МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
Факультет електроніки та інформаційних технологій

Кафедра комп'ютерних наук

Кваліфікаційна робота бакалавра

РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ НА БАЗІ ОС ANDROID ДЛЯ УСПІШНОГО ФУНКЦІОНУВАННЯ ЕВАКУАТОРНОЇ СЛУЖБИ

Здобувач освіти
студент групи ІН-81 — Даниїл БУБЕНЩИКОВ

Науковий керівник,
доцент, кандидат фіз.-мат. наук — Олена ПРОЦЕНКО

Завідувач кафедри,
доктор технічних наук, професор — Анатолій ДОВБИШ

Суми 2022

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
Факультет електроніки та інформаційних технологій

Кафедра комп'ютерних наук

Затверджую_____

Зав. кафедрою Довбиш А.С.

"_____" _____2022 р.

ЗАВДАННЯ

до кваліфікаційної роботи

здобувача вищої освіти четвертого курсу, групи ІН-81 спеціальності «122 – Комп'ютерні науки» денної форми навчання Бубенщикова Даниїла Євгеновича.

тема «Розробка мобільного додатку на базі ОС Android для успішного функціонування евакуаторної служби»

Затверджена наказом по СумДУ

№_____ від _____2022 р.

Зміст пояснювальної записки: 1) огляд предметної області; 2) аналіз конкурентів; 3) постановка задачі; 4) вибір методів рішення задачі; 5) практична реалізація; 6) опис основних блоків; 7) висновки

Дата видачі завдання «_____»_____ 2022 р.

Керівник роботи _____ Олена ПРОЦЕНКО

Завдання прийняв до виконання _____ Даниїл БУБЕНЩИКОВ

РЕФЕРАТ

Записка: 127 стор., 37 рис., 4 додатка, 17 джерел.

Об'єкт дослідження — методи та принципи сучасної розробки мобільних додатків.

Мета роботи — розробка мобільного додатку на базі операційної системи Android для функціонування евакуаторної служби.

Методи дослідження — методи спостереження, порівняння та аналізу.

Результати — розроблено мобільний додаток на базі операційної системи Android, призначення якого забезпечити правильне функціонування евакуаторної служби. Застосунок створений за допомогою мови програмування Java у поєднанні з мовою розмітки XML та з використанням можливостей Google APIs та платформи Firebase.

МОБІЛЬНИЙ ДОДАТОК, ANDROID, UI-ДИЗАЙН, JAVA, XML, FIREBASE, GOOGLE MAPS, API, СУБД.

ЗМІСТ

ВСТУП

Сьогодні наявність зручного мобільного додатку, який би міг забезпечити повноцінне функціонування евакуаторної служби, набуває нового змісту, адже, на жаль, під час повномасштабної війни на території України, ще більша кількість людей потребує допомоги у вивезенні свого авто, його ремонті або ж, наприклад, евакуації транспортного засобу з територій, де ведуться активні бойові дії.

Застосунок, що буде створено у результаті виконання даної кваліфікаційної роботи бакалавра, є актуальним не тільки у даній ситуації, а й загалом, так як постійні транспортні пригоди, що виникають на дорогах України, а, також, регулярні проблеми із правильним функціонуванням особистих транспортних засобів людей нікуди не зникають.

Первинний огляд ринку та наявних конкурентів, проведений під час переддипломної практики, показав досить негативні результати, причиною яких стала повна відсутність аналогічних рішень, які б могли працювати на території усієї країни.

Це говорить про актуальність проблеми, що розглядається, та наявну потребу у розробці програмного рішення, яке б могло задовольнити функціонально як роботу водіїв евакуаторів, об’єднавши їх на одній платформі, так і власників авто, які з будь-яких причин потребують кваліфікованої допомоги і можуть легко, а головне швидко її отримати.

Саме тому головною метою даного проекту є розробка самостійного мобільного додатку на базі операційної системи Android, який би міг забезпечити потреби своїх користувачів, що виникли у результаті тієї чи іншої аварійної ситуації.

1. ІНФОРМАЦІЙНИЙ ОГЛЯД

У даному розділі викладено огляд проблемної області, тобто актуальність обраної теми на сьогодні, огляд подібних рішень на вітчизняному ринку та аналіз наявних конкурентів відповідно. Також, у даному пункті відображено постановку задачі, або ж технічне завдання, яке стало результатом проведених досліджень.

1.1. Огляд проблемної області

Формуючи огляд проблемної області, пов’язаний з обраною темою щодо розробки мобільного додатку для функціонування евакуаторної служби, важко не помітити актуальність даного рішення, тим паче зараз.

Безперечно, сьогодні наявність такого зручного застосунку набуває нового змісту, адже під час повномасштабної війни на території України, ще більша кількість людей потребує допомоги у вивезенні свого авто, його ремонті або ж, наприклад, евакуації транспортного засобу з територій, де ведуться активні бойові дії.

Програмне рішення, що розглядається, є актуальним не тільки у ситуації, що склалася, а й загалом, адже постійні транспортні пригоди, що виникають на дорогах України, а, також, регулярні проблеми із правильним функціонуванням особистих транспортних засобів людей нікуди не зникають. Усі вони потребують допомоги і даний мобільний додаток готовий вирішити їх проблеми та забезпечити їх потреби, що виникли у результаті тієї чи іншої аварійної ситуації.

1.2. Огляд подібних рішень

Огляд подібних рішень, в першу чергу, необхідний для первинного аналізу ринку та виявленні конкурентів у разі їх наявності.

Здійснення дослідження відбувалося засобами мережі Інтернет, а отримані результати продемонстровані за допомогою відповідних скріншотів та їх короткого опису.

Після виконання пошукового запиту, що має вигляд «Евакуатори України, мобільний додаток», було отримано відносно песимістичні результати, так як виклик евакуатора можна здійснити лише за допомогою того чи іншого сайту, подзвонивши по телефону, тобто це нагадує звичайне оголошення, а не повноцінний відповідний веб-ресурс. Приклад такого розміщення інформації можна побачити на рисунку 1.1.

Рисунок 1.1 – Сайт для виклику евакуатора у місті Суми

На рисунку, наведеному вище, відображено сайт місцевої евакуаторної служби. Аналізуючи його функціонал, можна помітити лише створене слайд-шоу та текстове поле, що містить номер телефону, який потрібно набрати вручну для отримання необхідної консультації чи допомоги.

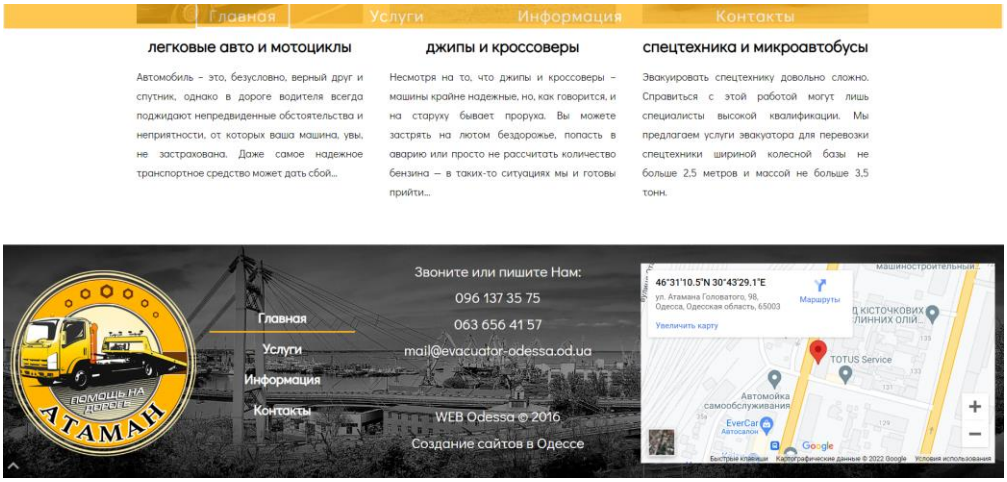

Рисунок 1.2 – Сайт компанії «Отаман» для виклику евакуатора у місті Одеса

Сайт, який відображено на рисунку 1.2, вже більше нагадує інтернет-ресурс типу лендінг. Він містить докладну інформацію про компанію, її переваги та види послуг, що надаються, та треба зазначити, що процедура замовлення евакуатора, також, полягає у виклику авто по набраному телефону. Мобільний додаток відсутній, а робота служби обмежена лише територією Одеської області.

Єдине рішення, реалізоване у середовищі Android та представлене у крамниці застосунків «Google Play» від однойменної компанії «Google» відображено на рисунку 1.3 відповідно.

Рисунок 1.3 – Мобільний додаток «Евакуатор Полтава» у «Google Play»

На рисунку 1.3, можна побачити мобільний застосунок, що має назву «Евакуатор Полтава». Перший недолік – це регіональна обмеженість, тобто робота даного застосунку покриває лише потреби міста Полтава, навіть не області. Детальна інформація відсутня, а процес роботи полягає у натисканні кнопки на панелі екрану вашого смартфона, яка автоматично набирає номер диспетчера евакуаторної служби. Так, це зручніше, ніж шукати інформацію на сайті, але не покриває і половини потреб середньостатистичного водія, що потрапив у аварійну ситуацію.

1.3. Аналіз конкурентів

Результатом проведеного огляду подібних рішень, процес якого описаний у пункті 1.2 даного розділу, є повна відсутність конкуренції на ринку мобільних додатків на базі ОС Android, що б обслуговували евакуаторну службу, можливості якої дозволяли б забезпечувати потреби водіїв з усієї території України.

Дана ситуація має як позитивні, так і негативні аспекти. Безперечно, відсутність, так би мовити, здорової конкуренції робить додаток, що розробляється, більш універсальним та бажаним серед його майбутніх користувачів.

З іншого боку, відсутність конкурентів не дозволяє у повній мірі оцінити потреби сучасного ринку та побажання його споживачів. Також, це негативно впливає на унікальність додатку, так як відсутність інформації про досвід аналогічних наявних застосунків не дозволяє створювати нетиповий функціонал, регулювати цінову політику тощо.

Підсумовуючи усе вище сказане, було прийнято рішення про проведення аналізу найпопулярніших на території України мобільних додатків у середовищі Android, які забезпечують роботу служб таксі, так як

специфіка розробки має багато спільних рис, а логіка роботи програми майже ідентична.

Згідно вітчизняного інформаційного ресурсу «The Page» топ-20 найпопулярніших мобільних додатків українського походження виглядають відповідно до рисунку 1.4, що наведено нижче.

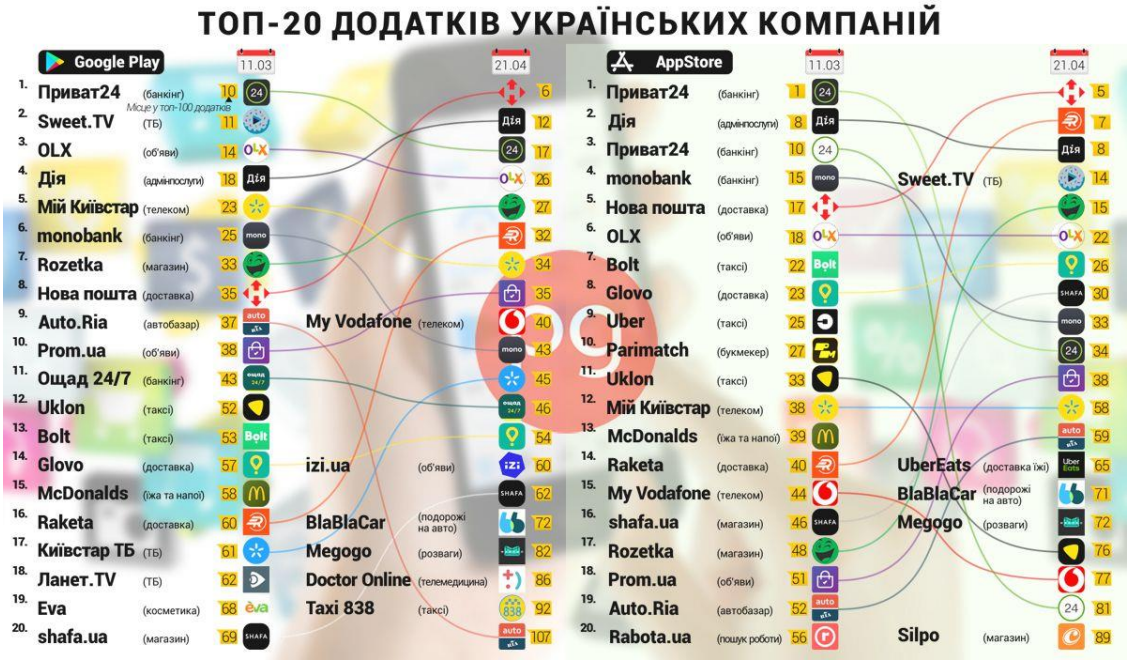

Рисунок 1.4 – Топ-20 додатків українських компаній (ресурс «The Page») [14]

У результаті аналізу інфографіки, можна побачити, що до топ-15 мобільних застосунків на базі операційної системи Android входять 2 додатки служб таксі. Вони представлені компаніями «Uklon» та «Bolt» відповідно.

