене наступне:

- провести огляд інформаційних систем, що враховують відомості про локацію користувачів, встановити характерні риси та напрями удосконалення.
- визначити програмні засоби (інтерфейси API), які дозволяють отримувати дані про локацію користувача, встановити їх особливості та ступінь вживаності для поставлених завдань.
- скласти програмне забезпечення, реалізувавши у ньому результати, досягнуті у попередніх задачах, провести випробування за різних умов та для різних пристроїв.
- впровадити у систему програмні засоби захисту даних для покращення показників інформаційної безпеки створеної системи.

Методи дослідження, що використані для досягнення поставлених задач, утворені суперпозицією методів системного аналізу, моделювання та когнітивної методології.

При складанні звіту використано 42 посилання на друковані та електронні джерела інформації.

ЗМІСТ

ВСТУП

Розвиток і поширення інформаційних технологій супроводжується перенесенням великої кількості послуг у цифровий простір. Стратегія розвитку інформаційного суспільства в Україні визначає, що «комунікаційні технології є необхідним інструментом соціально-економічного прогресу, одним з основних чинників інноваційного розвитку економіки» [1].

Сучасні мобільні пристрої, - смартфони, планшети, ноутбуки, - практично всі обладнані пристроями визначення геолокації, що значно покращує ефективність прикладного програмного забезпечення, яке здатне врахувати у своїй роботі дані щодо розміщення користувача. Подібні технології широко проникають у всі галузі діяльності, створюючи окремі напрями наукового дослідження - ГІС (геоінформаційні системи) [2].

У той же час, широкий вжиток інформаційних систем, особливо з реалізованими засобами локації, потребує покращення інформаційної безпеки як щодо захисту персональних даних користувача, так і щодо відомостей про його місце-знаходження [3]. Це актуалізує додаткові дослідження і розробки в зазначеній області.

В якості предмету даної роботи було обрано досить поширену систему узгодження взаємодії користувачів до функцій якої належать розміщення пропозицій, що надходять з боку одних користувачів, та пошук відповідних пропозицій з боку інших. При пошуку має враховуватись обмеження, що їх накладають географічні локації обох користувачів.

Також функції переважної більшості подібних системи мають передбачати можливість збереження історії розміщення, пошуку пропозицій, а також фактів їх прийняття чи відхилення. У перспективі - засобів оплати. Відповідно, це передбачає засоби авторизації та автентифікації, а також убезпечення персональних даних від несанкціонованого доступу інших користувачів.

Детальне технічне завдання до проекту наводиться далі. У рамках даного звіту розглядається клієнтська частина, що являє собою мобільний додаток, створений для користувачів пристроїв з операційною системою «Android». Звіт складається з трьох розділів у яких описується загальна характеристика системи, деталі створення клієнтської частини та результати її розгортання та тестування. Висновки відбивають результати, що досягнуті у рамках даної частини проекту, загальний висновок є сукупністю звітів усіх учасників проекту.

ТЕХНІЧНЕ ЗАВДАННЯ ДО ПРОЕКТУ

Загальний опис.
Програмне забезпечення (ПЗ) призначене для створення розподіленої системи узгодження взаємодії користувачів, що поділяються за різними ролями. Головна функція ПЗ полягає в узгодженні інтересів користувача, який надає послуги, та користувача, що споживає послугу. Додаткова функція ПЗ має дозволити залишати відгуки та оцінки щодо усіх видів користувачів

Призначення:
ПЗ орієнтується на різні види узгодження користувачів: за географічним розташуванням, за видом послуги, за вартістю, за рейтингом тощо. ПЗ створюється за прикладом замовлення транспортних послуг, у той же час має складатись за принципом універсальності, мати змогу змінити змінити цільове призначення.

Вимоги до функціоналу:
Реєстрація користувачів:
Користувач ПЗ має змогу зареєструватись як за допомогою авторизаційних даних (логіну/паролю), або за допомогою облікових записів популярних соціальних мереж (Google, Facebook, тощо).
При реєстрації обов'язково зазначається засіб оперативного контактування (ЗОК) - телефон. На тестовому етапі, за умови ускладнення замовлення послуг телефонії (та/або СМС) можливе використання електронної пошти. Засіб оперативного контактування при реєстрації підлягає верифікації через відповідь на тестовий виклик (лист) або введення коду, пересланого через повідомлення. Рекомендується використовувати ЗОК у якості логіну користувача.
Для можливості ввічливого спілкування між користувачами при реєстрації має зазначатись ім'я, за яким до користувача будуть звертатись. Відповідне повідомлення (про те, що зазначене ім'я буде у листуванні) має бути розміщене на сторінці реєстрації.

Особистий кабінет споживача послуги
Так само, являє собою основний робочий засіб. Має реалізовувати можливості

- Розміщення замовлень
- Пошук пропозицій (оперативні замовлення у реальному часі)
- Перегляд відгуків та рейтингу

Картографія
ПЗ має підтримувати засоби ГІС (геоінформаційних систем) для спрощення локалізації надавача і замовника послуги. ПЗ має запитати дозвіл на визначення поточного місця знаходження пристрою, з якого користувач

застосовує ПЗ. При розміщенні пропозиції (замовленні) користувач повинен мати можливість зазначити власне місце на інтерактивній карті автоматично або у ручному режимі. Аналогічну можливість повинен мати користувач - надавач послуги.
У режимі пошуку пропозицій або замовлень користувачі ПЗ повинні мати режим перегляду результатів пошуку на карті з можливістю одержати деталі кожного зі знайдених об'єктів через карту (вибір вказівником миші або сенсором)

Архітектура
ПЗ має бути спроектоване за клієнт-серверною архітектурою з розподіленими вузлами:
Backend - серверне програмне забезпечення, що забезпечує збереження та оброблення поточних даних, взаємодію з базою даних. Надає програмний інтерфейс взаємодії (API) з іншими вузлами
WebApp - додаток, доступний з веб-простору (сайт), що забезпечує візуальний інтерфейс користувачів усіх категорій
MobileApp - додаток, що встановлюється на мобільних пристроях (смартфон, планшет), додатково до WebApp забезпечує моніторинг замовлень у фоновому режимі, дозволяє накопичувати дані при відсутності мережного з'єднання (офлайн) з подальшою синхронізацією з Backend.
Усі модулі ПЗ повинні мати засоби