Ці 2 додатки стануть базою для формування аналізу конкурентів відповідно до результатів проведених досліджень

Аналіз мобільного додатку «Uklon»

Додаток «Uklon» – це перший і найбільший український сервіс, що обслуговує роботу служби таксі. Сьогодні дана компанія працює у 27 найбільших містах України. Користувачі застосунку «Uklon» мають

можливість викликати авто в межах міста або ж дістатися до потрібного обласного центру, скориставшись звичайним додатком і витративши на це до 5 хвилин власного часу .

Отже, аналіз мобільного додатку сервісу «Uklon» включає у себе загальний функціонал застосунку, характерні кольори, можливості та переваги. Вигляд інтерфейсу застосунку відображено на рисунку 1.5 відповідно.

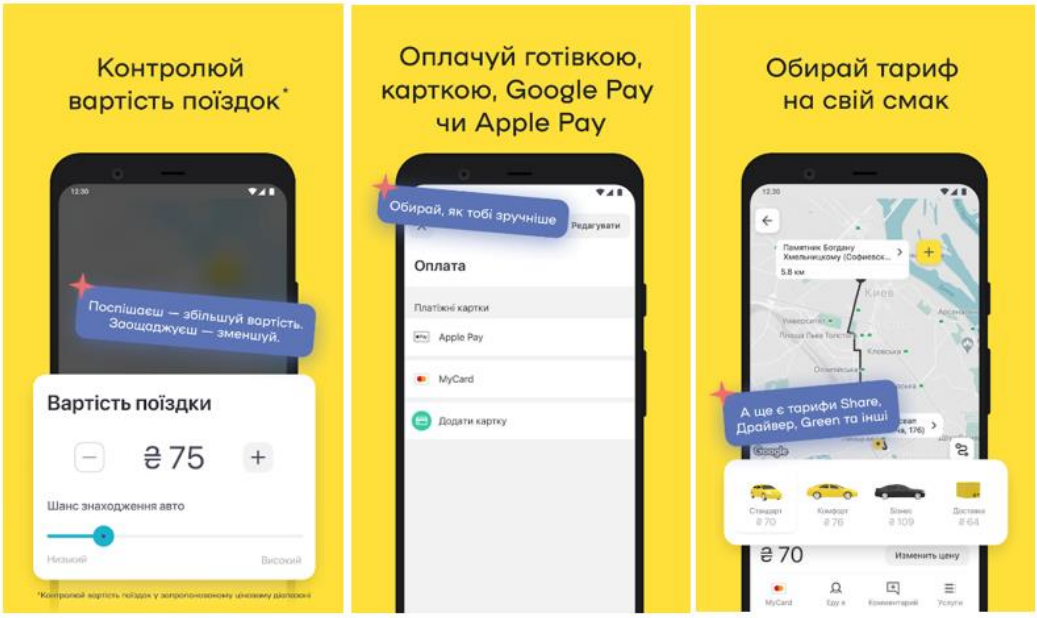

Рисунок 1.5 – Загальний вигляд додатку «Uklon»

У результаті аналізу, було зроблено певні висновки, які показують, що застосунок виконаний у характерних для компанії кольорах, тобто загальну кольорову гаму складають жовтий, чорний та білий кольори. Великою перевагою є наявність стилізованої карти, що реалізована на основі Google Maps, можливість вибору категорії авто, регульована вартість поїздки, оплата послуг як готівкою, так і за допомогою кредитної карти, а, також, можливість додавання нових пунктів маршруту під час його безпосереднього виконання.

Сервіс «Uklon» забезпечує 2 користувацькі ролі, а саме клієнта сервісу та водія, що обслуговує його. Для водіїв служби таксі, що розглядається, розроблено окремий додаток, який наведено на рисунку 1.6 відповідно.

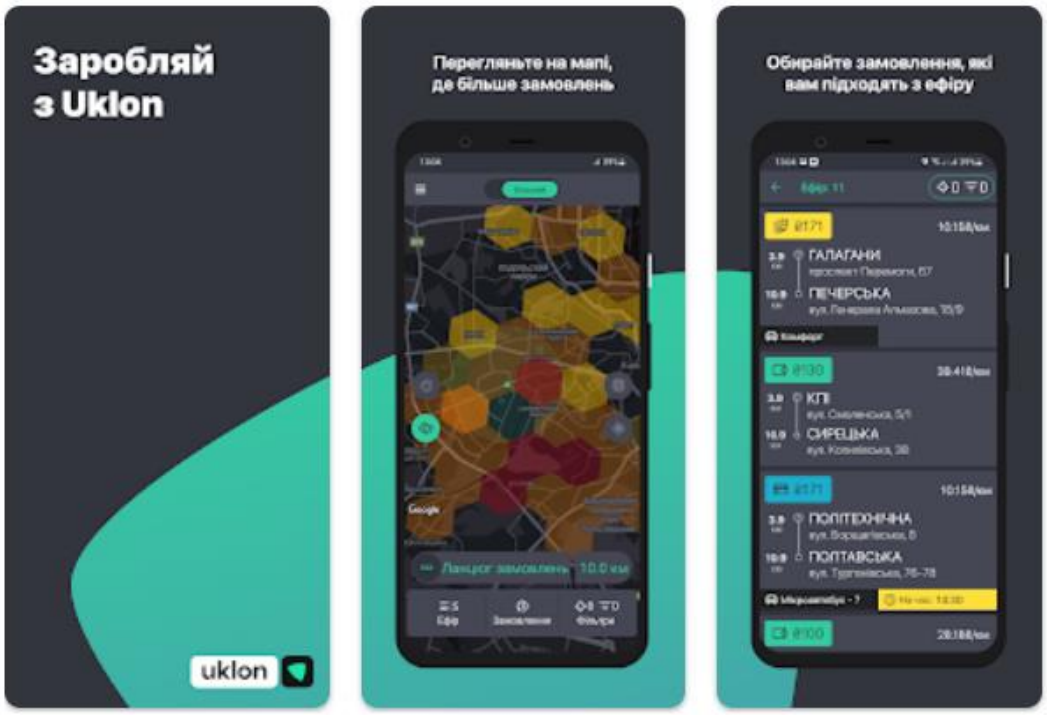

Рисунок 1.6 – Загальний вигляд додатку «Uklon Driver»

Отже, на рисунку 1.6 можна побачити мобільний додаток «Uklon Driver», задача якого допомогти водію служби отримати замовлення, супроводжувати його до користувача і виконати поставлене завдання.

Аналіз зовнішнього вигляду вказує на спільні риси у функціоналі з додатком, призначеним для клієнта служби, та повну відмінність у кольоровій гамі. Це допомагає відрізнити додатки між собою та легко зрозуміти, яку саме роль забезпечує той чи інший застосунок.

Функціонал додатку включає в себе стилізовану карту, на якій можна помітити не лише прокладений маршрут, але й певні зони, призначені для відтворення ситуації на дорогах (наявність чи відсутність заторів, аварійних ситуацій тощо).

Також, водій має можливість перегляду актуальних користувацьких запитів, що реалізовано у вигляду списку, історію замовлень, об’єм зароблених коштів та власний рейтинг, сформований на основі вже виконаних поїздок.

Отже, аналіз мобільних додатків, призначених як для водія, так і для клієнта, сервісу «Uklon» сформовано.

Аналіз мобільного додатку «Bolt»

Bolt – це компанія естонського походження, яка надає послуги служби таксі для своїх клієнтів, що використовують для цього однойменний додаток. Існують 2 застосунки: для пасажирів і водіїв. Щоб замовити поїздку, достатньо лише натиснути кнопку, в результаті чого прибуде найближчий водій для виконання поставленого замовлення.

Аналіз мобільного додатку сервісу «Bolt» включає в себе загальний функціонал застосунку, характерні кольори, можливості та переваги. Загальний вигляд відображено на рисунку 1.7 відповідно.

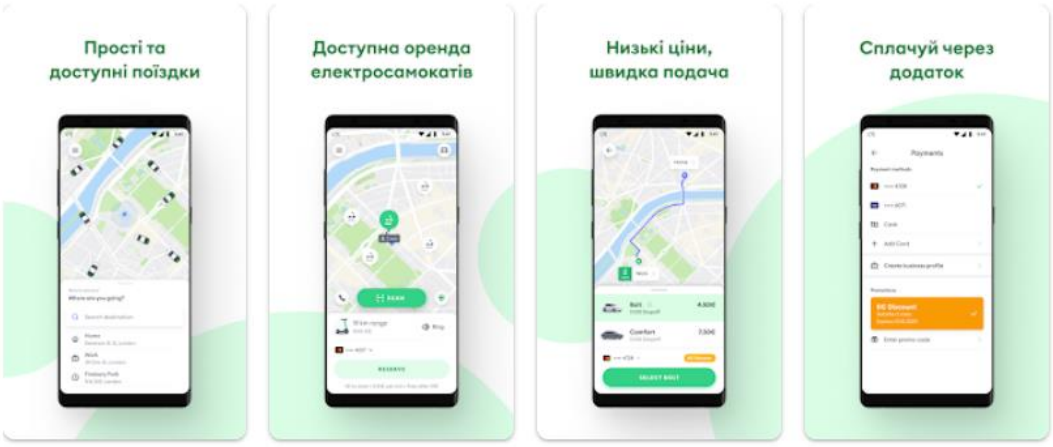

Рисунок 1.7 – Загальний вигляд додатку «Bolt»

Відразу можна помітити, що застосунок виконаний у характерних для компанії кольорах, тобто загальну кольорову гаму складають зелений та

білий кольори. Великою перевагою є наявність стилізованої карти, що реалізована на основі Google Maps.

Можливість вибору категорії авто, регульована вартість поїздки, оплата послуг як готівкою, так і за допомогою кредитної карти, можливість додавання нових пунктів маршруту під час його безпосереднього виконання, а, також, система оцінки водіїв та їх дій під час поїздки формують основний функціонал додатку.

Сервіс «Bolt» забезпечує 2 користувацькі ролі, а саме клієнта сервісу та водія, що обслуговує його. Для водіїв служби таксі розроблено окремий додаток.

Відповідно до рисунку 1.8, на якому продемонстровано загальний інтерфейс мобільного додатку «Bolt Driver», можна побачити, що застосунок надає можливість водіям, що користуються ним, виконувати свої замовлення зручно та легко, адже забезпечує їх чітким маршрутом, постійними підказками та надає можливість у разі виникнення проблем подзвонити у диспетчерську службу або ж зв’язатися, безпосередньо, з поточним клієнтом.

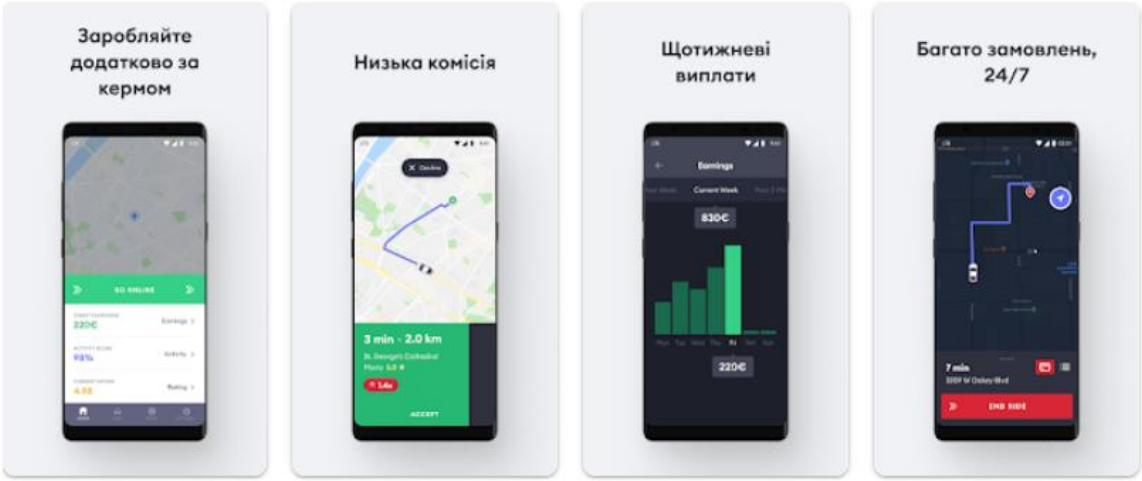

Рисунок 1.8 – Загальний вигляд додатку «Bolt Driver»

Також, на рисунку 1.8 наведено деякі статистичні дані, які допомагають водію відслідковувати кількість виконаних замовлень та об’єм зароблених за них коштів за окремий період часу відповідно.

Отже, аналіз додатку сервісу таксі «Bolt» також, успішно завершено.

Проміжні висновки після аналізу запропонованих мобільних додатків, виглядають наступним чином:

- застосунок повинен бути виконаний у традиційних для компанії та загального бачення продукту кольорах;
- додаток повинен містити 2 користувацькі ролі: клієнт та водій евакуатора;
- обов’язкова реєстраційна форма та подальша форма авторизації;
- зручний та легкий у використанні інтерфейс;
- наявність карти та стилізованих елементів на ній;
- можливість сформувати замовлення та відмінити його;
- підтримання зв’язку між клієнтом та водієм, що його обслуговує;
- статус замовлення, відслідковування проміжних результатів.
- сповіщення про успішне завершення поточного замовлення.

На основі проведеного аналізу та сформованих проміжних висновків було сформовано постановку задачі, що і стало основою для наступного пункту відповідно.

1.4. Постановка задачі

Постановка задачі або ж формування технічного завдання (ТЗ) – це вихідний документ для проектування необхідного технічного об'єкта, який встановлює основне призначення розроблюваного, у нашому випадку, мобільного додатку, його технічні характеристики, показники якості, а, також, певні спеціальні вимоги.

Даний етап у розробці програмного продукту є не менш важливим за наступні і вимагає відповідних високих аналітичних навичок.

Відповідно до матеріалу, наведеного вище, та отриманих знань було складено технічне завдання (ТЗ), в якому відображено усі вимоги для успішного функціонування створюваного додатку.

Основні вимоги до розробки

Основні вимоги до розробки додатку для обслуговування евакуаторної служби виглядають наступним чином:

- мобільний додаток на базі ОС Android із зв'язком з сервером БД (на вибір) засобами мережі Інтернет;
- функціонал реєстрації та авторизації користувачів за допомогою електронної пошти та сформованого під час реєстрації паролю;
- зберігання в БД персональних даних користувача: номер телефону, ім'я, e-mail, марка та номер авто, а, також, спеціальний ідентифікатор;
- наявна рольова модель: водій легкового автомобіля (замовник), водій евакуатора (виконавець);
- відображення геоінформаційної системи (ГІС) з об'єктами: місце розташування клієнта та водія евакуатора (геолокація мобільного пристрою, вказана точка на карті, введена адреса), розташування місця евакуації;
- можливість виходу з додатку та його повного видалення;
- перехід на інформаційні сторінки з матеріалами про додаток та сайт.

Наступним кроком буде формування вимог для певних блоків та процесів розроблюваного додатку відповідно.

Система замовлень (розміщення замовлення)

Основні вимоги до розробки системи замовлень додатку для обслуговування евакуаторної служби виглядають наступним чином:

- вказівка початкового і кінцевого пункту евакуації автомобіля;
- точна довжина маршруту, сформованого між водієм евакуатора та клієнтом з його авто;
- меню для зміни особистих даних, виходу з аккаунту чи його повне видалення, перехід на інші сторінки додатку;
- після формування замовлення відображення інформації про авто клієнта та марка евакуатора;
- можливість здійснення дзвінків між водіями.

Фінальний етап – це формування відповідних вимог до роботи із замовленням розроблюваного додатку.

Робота із замовленням

Основні вимоги до роботи із замовленням додатку для обслуговування евакуаторної служби виглядають наступним чином:

- відображення на карті евакуатора, що прийняв замовлення та інформацією про нього;
- реалізація перевантаження стану виконання замовлення та його відміни;
- повідомлення замовнику про прибуття евакуатора, повідомлення виконавцю в разі скасування замовлення клієнтом;
- повідомлення про успішне виконання замовлення.

Вимоги для найважливіших систем застосунку та відповідного загального функціоналу розроблюваного додатку сформовані. Технічне завдання (ТЗ), що було наведено вище, вже дозволяє розробити мінімально життєздатний продукт (MVP) та сформувати UI-дизайн додатку відповідно.

2. ВИБІР МЕТОДІВ РІШЕННЯ ЗАДАЧІ

У даному розділі викладено розробку UI-дизайну, яка включає у себе вибір основних кольорів, формування логотипу та відповідного дизайну основних сторінок мобільного додатку, а, також, огляд основних технологій, обраних для подальшої розробки застосунку.

2.1. Розробка UI-дизайну

UI-дизайн – це процес візуалізації прототипу, який розробили на підставі проаналізованого досвіду користувач, у результаті досліджень майбутньої цільової аудиторії та наявних на ринку конкурентів, що включає в себе роботу над графічною частиною інтерфейсу, а саме над різними ілюстраціями, кнопками, пунктами меню тощо [3].

У свою чергу, також, завдання розробки UI-дизайну – це визначення палітри кольорів і відповідного розташування об'єктів в інтерфейсі. Головне – це допомогти користувачеві легко і швидко зрозуміти, як користуватися тим чи іншим програмним продуктом.[3].

Основні кольори мобільного додатку

Процес розробки дизайну включає і процедуру вибору його кольорів. Вони можуть впливати на настрій користувача, його поведінку з додатком та формувати у нього певні асоціації, що пов’язані з ідеєю або функціоналом вашого застосунку.

Крім того, правильно обрана кольорова палітра – це одна із складових успішного UI-дизайну, яка буде передавати користувачу ідею, що закладена вами при розробці відповідної програми.

Для формування дизайну та подальшої розробки нашого додатку, було обрано 3 основні кольори, а саме: червоний, білий та чорний. Саме вони

допоможуть клієнту запам’ятати застосунок та відрізнити його від інших, аналогічних рішень.

Червоний колір – це найбільш емоційно насичений колір, що підвищує артеріальний тиск людини та прискорює серцебиття. Як правило, він пов'язаний із непередбачуваними ситуаціями, здебільшого аварійними та ризикованими.

Чорний колір у інтерфейсі додатку – це, переважно, написи різних типів та умовні позначення. Так як він є показником елегантного дизайну, це зробить застосунок більш сучасним.

Білий колір – це відповідно більше фоновий елемент, що є показником використання не лише сучасних трендів, а й ще є корисним у сприйнятті людей з вадами зору та загалом позитивно впливає на зорове сприйняття екрану.

Назва та логотип додатку

Згідно ролі додатку на ринку програмних рішень та його основного функціоналу, було прийнято рішення про вибір назви «AutoSOS».

Зміст, що закладений у сформовану назву, звучить, як негайна допомога авто у будь-який час та за найбільш швидкі терміни.

SOS – це міжнародний сигнал лиха, що використовується в усьому цивілізованому світі, який сповіщає про небезпеку, катастрофу або ж аварію, що загрожують життю людей [13].

Отже, можна зробити висновок про доцільність обраної назви, так як вона характеризує, по-перше, основне призначення додатку, по-друге, легко запам’ятовується та є невід’ємною частиною усього дизайну застосунку.

Логотип, сформований при розробці додатку, наведено на рисунку 2.1, що розташований нижче.

Рисунок 2.1 – Логотип мобільного додатку «AutoSOS»

Наступний етап – це безпосереднє відтворення UI-дизайну проекту для реалізації роботи служби евакуації авто.

UI-дизайн додатку

Спочатку потрібно навести деякі вимоги, необхідні для якісного формування UI-дизайну, а саме:

- ясність (немає двозначності, текст і структура чітко зрозумілі);
- лаконічність (інтерфейс не перевантажений підказками, спливаючими вікнами і анімацією);
- впізнаваність (елементи дизайну легко розпізнати, навіть при знайомстві з додатком);
- сталість (елементи інтерфейсу, меню та окремі кнопки повинні вести себе однаково на будь-якій сторінці додатку);
- ефективність (створений інтерфейс повинен заощаджувати час користувача і швидко виконувати його потреби) [15].

Основні вимоги до розробки UI-дизайну сформовані. Загальний вигляд інтерфейсу мобільного додатку було реалізовано за допомогою сервісу «Canva».

«Canva» – це онлайн-інструмент для створення дизайну і публікації матеріалів, завдання якого полягає у наданні можливості створювати будь-які дизайнерські рішення та безкоштовно публікувати їх.

Демонстрацію результатів роботи розпочинає вікно завантаження мобільного додатку «AutoSOS» та початкова сторінка застосунку, де необхідно обрати відповідну користувацьку роль.

Від вибору ролі (клієнт, водій евакуатора) будуть залежати і наступні вікна мобільного додатку, що буде продемонстровано у наступних розділах відповідно.

Рисунок 2.2 – Вікна завантаження та вибору ролі додатку «AutoSOS»

Вікна реєстрації нового користувача та подальшої його авторизації наведені на наступному рисунку 2.3. Так як поля, які необхідно заповнити як

для клієнта, так і для водія однакові, то було вирішено показати лише один із двох наявних варіантів.

Поля, які необхідно заповнити, а саме ім’я користувача, його номер телефону, марка або власного авто, бо ж евакуатора, а, також, його державний номер потрібні для подальшого коректного функціонування застосунку.

У кожному полі є відповідні підказки, ціль яких полегшити користування мобільним додатком та зробити його більш зручним у використанні та роботі із ним.

Для успішної авторизації потрібні електронна пошта та сформований користувачем при реєстрації пароль.

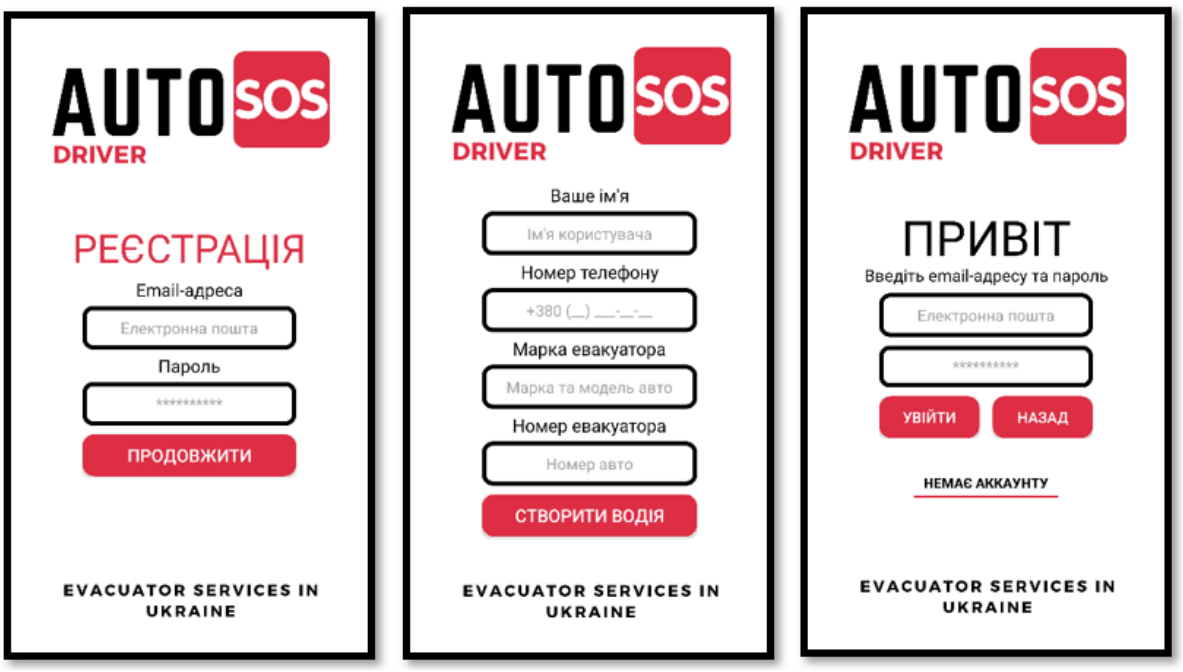

Рисунок 2.3 – Вікна реєстрації та авторизації додатку «AutoSOS»

Після успішної реєстрації та подальшої авторизації користувачем буде здійснено перехід на початкове вікно програми, де розташована карта заданої місцевості, відслідковується поточне місцезнаходження користувача та присутня можливість викликати евакуатор.