- тестування функціональності,
- розгортання на новому пристрої (інсталяції)

Технології
Зберігання вихідних кодів ПЗ повинно здійснюватись за допомогою системи контролю версій (вибір конкретної системи узгоджується групою проекту).
Структура файлів та каталогів повинна відбивати архітектуру ПЗ, на зразок:

- root

    Backend

    Web

    Mobile

Для кожного з модулів рекомендується (з можливістю часткової зміни) наступні технології створення:

Backend - PHP, MySQL, NodeJS
Web - JS, HTML, CSS,або JS фреймворки, зокрема ReactJS, AngularJS
Mobile - Java, або Kotlin, або Flutter

Вихідний код повинен оформлятися у відповідності до загальноприйнятих стандартів, на зразок стандартів GNU http://www.gnu.org/prep/standards/, у тому числі з додержанням вимог до коментування та документування коду.

РОЗДІЛ 1
ЗАГАЛЬНА ХАРАКТЕРИСТИКА СИСТЕМИ

У відповідності до технічного завдання (ТЗ) система була спроектована за архітектурою «Progressive Web Application» (PWA). У складі системи було умовно відокремлено наступні архітектурні частини:

- «Mockup»,
- «Backend»,
- «Frontend».

Частина «Mockup» являє собою набір моделей та описів поведінки користувача. Модель подає інтерфейси взаємодії користувачів різного типу з інформаційною системою, траєкторії рухів та переходів між різними інтерфейсами. При створенні інтерфейсів застосовано принципи розумної навігації, включно із підходами UI/UX, висновки колористики та ергономіки мобільних пристроїв. Частина розміщена за адресою https://www.figma.com/file/gDr7qqU2ZRd0wt3giXqOnf/Untitled?node-id=0%3A1.

Частина «Backend» реалізована як централізований вузол збереження та оброблення даних, що створюються та модифікуються протягом життєвого циклу інформаційної системи. Дана частина включає у себе базу даних проекту та програмний інтерфейс (API) доступу до основних її функцій, зокрема, керування контентом, авторизації, узгодження профілів користувачів різного типу та/або різних пристроїв.

Частина «Backend» виконана на базі програмного забезпечення NodeJS та СУБД «MySQL». Додатково використані програмні пакети «Express», «Mysql2»

Клієнтська частина системи «Frontend», призначена для надання користувачеві доступу до функцій системи через засоби візуального інтерфейсу. Також до функцій даної частини належить збереження локальних даних для можливості часткового використання системи без мережі передавання даних (offline). За наявності у пристрої користувача модулів визначення локації, засобами даної частини мають бути передбачені функції одержання та оброблення їх сигналів. Вихідні коди та ресурси даної частини розміщені на репозиторії https://github.com/KristarDen/Taxi_android_app.git. Подальший звіт присвячений детальному опису цієї частини.

З метою покращення безпеки, звертаючи увагу на той факт, що мобільні пристрої досить часто підключаються до незахищених мереж передавання даних, інформаційний обмін між частинами «Backend» та «Frontend» захищений попереднім шифруванням сучасним криптоалгоритмом AES-256. Це убезпечує роботу системи у відкритих мережах, а також спрощує задачі автентифікації та доступу до контенту з обмеженим доступом без наявності ключів розшифрування даних, навіть за умови дискредитації інших модулів системи.

РОЗДІЛ 2
РЕАЛІЗАЦІЯ КЛІЄНТСЬКОЇ ЧАСТИНИ

Клієнтська частина інформаційної системи реалізована з використанням наступних технологій:

- Операційна система: Android, гарантується підтримка систем з версіями API від 23 до 31 (поточна на час початку створення проекту).
- Засоби розроблення: мова програмування Java (OpenJDK 64-Bit Server VM by Oracle Corporation) та середовище Android Studio
- Додаткове ПЗ та засоби: Google-API для засобів картографії, прокладання маршрутів, пошуку за назвами об'єктів.

У складі клієнтської частини (мобільного додатку) передбачено та реалізовано наступні функції:

- Авторизація користувача, реєстрація нових користувачів
- Подання та пошук пропозицій з урахуванням локації
- Керування особистими та поточними даними (кабінетом)

Початок роботи користувача із додатком супроводжується його реєстрацією як учасника інформаційної системи. При реєстрації користувач обов'язково надає відомості про електронну пошту, через яку буде здійснюватись оповіщення користувача щодо актуальних питань, а також про зміну активності у його особистому просторі. Попередньо до реєстрації введені дані проходять попередню валідацію за допомогою засобів регулярних виразів і відправляються на серверну частину тільки після її проходження

```
Pattern phone = Pattern.compile("^(\\s*)?(\\+)?([- _():=+]?\\d[- _():=+]?){12,14}(\\s*)?$"),
pass = Pattern.compile("(?=.*[0-9])(?=.*[!@#$%^&*])(?=.*[a-z])(?=.*[A-Z])[0-9a-zA-Z!@#$%^&*]{6,}"),
name = Pattern.compile("[a-zA-Z]|[a-яA-я]|[a-яA-я]{2,}"),
email = Pattern.compile("^([a-z0-9_\\.-]+)@([a-z0-9_\\.-]+)\\.([a-z\\.]{2,6})$", Pattern.CASE_INSENSITIVE);
```

Авторизація користувача здійснюється за допомогою персонального секрету (логіну та паролю), введеного на етапі реєстрації.

При реєстрації чи авторизації здійснюється комунікація з частиною «Backend» у результаті чого авторизація підтверджується або спростовується. Дані, що передаються між зазначеними частинами, шифруються за допомогою алгоритму AES-256 (режим CBC) та перетворюються у транспортне кодування Base64. Це убезпечує дані при передаванні. Для використання у різних частинах системи засоби шифрування реалізовані у вигляді окремого класу.

```
public static String encrypt(String strToEncrypt) {
    try {
        byte[] iv = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
        IvParameterSpec ivspec = new IvParameterSpec(iv);
```

```
            SecretKeyFactory factory = SecretKeyFactory.getInstance("PBKDF2WithHmacSHA256");
            KeySpec spec = new PBEKeySpec(SECRET_KEY.toCharArray(), SALT.getBytes(), 65536,
256);
            SecretKey tmp = factory.generateSecret(spec);
            SecretKeySpec secretKey = new SecretKeySpec(tmp.getEncoded(), "AES");
            Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
            cipher.init(Cipher.ENCRYPT_MODE, secretKey, ivspec);
            return Base64.getEncoder()
                    .encodeToString(cipher.doFinal(strToEncrypt.getBytes(StandardCharsets.UTF_8)));
        } catch (Exception e) {
            System.out.println("Error while encrypting: " + e.toString());
        }
        return null;
    }
    @RequiresApi(api = Build.VERSION_CODES.O)
    public static String decrypt(String strToDecrypt) {
        try {
            byte[] iv = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
            IvParameterSpec ivspec = new IvParameterSpec(iv);
            SecretKeyFactory factory = SecretKeyFactory.getInstance("PBKDF2WithHmacSHA256");
            KeySpec spec = new PBEKeySpec(SECRET_KEY.toCharArray(), SALT.getBytes(), 65536,
256);
            SecretKey tmp = factory.generateSecret(spec);
            SecretKeySpec secretKey = new SecretKeySpec(tmp.getEncoded(), "AES");
            Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5PADDING");
            cipher.init(Cipher.DECRYPT_MODE, secretKey, ivspec);
            return new String(cipher.doFinal(Base64.getDecoder().decode(strToDecrypt)));
        } catch (Exception e) {
            System.out.println("Error while decrypting: " + e.toString());
        }
        return null;
    }
```

Обмін даними з серверною частиною реалізовано за допомогою окремих програмних засобів. У якості прикладу далі наведено приклад передавання даних методом POST, що широко вживається у різних частинах проєкту

```
public static String POST(String path, String jsonData) {

    OkHttpClient client = new OkHttpClient().newBuilder()
            .build();

    RequestBody formBody = new FormBody.Builder()
            .add("data", AES256.textToBase64(jsonData))
            .build();

    Request request = new Request.Builder()
            .url(path)
            .method("POST",
                    formBody )
            .build();
```

```
    try {

        Response response = client.newCall(request).execute();
        String decryptedData = AES256.base64ToText( response.body().string() );

        return decryptedData;

    } catch (IOException ex){
        return  ex.getMessage();
    }
}
```

Основну частину взаємодії користувача із додатком реалізовано за допомогою роботи з картою. Користувач має змогу визначити локацію, що становить його інтерес, за допомогою маркеру або через пошук за адресою чи назвою об'єктів.

Для забезпечення описаної функціональності у проєкті вжито наступні засоби: Geocoding API - одержання даних за координатами (https://developers.google.com/maps/documentation/geocoding); Direction API - для складання маршруту між заданими точками у вигляді полілінії (https://developers.google.com/maps/documentation/directions); Places API - пошук повної інформації про об'єкт на базі його назви чи тексту адреси (https://developers.google.com/maps/documentation/places/web-service).

Клас GMapApi реалізує усі перелічені вище функції. Для прикладу наведено опис класу та метод FindPlaceByLatLan, що здійснює пошук за відомими координатами об'єкта.

```
public class GMapApi {
    private static String API_KEY = "****************************************";
    private static String FIND_PLACE_URL =
"https://maps.googleapis.com/maps/api/geocode/json?latlng=%s,%s&key=%s&language=%s";
    private static String DIRECTION_GET_URL =
"https://maps.googleapis.com/maps/api/directions/json?origin=%s,%s&destination=%s,%s&key=%s
";
    private static String FIND_PLACE_BY_NAME =
"https://maps.googleapis.com/maps/api/place/findplacefromtext/json?input=%s&inputtype=textquer
y&fields=formatted_address%2Cname%2Crating%2Copening_hours%2Cgeometry&key=YOUR_A
PI_KEY";
    public static String FindPlaceByLatLan(LatLng latlng, String language){
        OkHttpClient client = new OkHttpClient().newBuilder().build();
        Request request = new Request.Builder().url(
                String.format( FIND_PLACE_URL,
                        latlng.latitude,
                        latlng.longitude,
                        API_KEY, language )
            )
            .method("GET", null)
            .build();
        try {
            Response response = client.newCall(request).execute();
```

```
            JSONObject json_resp = new JSONObject(response.body().string());
            JSONArray results_json =  json_resp.getJSONArray("results");
            JSONObject address_components_json =  results_json.getJSONObject(0);
            String address_Name = "" + address_components_json.getString("formatted_address");
            return address_Name;
        } catch (IOException | JSONException ex){
            return ex.getMessage();
        }
    }
```

Як вже зазначалось вище, з повним кодом проекту можна ознайомитись у репозиторії https://github.com/KristarDen/Taxi_android_app.git

РОЗДІЛ 3
РЕЗУЛЬТАТИ ВИПРОБУВАНЬ

Результатом компіляції програмного коду клієнтської частини являється інсталяційний (APK) файл додатку, що може бути встановлений на смартфон або планшет з операційною системою «Android». Головна серія випробувань, звіт з якої наведено далі, проведена на смартфоні «Xiaomi MI A1» з операційною системою «Android 9».

Початок роботи нового користувача з додатком починається початкового екрану з логотипом додатка та анімованого елемента завантаження, його зовнішній вигляд наведено на рис. 1. Зображення на фоні це векторне зображення яке адаптується під розмір екрану пристрою, після чого конвертується у растрове зображення на яке додається ефект “розмиття”. Використання векторного зображення надає можливість отримати найвищу якість зображення для пристрою з будь якою роздільною здатністю екрану. Взагалі у проекті повсякчас використовуються векторні зображення для елементів інтерфейсу, що робить інтерфейс додатку чітким для будь якого екрану.