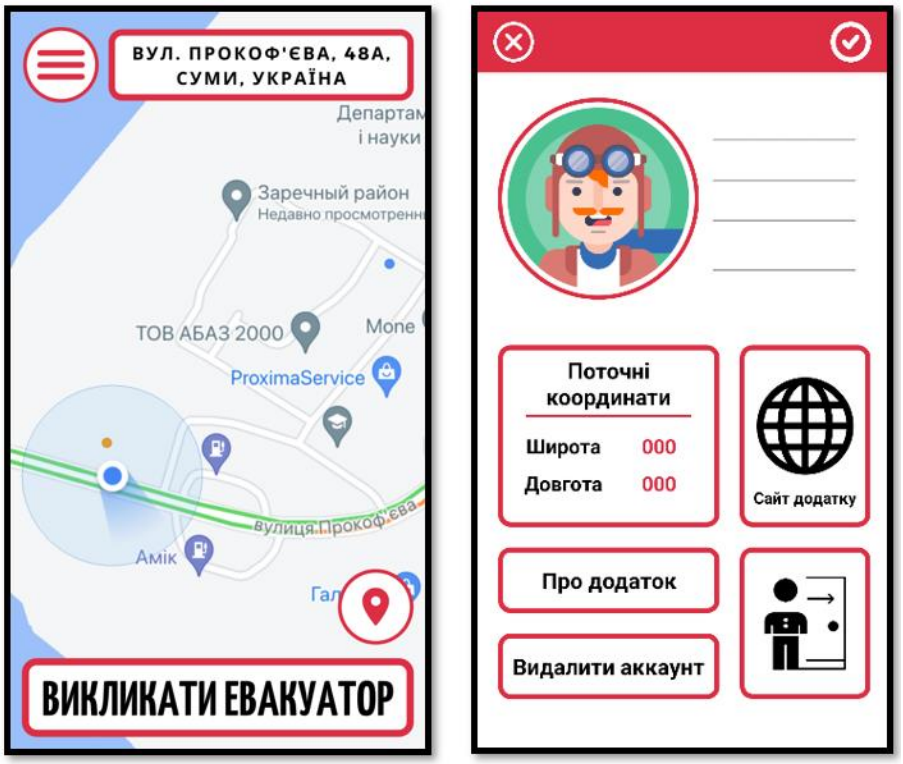

Рисунок 2.4 – Головне вікно та меню додатку «AutoSOS»

Також, на рисунку 2.4 можна побачити меню користувача, де при роботі додатка будуть відображені його особисті дані, можливість переходу на інші сторінки програми, а, також, реалізовано процедуру виходу та повного видалення існуючого аккаунту.

Отже, можна побачити, що майбутній вигляд основних вікон додатку, що розробляється, успішно наведено. Усі вони поєднані спільною ідеєю та рисами, такими як основні кольори, однаковий дизайн та поведінка кнопок, зручний та нескладний дизайн.

2.2. Вибір середовища розробки

Мобільний додаток «AutoSOS», призначений для функціонування евакуаторної служби, розроблено на базі операційної системи Android.

Android – це операційна система, яка слугує потужною платформою для мобільних телефонів та планшетних комп'ютерів [1].

Безпосередня розробка відбувалася у спеціалізованій для цього програмі «Android Studio». Це редактор коду, що дає можливість працювати як з дизайном додатку, так і розробляти його функціонал. Також, програма має власний емулятор, який дозволяє запускати Android-застосунки на базі ОС Windows.

Вибір середовища розробки був сформований не лише за рахунок вибору технологій та відповідних мов програмування, а й через високу популярність даної ОС на території України.

Так, в Україні, як і в інших країнах, смартфони на базі Android охоплюють більшу частину вітчизняного ринку. Цією операційною системою користуються майже 80% від усіх наявних користувачів.

Безперечно, частка людей, що користуються смартфонами на базі операційної системи iOS зростає кожного року, та, навіть, для так званого паритету у кількості користувачів повинно пройти ще багато років.

Звичайно, в майбутньому додаток, що розробляється буде адаптований і під інші ОС, але зараз він буде працювати лише на базі відповідної системи.

2.3. Огляд технологій

У даному пункті наведено перелік технологій, використаних протягом розробки даного мобільного додатку.

Згідно до пункту 2.2 застосунок буде функціонувати на базі операційної системи Android.

Тепер потрібно розглянути відповідні мови програмування та спеціальне програмне забезпечення, необхідне для подальшої розробки.

Мова програмування Java

Основою мобільного додатку, що розробляється, стала мова програмування Java, яка дозволила відтворити загальну логіку застосунку,

взаємодію його елементів між собою та встановила зв'язок між програмним кодом та зовнішнім відповідним ПО.

Java – це об'єктно-орієнтована мова програмування, випущена 1995 року «Sun Microsystems», а з 2009 року мовою займається компанія «Oracle». Java-програми компілюються у байт-код, що при власному виконанні інтерпретується віртуальною машино, відповідно до вимог конкретної платформи [2].

Мова розмітки XML

Зовнішній інтерфейс додатку разом зі стилем кнопок, текстових полів тощо був розроблений на основі мови розмітки XML.

Мова розмітки XML – це запропонований консорціумом W3C стандарт побудови мов розмітки даних, що ієрархічно структуровані для більш швидкого обміну між різними застосунками відповідно [7].

Обробляти XML-документ відносно легко, так як він має наступні характеристики:

- простий у розумінні текст, що легко оброблюється;
- може бути перетворений на інші формати;
- файли є ієрархічними [6].

Наступний етап – це опис програмного забезпечення відповідно, що використовувалося при розробці додатку.

Платформа для розробки мобільних додатків Firebase

Firebase – платформа, призначення якої це швидка розробка мобільних додатків різних типів. Головна перевага – полегшення роботи розробника з backend-частиною.

Firebase, перш за все, це одне із BaaS-рішень (Backend as a Service), яке пропонує велику кількість можливостей, таких як віддалений сервер, бази даних різних типів, хостинг та аутентифікація.

Так, наприклад, Firebase Realtime Database надає розробникам власно розроблений API, що повністю забезпечує формування бази даних та її безпосереднє підтримання.

Отже, можна побачити, що саме це спеціалізоване ПО допомогло при розробці додатку з авторизацією користувачів, а, також, засобами саме Firebase Realtime Database було створено власну БД для повноцінного функціонування застосунку.

Інтерфейси програмного забезпечення Google APIs

Google APIs – це інтерфейси програмного забезпечення, розроблені компанією «Google», призначення яких полягає у взаємодії вашого додатку із службами Google, що виконують задачі різних типів [8].

При розробці мобільного додатку для обслуговування евакуаторної служби була використана саме карта на основі Google Maps API. Її було інтегровано у додаток за допомогою отриманого при реєстрації спеціалізованого відповідного API-key. Усі дії із знаходження позицій розташування, як водія авто, так і водія евакуатора, прокладення маршруту, визначення координат було відтворено саме на карті Google.

Емулятор смартфону Genymotion

Для тестування правильності роботи мобільного додатку та моделювання ситуації, які б могли виникнути у його користувачів при взаємодії з ним, необхідно було запускати застосунок засобами емулятора, який би дозволив робити це на базі ОС Windows.

Для цього, у якості зовнішнього програмного забезпечення, було обрано саме Genymotion. Це емулятор, який можна запускати локально на своєму ПК. Його призначення – це тестування додатків різних типів, процесу ігор та моделюванні відповідних процесів у середовищі Android.

Підсумовуючи огляд технологій, що були застосовані при розробці мобільного додатку «AutoSOS» для реалізації процесу роботи евакуаторної служби, можна зробити висновки, що використання мови програмування Java, мови розмітки XML разом із застосуванням Firebase та Google APIs дало змогу створити повноцінний застосунок, який має процедуру реєстрації та подальшої авторизації, базу даних, яка містить дані про користувачів, а, також, геоінформаційну систему (ГІС), що потрібна для роботи з картами, відслідковування розташування користувачів та взаємодії між ними.

Отже, огляд технологій та їх можливостей успішно наведено.

3. ПРАКТИЧНА РЕАЛІЗАЦІЯ

У даному розділі викладено інформацію про логіку роботи мобільного додатку відповідно до ролі, що використовується, код програми та його особливості, опис основних блоків та можливі оновлення функціоналу у майбутньому.

3.1. Розробка моделі інформаційної системи

У пункті 3.1 відтворено модель інформаційної системи у вигляді відповідних схем, задача яких відобразити логіку роботи мобільного застосунку, що розробляється.

Модель ІС – це сукупність інформації, що характеризує певні властивості і стани програмного продукту, процесу, призначення якої відтворити процеси переходу та взаємодії між окремими блоками додатку [11].

Спочатку відобразимо схемою логіку роботи з додатком у ролі клієнта, якому потрібна допомога евакуаторної служби.

У результаті аналізу рисунку 3.1, на якому продемонстровано схему роботи з додатком у ролі клієнта, тобто замовника послуги, можна побачити логіку виконання послідовних процесів.

Також, потрібно зазначити, що мобільний додаток призначений не для виконання єдиного замовлення, а може працювати циклічно, що забезпечено необхідними для цього кнопками та переходами з одного вікна на інше і так далі.

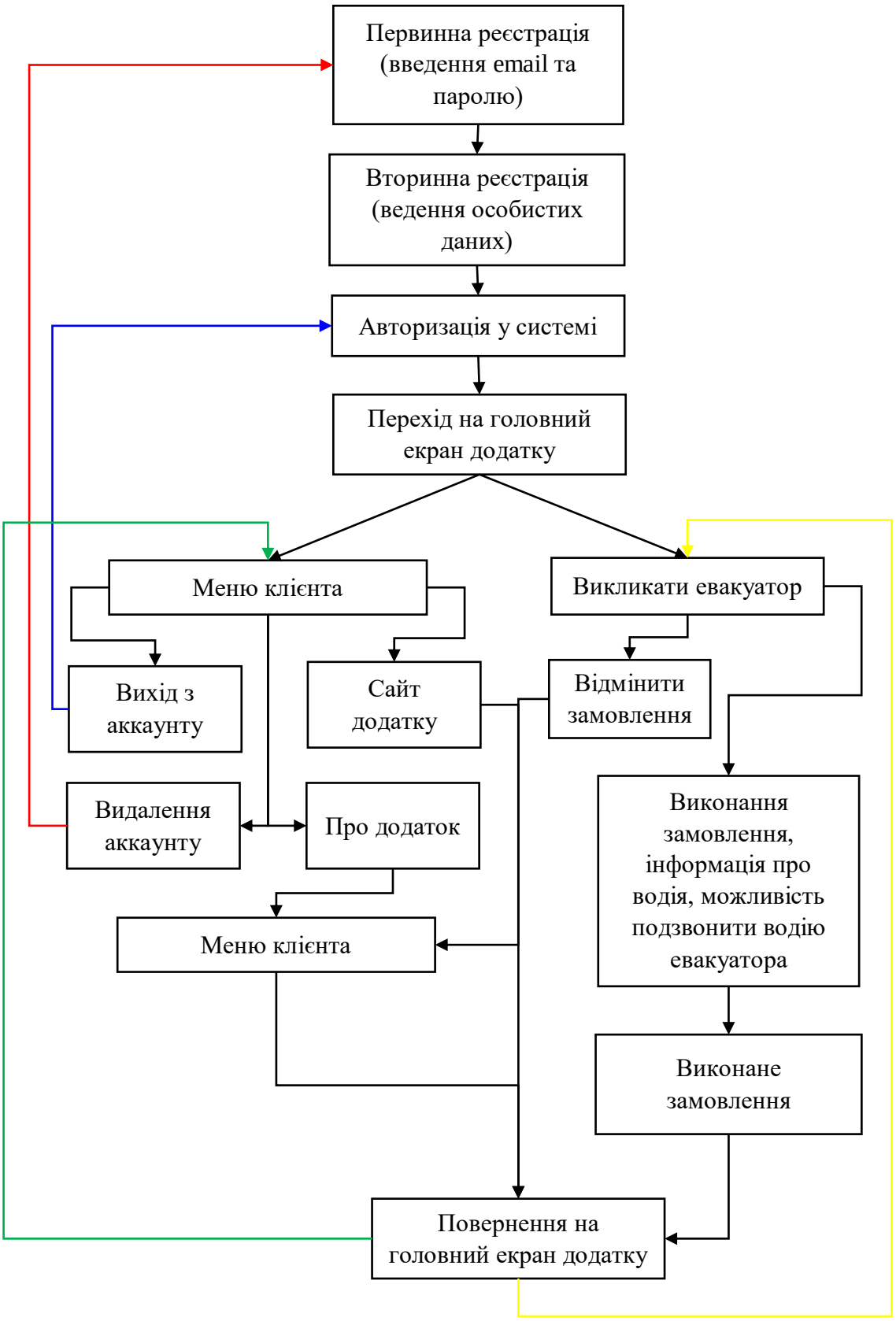

Рисунок 3.1 – Схема роботи з додатком у ролі клієнта

Після цього було відображено схемою логіку роботи з додатком у ролі водія евакуатора, завдання якого виконати замовлення, що надійшло.