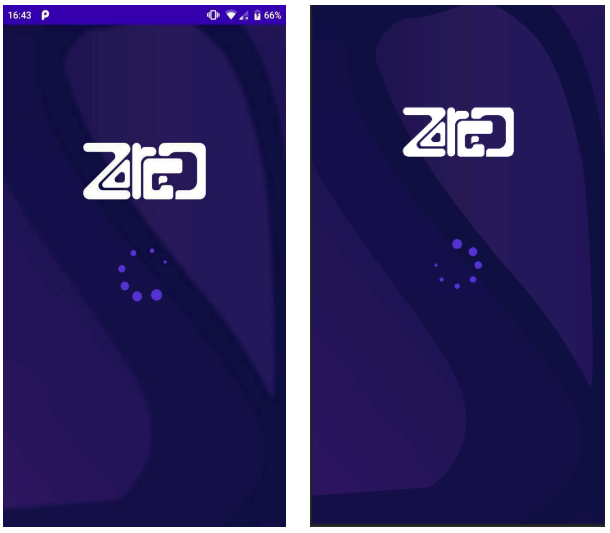

Рисунок 1 — Стартова сторінка (з ефектом розмиття, та без нього)

На цьому етапі виконується перевірка наявності у бази даних SQLite і наявність у базі запису з токеном авторизації який представлений у вигляді строки яку додаток отримую від серверу під час успішної реєстрації або авторизації і використовується для ідентифікації користувача сервером. Якщо такої бази ще нема, вона створюється додатком, якщо така база є то додаток намагається отримати токен авторизації користувача. Якщо токен авторизації є в базі то додаток переходить до основної активності — екрану мапи, а токен використовується для підтвердження і авторизації запитів до серверу. Але якщо його немає додаток переходить до екрану вибору методу авторизації : “Вхід” — за існуючим аккаунтом, “Реєстрація” — створити новий аккаунт, зовнішній вигляд екрану наведено на рис. 2.

Рисунок 2 — Екран вибору методу аутентифікації

За натиском кнопки “Зареєструватись” відкривається екран з реєстрації користувача у системі. Під час реєстрації здійснюється перевірка засобів комунікації - на електронну пошту надсилається код підтвердження, який має бути введений для завершення етапу реєстрації.

Додаток і усі його елементи інтерфейсу локалізовані на українську, англійську та російську. Додаток автоматично вибирає локалізацію за мовою системи Android пристрою. Система локалізації android додатків надає змогу швидко додавати нові мови для локалізації через те що усі текстові елементи представленні у виді таблиці, де кожен новий стовпець це мова локалізації, а рядок це певний перекладений текстовий елемент. Приклад того як виглядає локалізація на різних мовах на прикладі сторінки реєстрації рис 3.

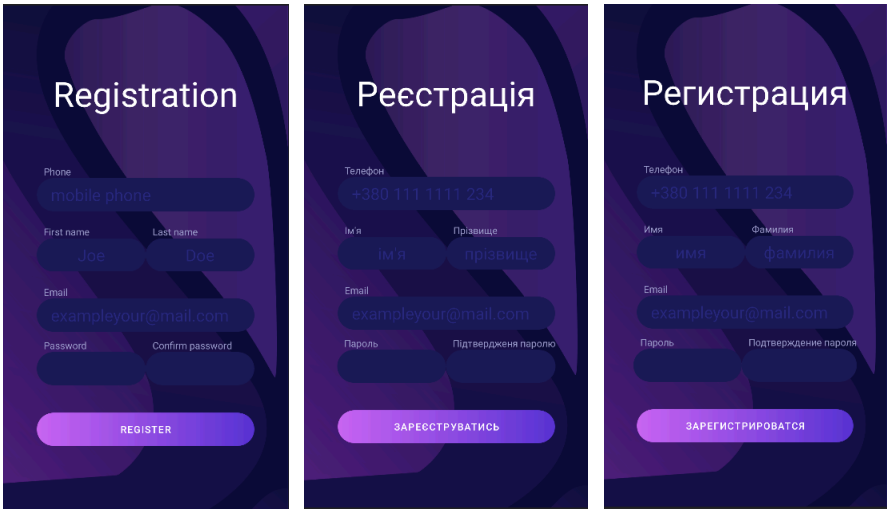

Рисунок 3 – Приклад локалізації на різних мовах

Якщо користувач раніше проходив реєстрацію, то він може за натиском кнопки “Вхід” увійти до системи за допомогою персональних даних, зазначених при реєстрації. Зовнішній вигляд екранів реєстрації та входу наведено на рис. 4.