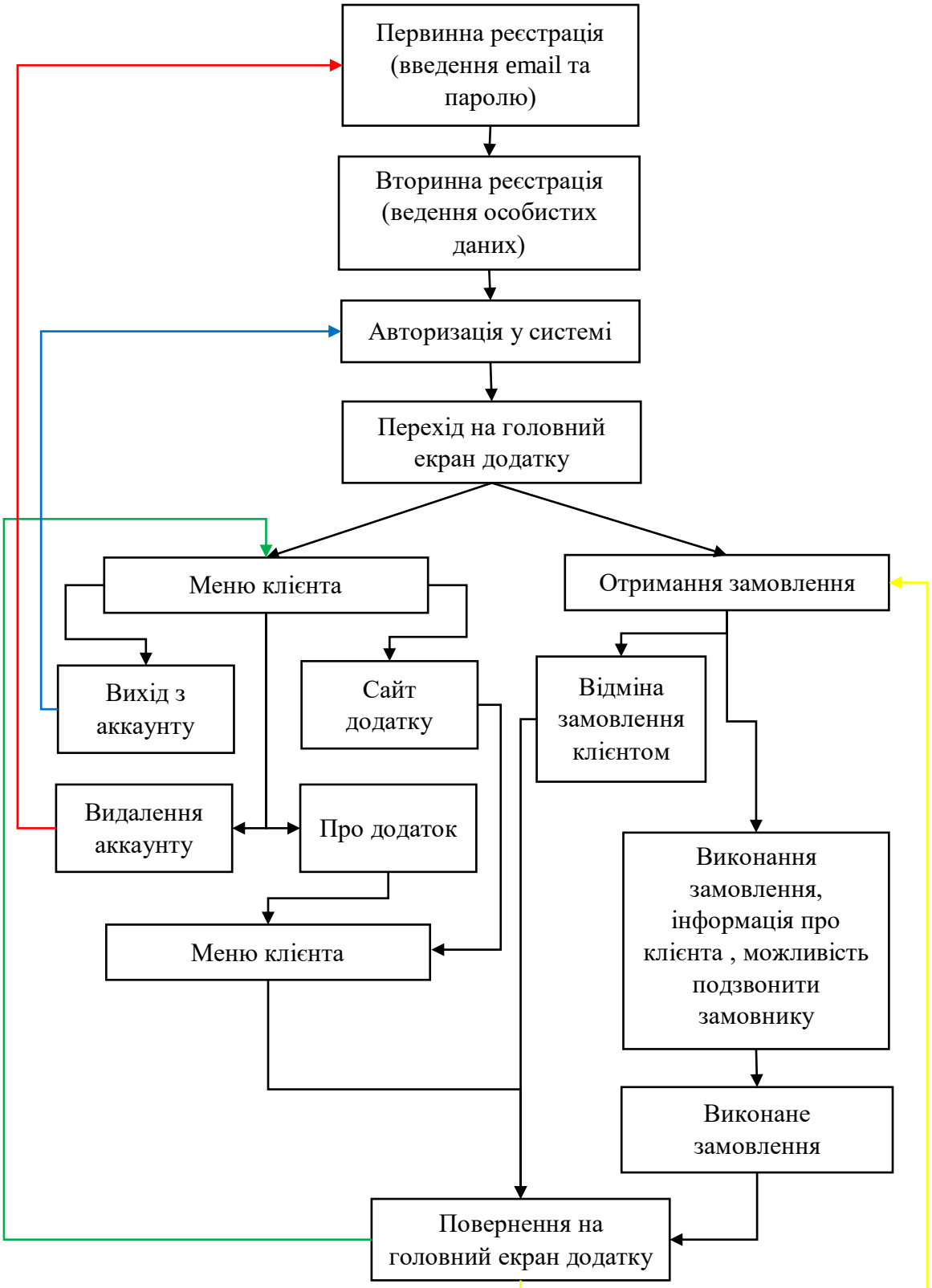

Рисунок 3.2 – Схема роботи з додатком у ролі водія евакуатора

Отже, можна побачити, що модель інформаційної системи у вигляді відповідних схем, задача яких відобразити логіку роботи мобільного застосунку, що розробляється, успішно відтворена. Перейдемо до формування наступного пункту відповідно.

3.2. Розробка СУБД

Для повноцінної та, звичайно, коректної роботи мобільного додатку «AutoSOS», необхідно використовувати базу даних, яка працюватиме та оброблятиме дані у режимі реального часу, забезпечуючи потреби як клієнтів, так і водіїв евакуаторів.

Для вирішення поставлених задач було обрано, як програмне рішення, Firebase Realtime Database відповідно.

Рисунок 3.3 – Створена база даних для роботи із додатком «AutoSOS»

На рисунку 3.3 можна побачити вже створену базу даних, розміщену на сервері, що розташовується на території США, та здатну працювати із запитами користувачів додатку.

Зараз вона порожня, та протягом опису основних блоків додатку та симуляції роботи застосунку БД буде заповнюватися, а результати будуть продемонстровані за допомогою відповідних скріншотів.

3.3. Програмна реалізація та опис основних блоків додатку

Код основних блоків програми, що включає у себе як опис функціоналу додатку, так і його зовнішній вигляд разом із системними файлами,

необхідними для налаштування застосунку, наведено у додатках А, Б, В та Г відповідно.

Також, можна ознайомитись із програмною реалізацією додатку, структурою проекту та матеріалами, перейшовши за посиланням на відповідний репозиторій GitHub:

- https://github.com/Bubenshchykov-IT/AutoSOS-Mobile-App

Опис основних блоків створеного додатку сформований на базі моделювання роботи з додатком одночасно як клієнта, так і водія евакуатора. Цей процес буде супроводжуватись пояснювальними коментарями та відповідними скріншотами. Перед початком моделювання, необхідно зазначити, що створена СУБД порожня і авторизованих користувачів у ній немає.

Отже, спочатку нам потрібно запустити емулятор смартфону на базі ОС Android. Це було зроблено за допомогою спеціалізованої програми Genymotion.

Спершу потрібно налаштувати емулятор, що запустився, тобто, необхідно включити GPS-передачу даних та увімкнути дозвіл на використання поточної локації та можливості здійснення дзвінків засобами додатку «AutoSOS».

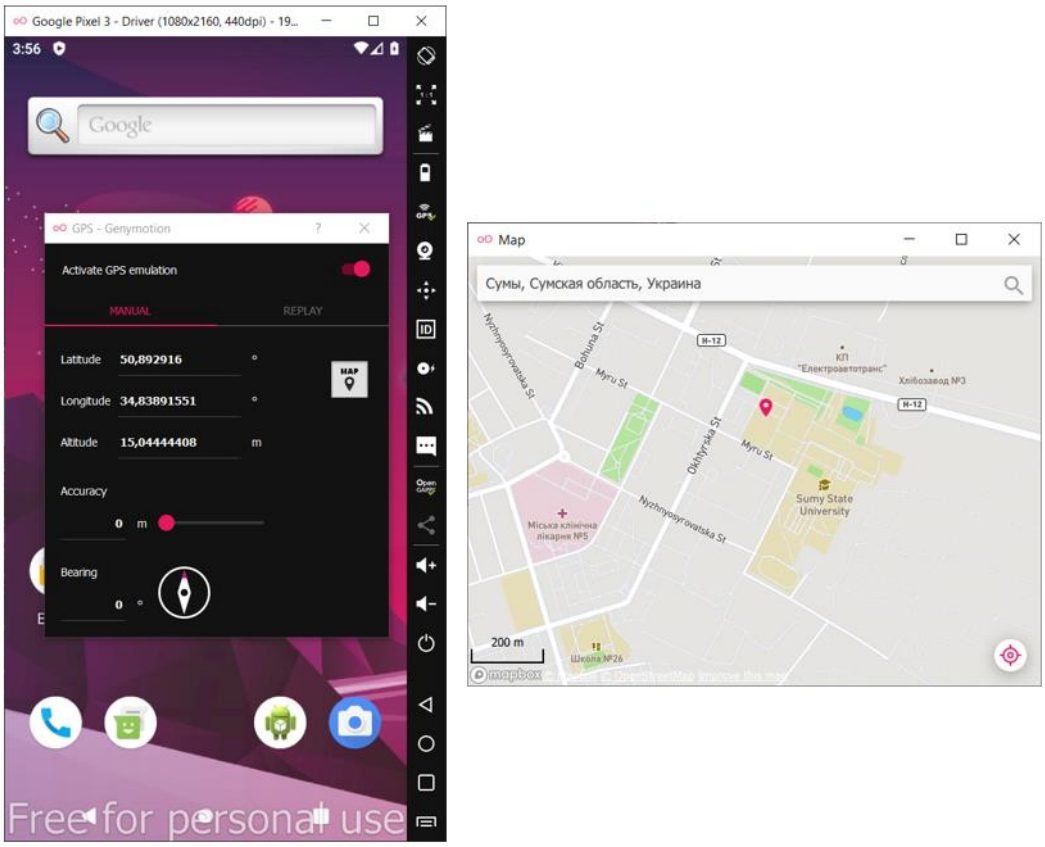

Рисунок 3.4 – Процес налаштування GPS-емуляції на смартфоні

На рисунку 3.4 можна побачити, що поточне місцезнаходження користувача успішно налаштоване. Мітка з адресою розташована на території Сумського державного університету у місті Суми, Сумська область, Україна.

Після цього було реалізовано дозвіл на використання GPS-даних додатком та налаштовано можливість відтворення дзвінків з додатку, що розробляється, та продемонструємо отримані результати на рисунку 3.6.

Для цього необхідно запустити додаток «AutoSOS» на емуляторі та перейти у налаштування програми відповідно.

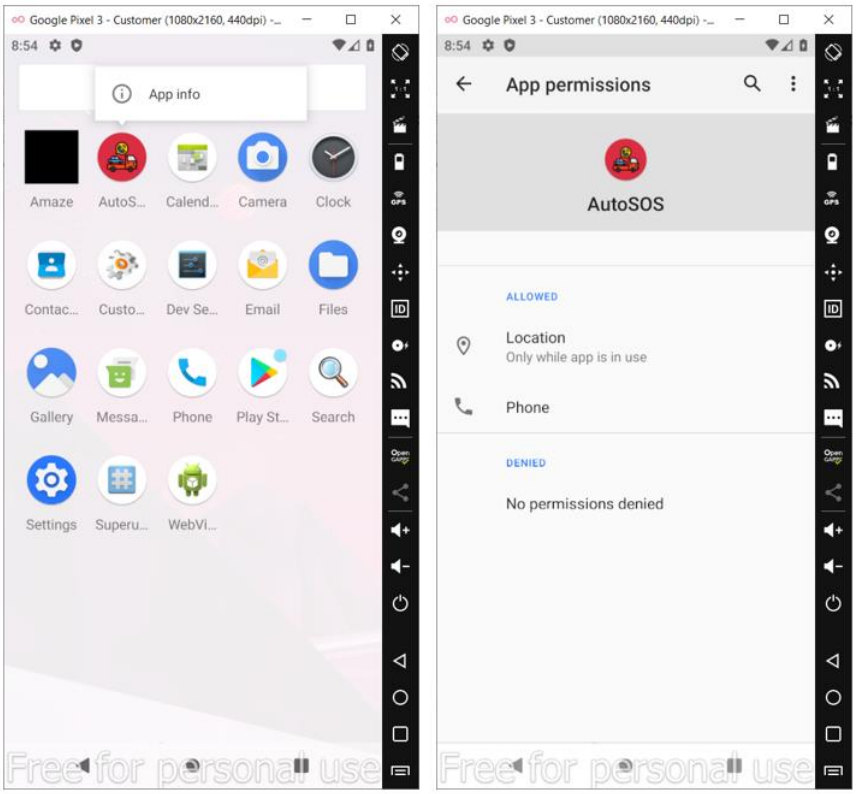

Рисунок 3.5 – Процес налаштування дозволів додатку «AutoSOS»

Після необхідних нам налаштувань потрібно запустити додаток та зареєструвати нового користувача. Процес моделювання було розпочато із створення аккаунту водія евакуатора.