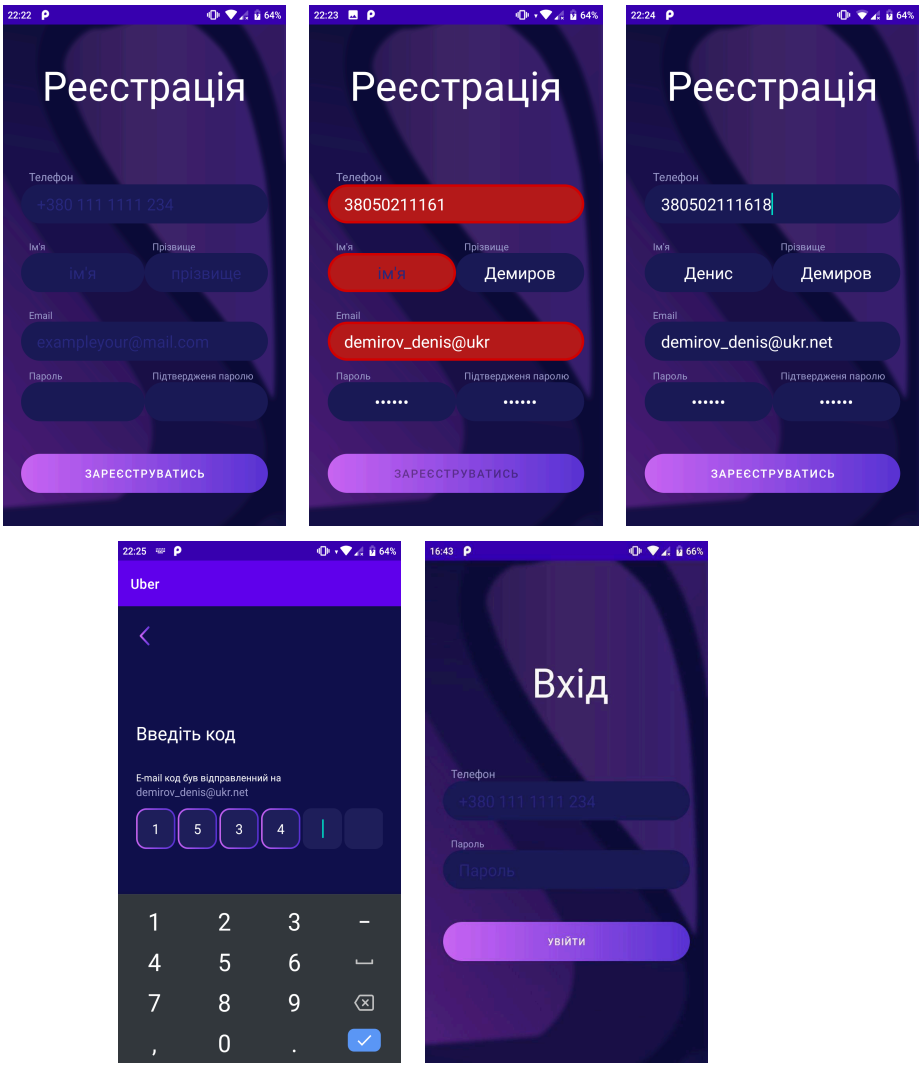

Рисунок 4 - Реєстрація користувача, підтвердження пошти та вхід

Форми вводу даних мають перевірку на валідність даних відповідному патерну. Форма номеру мобільного телефону перевіряє щоб текстовий рядок складався тільки з : 12 цифр і, не обов'язкового символу “+”, при цьому допускається наявність : пробілу, символу “–” між цифрами, та код країни у дужках, багато користувачів використовують такий стиль запису як у старих телефонних записниках, тому форма телефону підтримує як варіанти типу “380501111111”, так і “+(380) 50-111-1111”. Усі зайві символи видаляються з строки та записується у змінну. Форма паролю продубльована і всі перевірки в них теж, пароль має бути не менше шести символів і має обов’язково містити : одну малу та одну велику літеру латинського алфавіту, одну цифру та один з символів “!@#$%^&*” – така кількість та різноманіття символів збільшує кількість можливих комбінацій паролю що збільшує час та необхідні обчислювальні ресурси для підбору паролю методом підбору паролю перебором символів типу brute force (“груба сила”). Усі символи введені користувачем у форми паролю відображаються як крапки, але сам оригінал текст зберігається у змінну. Це

потрібно щоб пароль користувача було складніше піддивитися під час його введення у форму. На сторінці реєстрації ці форми продубльовані і вводити в них треба один однаковий пароль, це потрібно через те що користувач через заміну символів на крапки може допустити помилку і щоб він міг переконатись що пароль введено правильно, адже форми паролю перевіряють одна оду і якщо вони не однакові, то форма в яку вводився наразі текст виділяється червоним і користувач повідомляється спливаючим повідомленням на екрані. Форми ім’я та прізвища перевіряють щоб у вони складались символів кирилиці чи латиниці без цифр і знаків, а мінімальна довжина має бути не менше двох символів. Форма адреси електронної пошти має містити : ім'я користувача електронної скриньки, роздільний символ “@” та домен електронної скриньки (приклад “username@domain.net”). Якщо хоча б в одну з форм введено не правильні для нього данні і вони не проходять перевірку то кнопка реєстрації блокується і натиск на неї ні до чого не призводить, а усі форми у які введено неправильні данні виділяються червоним кольором, а користувача повідомляють спливаючими повідомленням на екрані. Перевірка форм перед відправкою має зменшити навантаження серверної частини та її мережевої інфраструктури. Після введення в усі форми валідні для них данні розблокується кнопка реєстрації за натиском на яку данні форм : компонуються, шифруються для безпеки даних на випадок перехоплення і відправляються на серверну частину яка для підтвердження того що скринька електронної пошти справді належить користувачу, відправляє на цю адресу цифровий код який треба відправити серверу назад щоб підтвердити що адреса належить користувачу і завершує реєстрацію. Після відправки серверу даних реєстрації додаток переходить на екран з формою вводу коду підтвердження, код підтвердження складається з шести цифр. Форма для введення коду підтвердження складається з шести форм для введення однієї цифри, інші символи заблоковані для вводу і ігноруються, а клавіатура яка відображається на екрані є цифровою без літер. Після введення у форму цифри фокус введення переводиться на наступну за чергою форму, а форма в яку ввели цифру виділяється градієнтним обрамленням. Після введення у останню форму цифри, додатком перевіряється чи введені в усі шість форм усі шість цифр, якщо так то усі цифри з форм компонуються, шифруються і відправляються на серверну частину, якщо коди збігаються то сервер відправляє у відповідь токен авторизації який записується додатком у змінну та буде використовуватися для підтвердження дій та аутентифікації користувача. Розділення форми коду на шість різних форм надає змогу користувачу бачити скільки цифр з потрібних шести вже введено і скільки залишилось.

Після входу до системи користувач отримує доступ до основних функцій системи. Однією з таких функцій є встановлення бажаної локації, яка може бути визначена декількома способами:

- з аналізу інформації від модуля встановлення локації пристрою,

- шляхом зазначення на мапі маркеру бажаної локації,

- через пошук локації за назвою або адресою.

На рис. 5 наведено відбитки екрану у режимі встановлення бажаної локації за допомогою пошуку з використанням адреси.

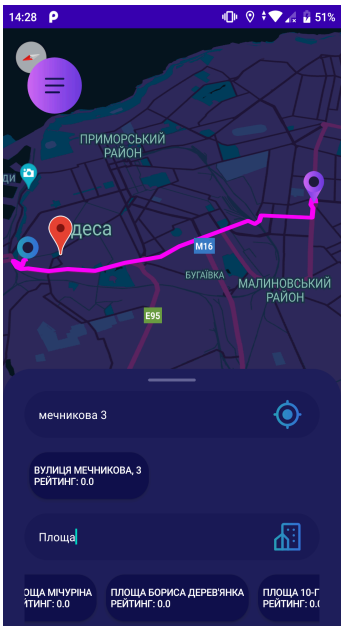

Рисунок 5 - Встановлення бажаної локації

Додаток отримує дані про місцезнаходження пристрою за запитом до операційної системи Android, операційна система самостійно за встановленими в нього алгоритмами вирішує за якими даними розрахувати місцезнаходження пристрою на мапі, це можуть бути як дані супутникові системи навігації : GPS, GLONASS, BeiDou, Galileo, так і дані мережі мобільного зв'язку типу 2G, 3G, 4G, 5G або даними о розташуванні за IP адресою пристрою у мережі internet, чи може використовувати всі дані разом для максимального уточнення місцеположення. Операційна система android раз у певний заданий додатком час (10 мілісекунд) викликає, встановлений додатком, метод оновлення координат куди передаються координати в форматі географічних широти і довготи. За координатами поточного місце розташування пристрою на мапі створюється маркер який відображає місцеположення пристрою, а при зміні координат пристрою маркер на мапі переміщується на нові координати. Для реалізації системи картографії і навігації (ГІС) використовується система картографії компанії Google яка складається з трьох частин :

1. Java бібліотека для системи android яка надає елемент інтерфейсу який відображає мапу з усіма графічними елементами мапи типу маркерів та полі-ліній.
2. Серверної частини компанії Google яка видає актуальні дані про мапу і елементи на на ній

3. Google maps API. Це програмний інтерфейс додатку реалізований як серверна частина яка отримує і відповідає на запити через протокол https, для доступу до цієї системи треба зареєструватись у системі *https://developers.google.com* та отримати токен для доступу до API. Цей токен використовується в усіх запитах до сервісу Google maps API. Цей сервіс може надавати інформацію про : оптимальний шлях між двома заданими точками з можливістю вибору типу пересування (автомобілем, громадським транспортом, пішки), пошуку інформації про можливу адресу, назву, поштовий індекс за зазначеним координатами у вигляді географічної ширини та довготи, пошук місця на мапі за текстом (це може бути як адреса так і назва об'єкту).

   При використанні цих систем були виявлені такі проблеми :

   Усі запити до системи Google Maps API коштують грошей які автоматично списуються з рахунку за кожен запит. Якщо система не зможе списати гроші з рахунку наприклад через їх нестачу то відповідей від системи з даними не буде. Чим більше користувачів та запитів тим більше потребується кошт на рахунку.

   Графічний елемент мапи постачається закритою бібліотекою. Через це його використання накладає деякі обмеження на створення графічного інтерфейсу в який його буде вбудовано, наприклад елемент мапи перехоплює події натиску на екран. Через це натиск може пройти скрізь інші елементи інтерфейсу які знаходяться поверх мапи і буде зафіксований як натиск на мапу, крім елементів EditText та Buttot, тому саме вони здебільшого використовуються в інтерфейсі.

Графічний елемент мапи підтримує систему стилів що надає змогу змінити колір елементів мапи. Кольорову схему мапи змінено згідно вимог дизайну представлених в mockup. Приклад змін наведено на рис. 6

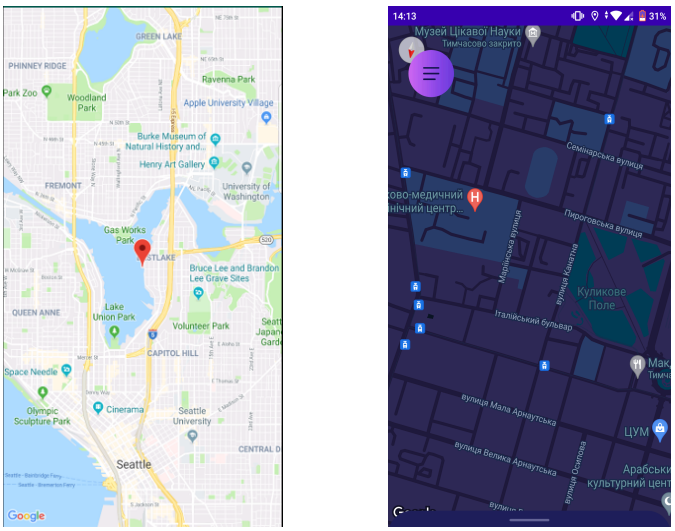

Рисунок 6 — З ліва оригінальна мапа, З права стилізована

Головний екран дає можливість користувачу :

Бачити своє місцезнаходження на мапі за встановленим на ній маркером. Додаток отримує дані про місцеположення пристрою та переміщує маркер місцеположення за цими координатами

За натиском на певне місце на мапі з’являється маркер місця призначення та отримує і відображає на мапі оптимальний шлях до нього за даними Google maps API. Графічний елемент мапи передає дані про координати місця на мапі на яке натиснув користувач до методу пошуку шляху. Метод робить запит до Google maps API, отримує дані про назви, адреси точок початку та кінця маршруту, довжину маршруту у метрах, оптимальний шлях від точки початку до точки кінця у вигляді масиву координат.