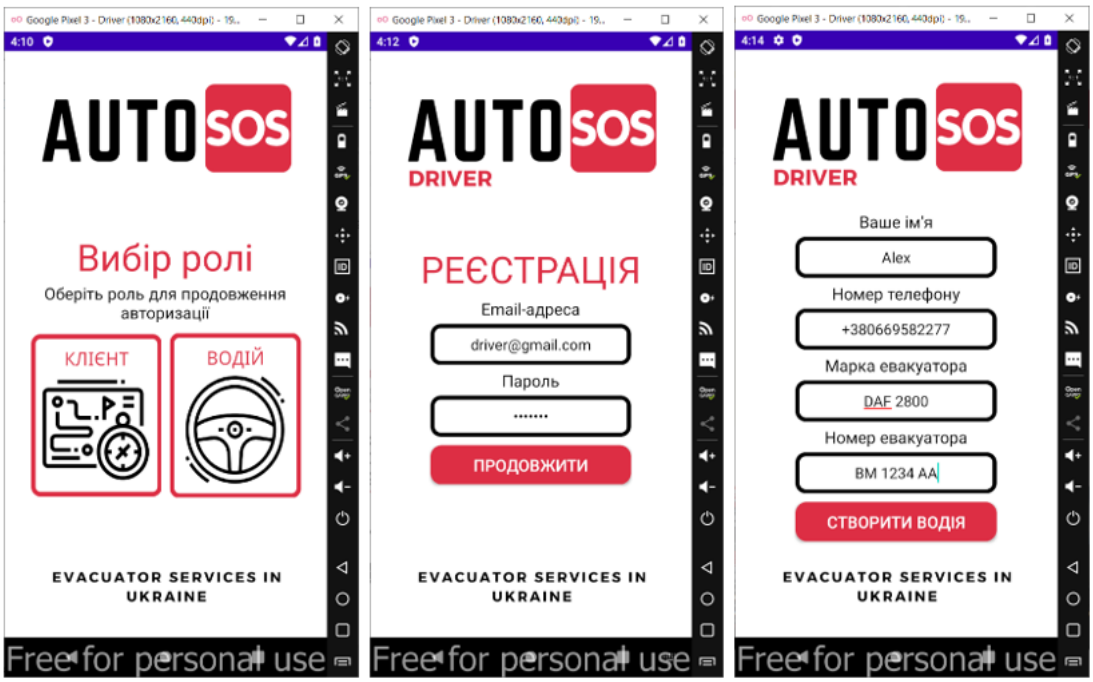

Рисунок 3.6 – Процес реєстрації водія евакуатора у додатку «AutoSOS»

На рисунку 3.6 можна побачити відтворений процес реєстрації нового користувача у ролі водія евакуатора, а саме вибір ролі, а, також, заповнення усіх необхідних полів, таких як електронна пошта, пароль, ім’я, номер телефону для подальшого зв’язку та дані про евакуатор (марка, модель та державний номер).

При успішній реєстрації дані про користувача повинні з’явитися у створеній раніше СУБД, а водія авто успішно авторизовано.

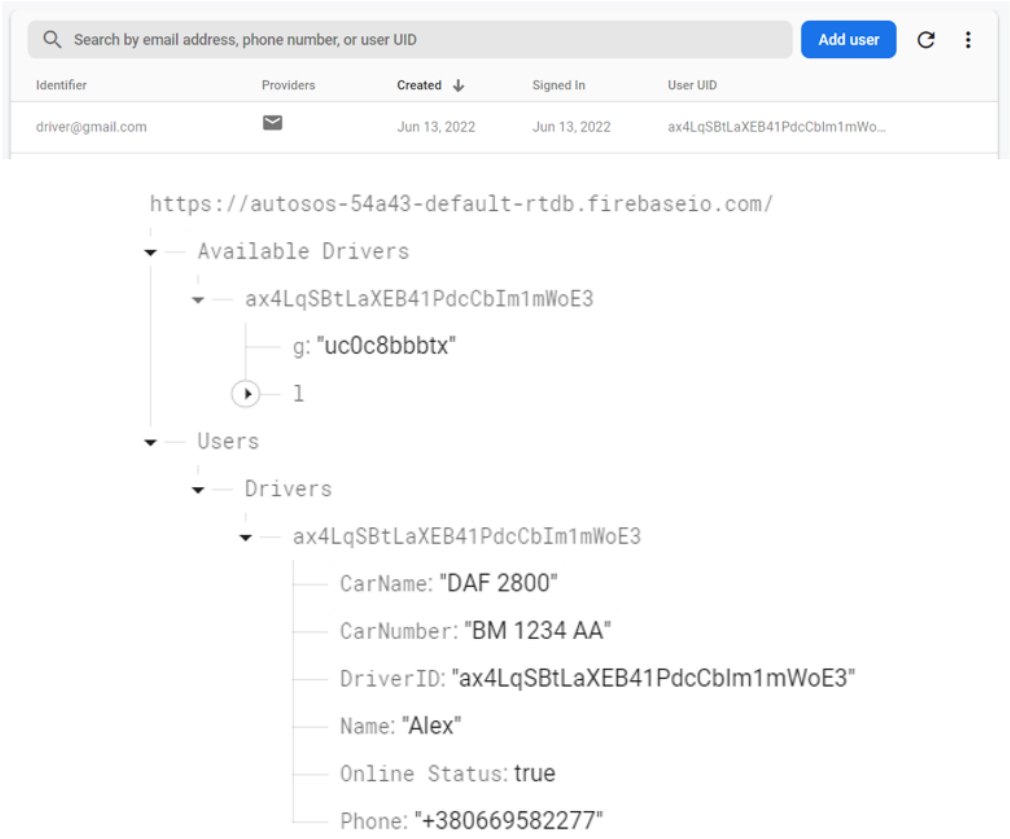

Рисунок 3.7 – Дані про зареєстрованого водія евакуатора у БД мобільного додатку «AutoSOS»

На рисунку 3.7 можна побачити, що у результаті успішної реєстрації нового водія евакуатору, його особисті дані було авторизовано та занесено до бази даних.

Так, були утворені нові вузли та сформована ієрархія «Users → Drivers → Driver_ID» відповідно. Варто зазначити, що спеціальний ідентифікатор

водія було створено автоматично і він є унікальним. Також, можна побачити, поля із даними користувача, що були щойно занесені відповідно до рисунку, який наведено вище.

На рисунку 3.7 відображено ще один вузол «Available Drivers», де розташований на рівень нижче вузол, що має назву ідентифікатора користувача та координати його поточного розташування (довгота та широта). Якщо поле «Online Status» є істинним і водій евакуатора не виконує наявне замовлення, він доступний до виконання нових.

Тепер, для перевірки авторизації користувача, потрібно вийти з аккаунту та продемонструвати процес входу водія евакуатора.

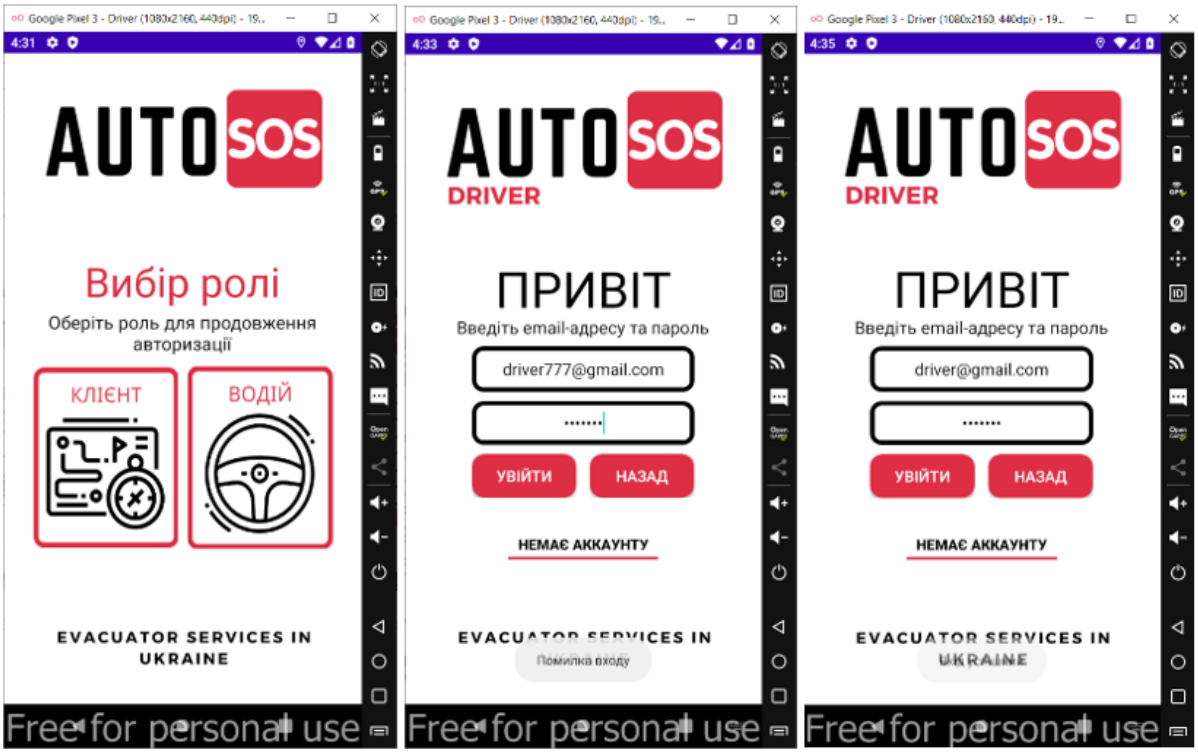

Рисунок 3.8 – Процес авторизації водія евакуатора у додатку «AutoSOS»

Для авторизації користувача необхідно, також, обрати роль, яка була вказана при реєстрації, та заповнити текстове поле з поштою та паролем. При помилковому виборі ролі, можна повернутися назад, використавши для цього однойменну кнопку.

Також, можна побачити, що при помилковому введені даних буде виведене спеціальне повідомлення «Помилка входу» і поля потрібно буде заповнити ще раз.

При успішній авторизації про це, спершу, буде повідомлене користувача, а потім вже відбудеться перехід на головний екран додатку, який відображено на наступному рисунку 3.9.

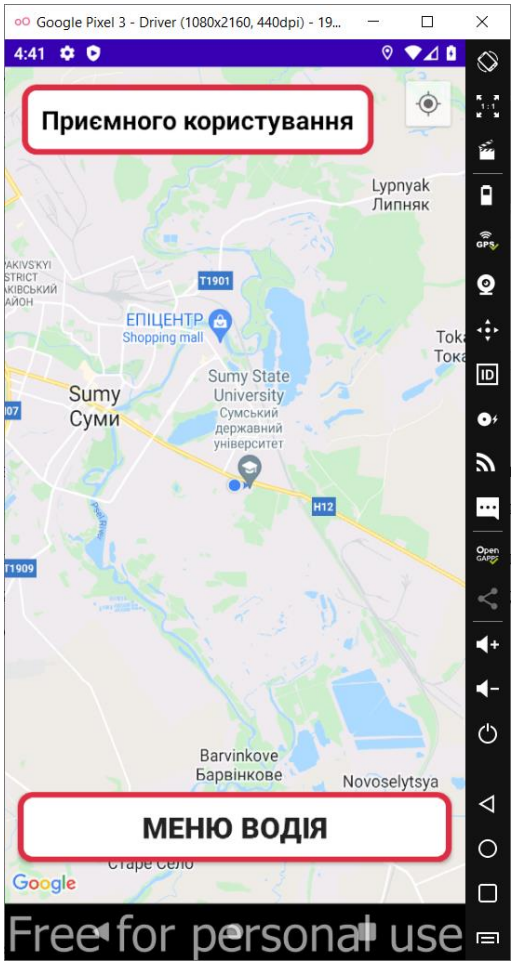

Рисунок 3.9 – Головне вікно додатку «AutoSOS» у ролі водія евакуатора

На рисунку 3.9 відображено головний екран додатку. Основу даного вікна складає карта місцевості, що відтворена засобами Google Maps. Також, можна побачити мітку користувача, яка вказує на його поточне місцезнаходження.

З головного вікна водія евакуатора можна перейти лише у меню водія, скориставшись для цього відповідною кнопкою. Також, реалізоване текстове

поле «Приємного користування». При виконанні замовлення його зміст буде змінюватися і показувати актуальну відстань до замовника, або ж клієнта.

Тепер потрібно здійснити перехід до меню водія і продемонструвати блоки та кнопки, що там розташовані.