Можливість пошуку місця за текстом так і за назвою, наприклад за пошуком “Комп'ютерна Академія Шаг” чи за точною текстовою адресою цього місця “Садова 3” видасть одні й ті самі координати. Це дає можливість користувачу знайти потрібне місце на мапі не знаючи його точної адреси. Якщо за текстовою назвою знайдено декілька місць які містять такий текст, наприклад за текстом “Площа” може міститись у декількох назвах наприклад : “Площа Молоді”, “Соборна Площа”, “Думська Площа”. Додаток отримує перші 20 запропонованих місць з їх координатами якщо вони знаходяться в радіусі 50000 метрів від поточного місцезнаходження пристрою. Такі обмеження потрібні для того щоб користувачу у запропонованих місцях не попадались непотрібні місця, наприклад якщо користувач знаходиться в місті Одеса і він шукає місце яке містить у назві “Площа”, то йому повинні траплятись лише площі які знаходяться в місті Одеса, а не з Києва чи Дніпра тощо. За цими даними динамічно створюються елементи з адресами під формою пошуку, вони також автоматично видаляються при новому введені у форму. Форма пошуку має метод який автоматично перевіряє введений у форму текст, у цьому методі міститься таймер який відраховує 3 секунди від останнього введення у форму, якщо за 3 секунди у форму не було введено нових даних то текст з неї передається до методу пошуку можливих адрес, який відправляє ці дані Google Maps API та створює елементи інтерфейсу з запропонованими адресами. При натиску на елемент з запропонованою адресою вона фіксується у змінну, якщо заповнено обидві змінні з точкою початку та точкою кінця то викликається метод створення шляху, так само як в попередньому пункті.

Побачити довжину дистанції між двома точками на мапі у елементі інтерфейсу з показником дистанції.

Побачити дані про себе у особистому кабінеті та додати фото до профілю. Фото додається за кліком на елемент інтерфейсу з фотографією користувача, викликається системний діалог вибору фото з пам’яті пристрою, передача вибраного фото у активність з редагуванням фотографії для того щоб вона вписувалась у формат 1 до 1 у інтерфейсі профілю. Активність редагування фото надає можливість його обрізати за розміром у фіксованому співвідношенні сторін 1 до 1, накладання графічних ефектів зміни яскравості, контрасту, насиченості, різкості зображення. Для редагування використовується

відкрита бібліотека UCrop. Це фото зберігається у базі даних через backend. На рис. 7 наведено приклад профілю та вибору і редагування фото

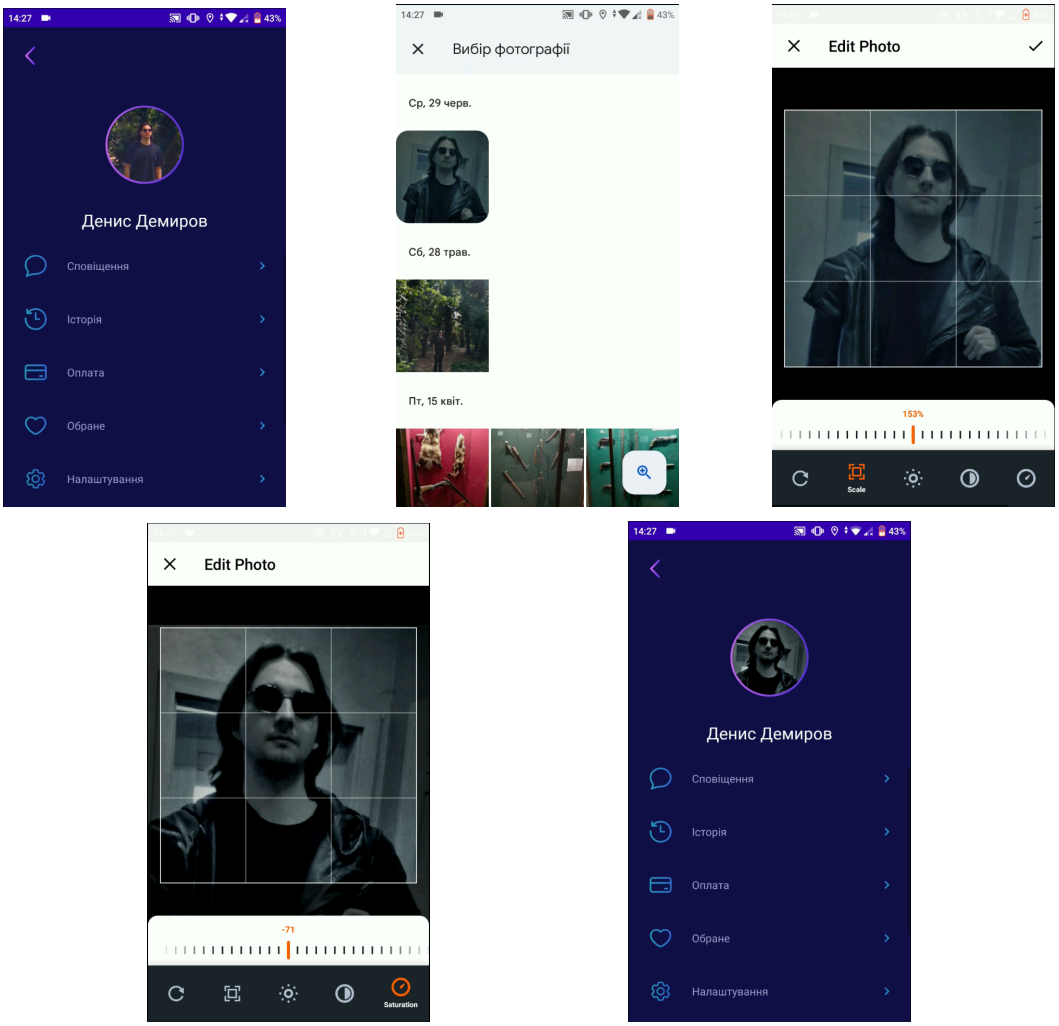

Рисунок 7 — Вибір і редагування фото (покроково)

Після вибору маршруту користувач може викликати таксі за натиском кнопки “Замовити”. За натиском елементи вибору адрес згортаються і на цьому місці з'являються елементи інтерфейсу вибору водія для поїздки. Картки водіїв створюються динамічно. У картці водія відображаються дані про його ім’я, рейтинг, модель автомобілю, фото, клас поїздки, ціна поїздки за вибраним користувачем маршрутом розрахована за ціною водія за метр помножена на дистанцію маршруту та кнопка замовлення поїздки саме в цього водія. На рис. 7 наведено приклад інтерфейсу під час вибору водія. Після натиску на кнопку замовлення інтерфейс згортається, користувачу видається повідомлення що таксі замовлено, по приїзду водія користувачу видається діалогове вікно підтвердження того що таксі прибуло і користувач сів в машину. Під час поїздки маркери водія і користувача переміщують на мапі за їх координатами.