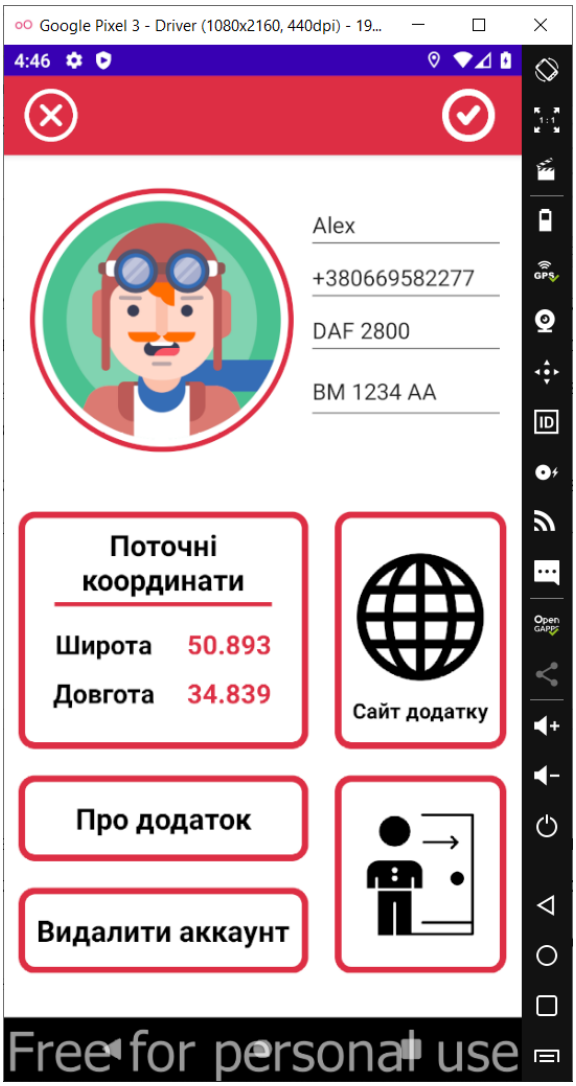

Рисунок 3.10 – Меню додатку «AutoSOS» у ролі водія евакуатора

На рисунку 3.10 можна побачити меню водія, яке надає можливість, по-перше, вийти з аккаунту, видалити його, по-друге, перейти на додаткові вікна, що мають назву «Про додаток» та «Сайт додатку» відповідно.

Також, на рисунку 3.10 наведені поля з особистими даними користувача. Їх можна змінити та зберегти результати.

Наприклад, була змодельована ситуацію, коли водій евакуатора з особистих причин змінив робочий номер телефону.

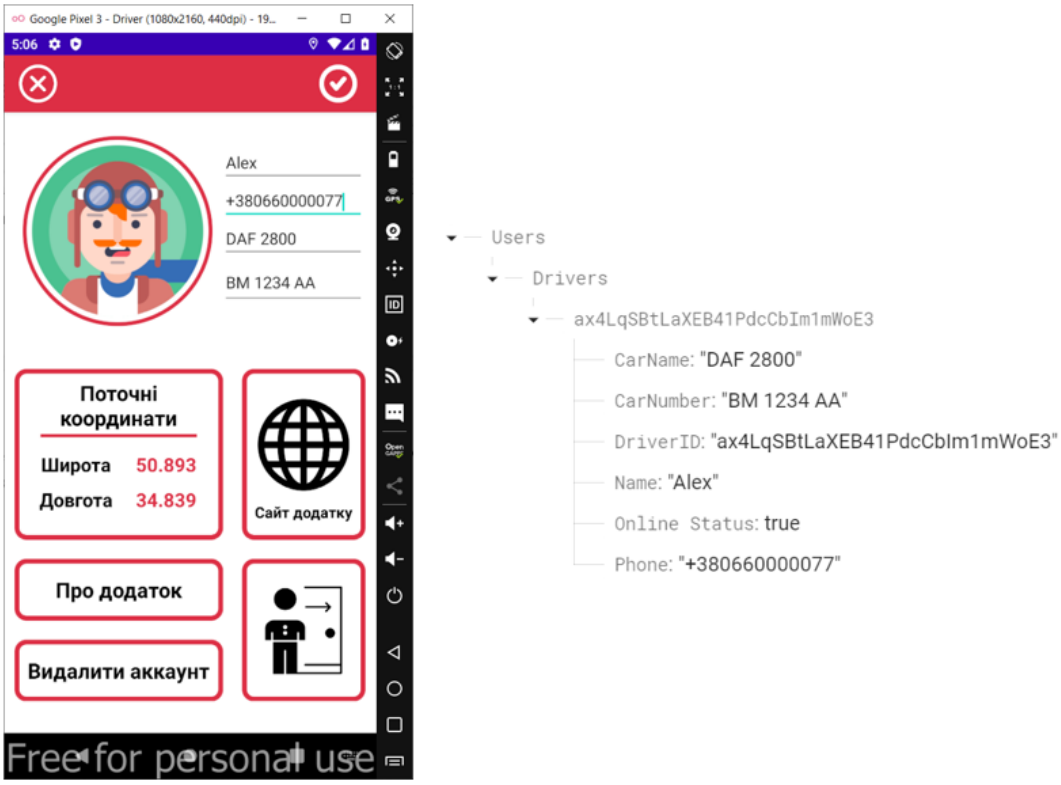

Рисунок 3.11 – Процес зміни особистих даних водія евакуатора у мобільному додатку «AutoSOS»

При натисканні кнопки «Зберегти» у правому верхньому куті екрану, номер телефону буде змінено, а користувача направлено на головне вікно. Також, результати зміни відображені і у вже існуючій базі даних.

При користуванні меню водія евакуатора, можна помітити певний інформаційний блок, де відтворено координати поточного розташування об’єкта. Даний блок несе виключно інформаційний характер.

Наступні вікна «Про додаток» та «Сайт додатку» наведено на рисунку 3.12, перехід на які можливий при натисканні на призначені для цього кнопки.

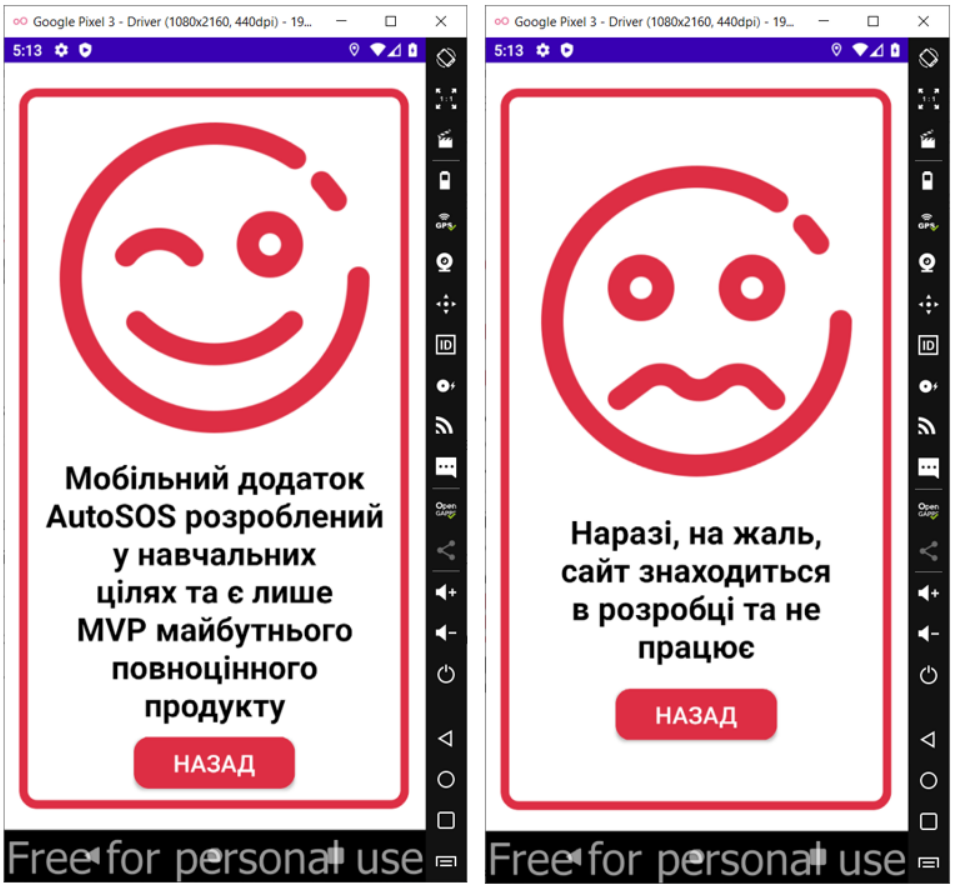

Рисунок 3.12 – Перехід на вікна «Про додаток» та «Сайт додатку» у ролі водія евакуатора у застосунку «AutoSOS»

На відповідному рисунку 3.12, продемонстровано інтерфейс вікон, що несуть виключно інформаційний характер. Після ознайомлення з наведеним матеріалом, користувач може повернутися знову у меню, натиснувши кнопку «Назад».

Отже, функціонал головного екрану водія успішно наведено. Також, описано меню водія евакуатора відповідно.

Для подальшої роботи додатка необхідно зареєструвати нового користувача, а саме клієнта або ж замовника.

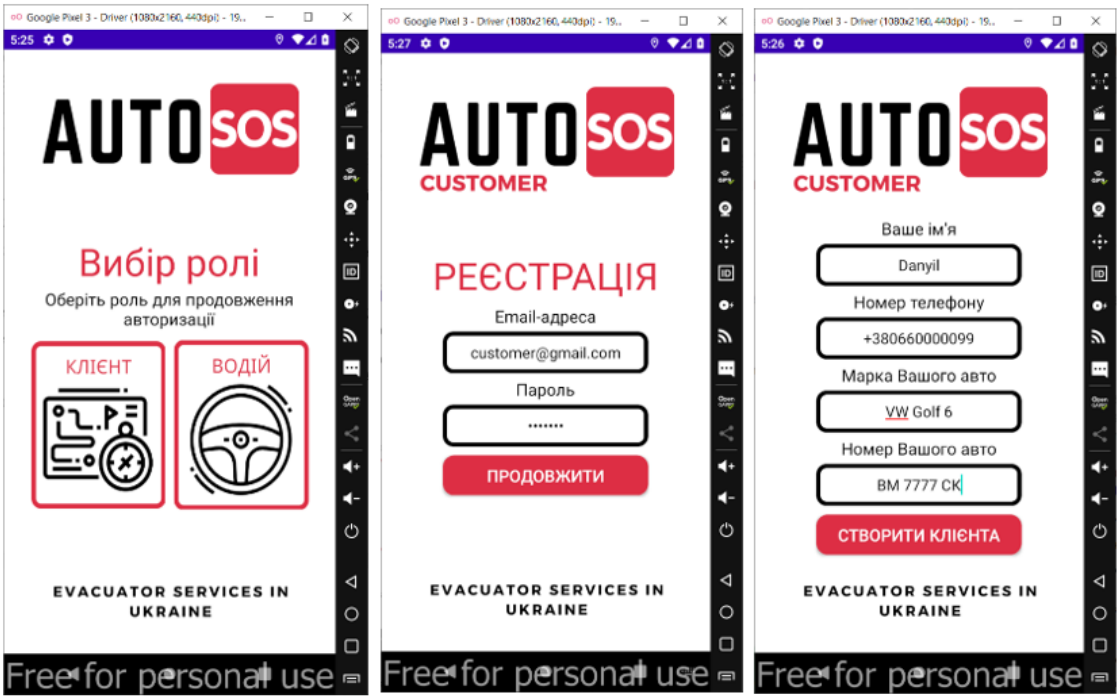

Рисунок 3.13 – Процес реєстрації нового клієнта у додатку «AutoSOS»

На рисунку 3.13 можна побачити відтворений процес реєстрації нового користувача у ролі замовника послуг.

При успішній реєстрації дані про користувача повинні з’явитися у створеній раніше СУБД, а водія авто успішно авторизовано.

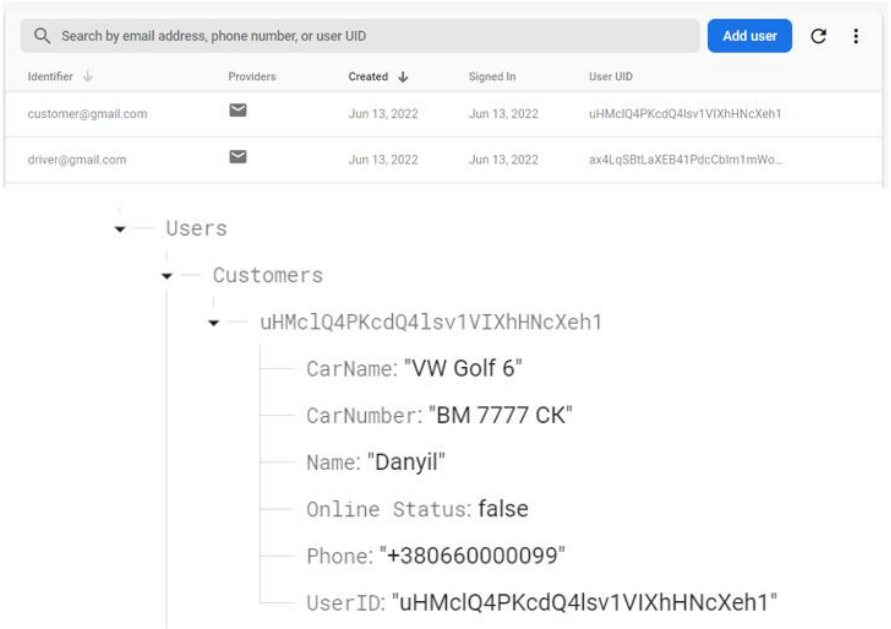

Рисунок 3.14 – Дані про зареєстрованого клієнта у БД додатку «AutoSOS»

На рисунку 3.14 можна побачити, що у результаті успішної реєстрації нового клієнта (замовника), його особисті дані було авторизовано та занесено до бази даних.