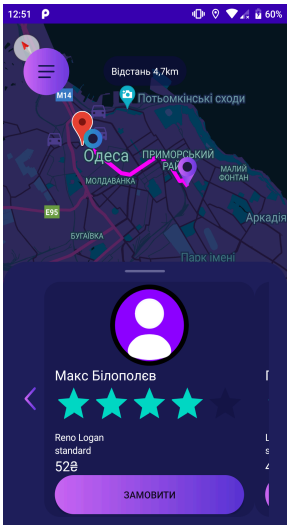

Рисунок 8 — Вибір водія

У підсумку, всі функції клієнтської частини, що були регламентовані технічним завданням було реалізовано та випробувано. Поточна версія додатку, що реалізує клієнтську частину, доступна у репозиторії https://github.com/KristarDen/Taxi_android_app.git

ВИСНОВКИ

Проведено огляд інформаційних систем, що враховують відомості про локацію користувачів, виявлено, що суттєве поширення мають системи, які пов'язані із наданням транспортних послуг. Саме для таких систем відомості про локацію мають основне значення. У якості прикладу вибрано для реалізації систему замовлення пасажирських перевезень (таксі).

Визначено програмні засоби (інтерфейси API), які дозволяють отримувати дані про локацію користувача, перевагу надано засобам від Google: Geocoding API, Direction API, Places API, Distance API. Дані засоби було вжито при створенні системи.

Використання систем картографії від третіх осіб накладає деякі обмеження на додаток, але надає актуальні дані. Зняття кошт за кожен запит потребує обмеження кількості можливих запитів від користувачів у певний проміжок часу

Спроектовано, реалізовано та випробувано систему узгодження взаємодії користувачів з урахуванням геопозиціонування її учасників. Система складена за клієнт-серверною архітектурою та складається з трьох частин, розміщених за адресами:

- «Mockup» https://www.figma.com/file/gDr7qqU2ZRd0wt3giXqOnf
- «Backend»https://bitbucket.org/bohdan_denys/taxi-website/src/master/
- «Frontend» https://github.com/KristarDen/Taxi_android_app.git.

У систему впроваджено засоби покращення інформаційної безпеки: додаткове шифрування даних, що передаються комунікаційним каналом, за алгоритмом AES-256. Також вжито засобів підтвердження персональних даних користувача, зокрема, номера телефону та електронної пошти. Це дозволяє реалізувати у майбутньому засоби двофакторної автентифікації, а також ускладнює зловмисникам вживання підроблених (невласних) даних для дискредитації системи.

В подальшому система може бути покращена додаванням нових функцій типу нових систем авторизації та оплати, введення бонусних систем та системи відгуків та створення окремого додатку для водія.

Систему було випробувано шляхом інсталяції на фізичні пристрої та проведення тестових сеансів комунікації. Результати випробування підтвердили загальну працездатність створеної системи.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Стратегія розвитку інформаційного суспільства в Україні. Розпорядження Кабінету Міністрів України від 15 травня 2013 р. № 386-р. // Урядовий портал. URL: https://www.kmu.gov.ua/npas/246420577
2. Лященко А. А. Особливості реалізації стандартів доступу до баз геопросторових даних в середовищі універсальних СКБД. // Геоінформаційні технології у територіальному управлінні: матеріали II міжнар. наук.-практ. конф. Одеса: ОРІДУ НАДУ, 2015. с. 17-21 URL: http://www.oridu.odessa.ua/7/7/Book_new_2.pdf
3. Стратегія кібербезпеки України. Указ Президента України від 14 травня 2021 року // Урядовий кур'єр від 28.08.2021 — № 165. URL: https://zakon.rada.gov.ua/laws/show/447/2021/print
4. Решетило В.П., Федотова Ю.В. Невизначеність та ризик: співвідношення понять та специфіка прийняття рішень // Проблеми системного підходу в економіці, випуск № 3(77)-2, 2020. с. 149. URL: http://psae-jrnl.nau.in.ua/journal/3_77_2_2020_ukr/23.pdf
5. Ладичук Д.О. Проектування бази геопросторових даних. Навчальний посібник / Ладичук Д.О., Шапоринська Н.М. - Херсон. - 2020. - 128 с.ISBN: 978-966-289-344-1
6. Олександр Смичніков. Параноя або сучасні реалії: як зберегти конфіденційність у мережі. - ФЬЮЧЕР МЕДІА. URL: https://mind.ua/openmind/20183329-paranoya-abo-suchasni-realiyi-yak-zberegti-konfidencijnist-u-merezhi (дата звернення 30.05.2022)
7. Булгакова О.С. Інформатика: візуальне програмування. Навчальний посібник (стереотипне видання) / Булгакова О.С., Зосімов В.В., Броницька Н.А., Танкова Н.В. - Херсон. - 2020. - 312 с. ISBN: 978-966-289-039-6
8. Чемакіна О.В. Дизайн системи візуальної інформації. Навчальний посібник / Чемакіна О.В., Рубцов А.Л., Свірко В.О. - Херсон. - 2019. - 200 с. ISBN: 978-966-289-216-1
9. Шеховцов А.В. Комп’ютерні технології для дизайнерів. Навчальний посібник (стереотипне видання) / Шеховцов А.В., Полєтаєва Г.Н., Крючковський Д.О., Бараненко Р.В. - Херсон. - 2019. - 318 с.ISBN: 978-966-2393-08-8
10. Чайковська М.П. Концептуально-методологічні засади управління маркетинговими ІТ-проектами в умовах цифрових трансформацій. Монографія / Чайковська М.П. - Херсон. - 2021. - 370 с.ISBN: 978-966-289-537-7
11. Гріффітс Д., Гріффітс Д., Head First Android Development Навчальний посібник 2021 - ISMB 978-1-492-07652-0
12. Google Inc. Платформа документації системи Google Maps, URL: https://developers.google.com/maps/documentation
13. Нудельман Г., Android Design Patterns: Interaction Design Solutions for Developers. Навчальний посібник 2013 — ISMB 978-1118394151