Так, були утворені нові вузли та сформована ієрархія «Users → Customers → User_ID» відповідно. Варто зазначити, що спеціальний ідентифікатор клієнта було створено автоматично і він є унікальним.

Також, можна побачити, поля із даними користувача, що були щойно занесені відповідно до рисунку 3.14.

Тепер, для перевірки авторизації користувача, потрібно вийти з аккаунту та продемонструвати процес входу саме клієнта відповідно.

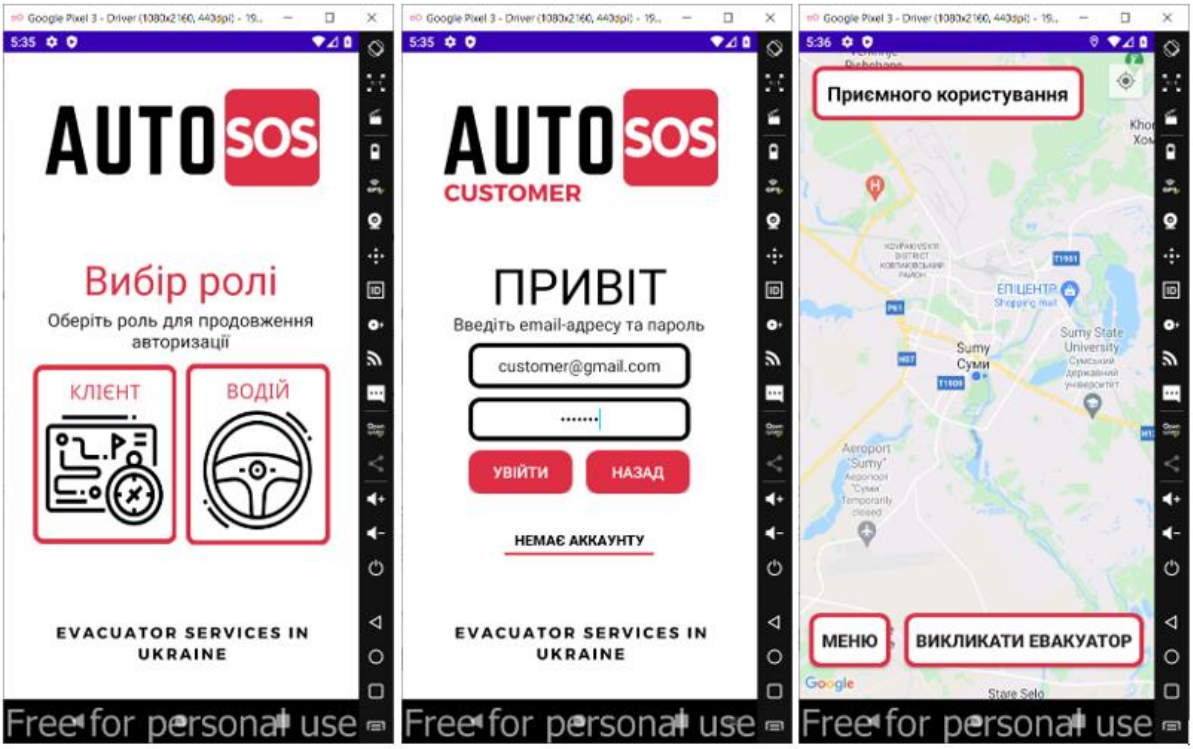

Рисунок 3.15 – Процес авторизації клієнта та відображення головного вікна додатку у ролі замовника

Для авторизації користувача необхідно, також, обрати роль, яка була вказана при реєстрації, та заповнити текстове поле з поштою та паролем. При помилковому виборі ролі, можна повернутися назад, використавши для цього однойменну кнопку.

Також, на рисунку 3.15 відображено головний екран додатку. Основу даного вікна складає, також, карта місцевості, що відтворена засобами Google Maps.

З головного вікна клієнта можна перейти у меню, скориставшись спеціальною кнопкою, та викликати евакуатор у разі потреби.

Реалізоване текстове поле «Приємного користування». При виконанні замовлення його зміст буде змінюватися і показувати актуальну відстань до водія евакуатора. Перейдемо до меню клієнта і продемонструємо блоки та кнопки, що там розташовані.

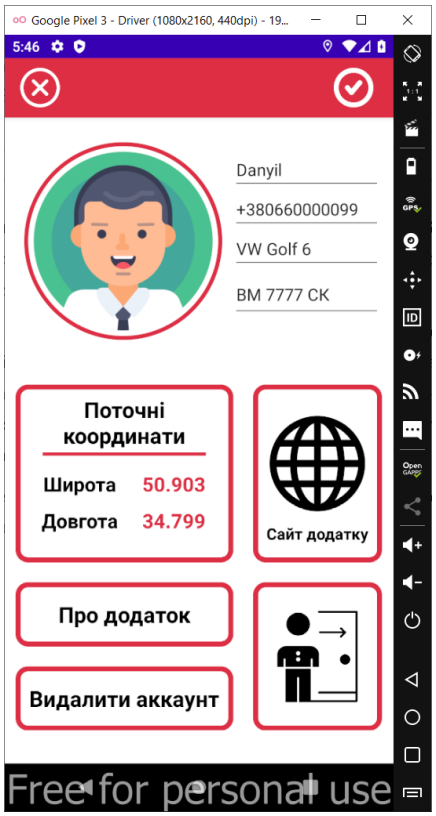

Рисунок 3.16 –Меню додатку «AutoSOS» у ролі клієнта (замовника)

На рисунку 3.16 можна побачити меню клієнта, яке надає можливість, по-перше, вийти з аккаунту, видалити його, по-друге, перейти на додаткові вікна, що мають назву «Про додаток» та «Сайт додатку» відповідно.

Також, на рисунку 3.16 наведені поля з особистими даними користувача. Їх можна змінити та зберегти відповідні результати.

Для подальшої роботи додатку необхідно запустити вже 2 емулятори смартфону задля демонстрації виконання замовлення як зі сторони клієнта, так і зі сторони відповідного водія евакуатора.

Для цього потрібно запустити спеціальну програму Genymotion та налаштувати 2 смартфони, тобто вказати відповідні дозволи та місцерозташування кожного із користувачів. Також, необхідно авторизуватися у ролі як водія, так і клієнта, якому потрібна допомога та змоделювати деяку аварійну ситуацію.

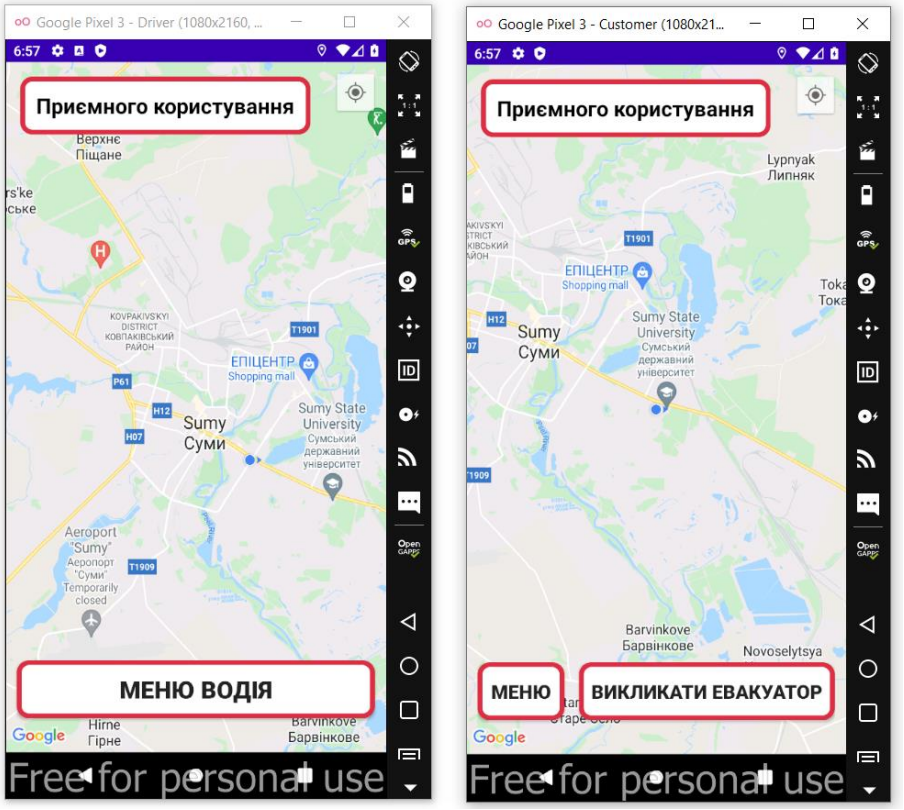

Рисунок 3.17 – Головні екрани водія евакуатора та клієнта перед початком замовлення

На рисунку 3.18 показано стан бази даних до початку моделювання ситуації, або ж до початку замовлення.

Рисунок 3.18 – Стан БД до початку замовлення клієнтом евакуатора

Для замовлення евакуатора, клієнту необхідно натиснути кнопку «Викликати евакуатор». Після цього інтерфейс екрану зміниться, а в БД сформується новий вузол «Customer Requests», що містить у собі унікальний ідентифікатор замовника та його поточні координати.

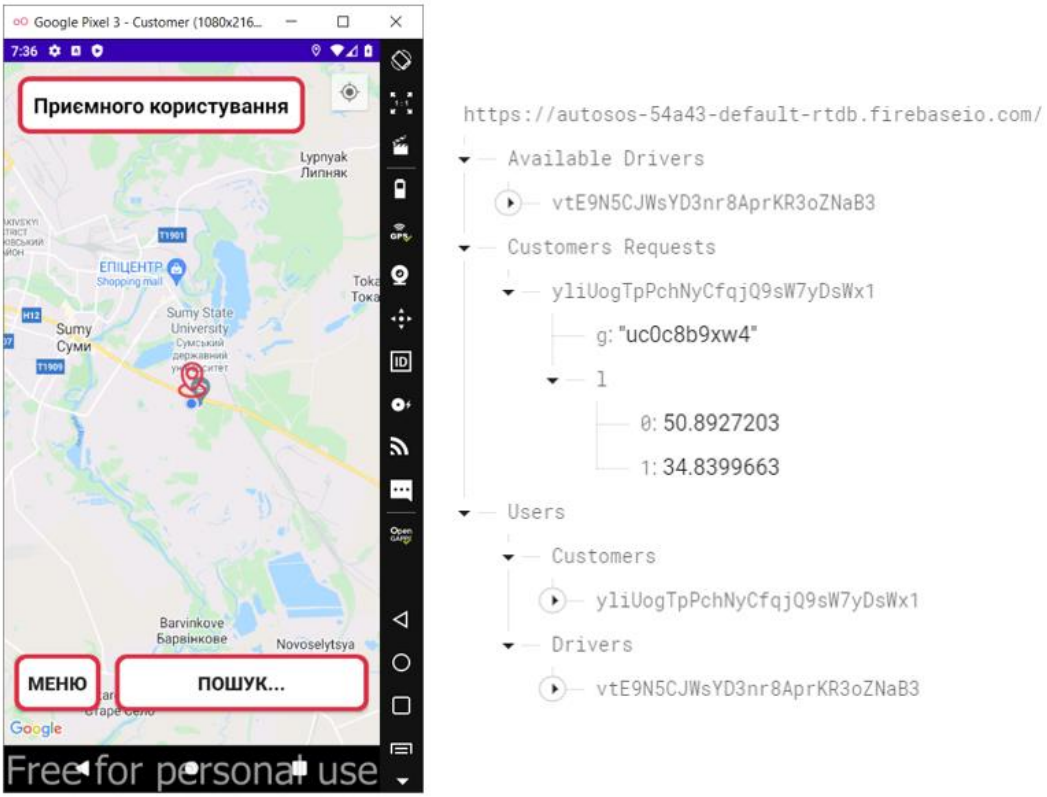

Рисунок 3.19 – Пошук евакуатора та зміни у БД під час пошуку клієнтом

Процедура пошуку сформована таким чином, що від поточного місцезнаходження клієнта формується спеціальне коло (зона) із початковим радіусом в 1 км. Якщо пошук невдалий, радіус з кожним кроком збільшується на 1 км, поки не знайде найближчого водія, який може прийняти замовлення.

Після закінчення пошуку, головні екрани як водія евакуатора, що отримав замовлення, так і клієнта, який це замовлення сформував, будуть доповнені відповідно до наступного рисунку 3.20.

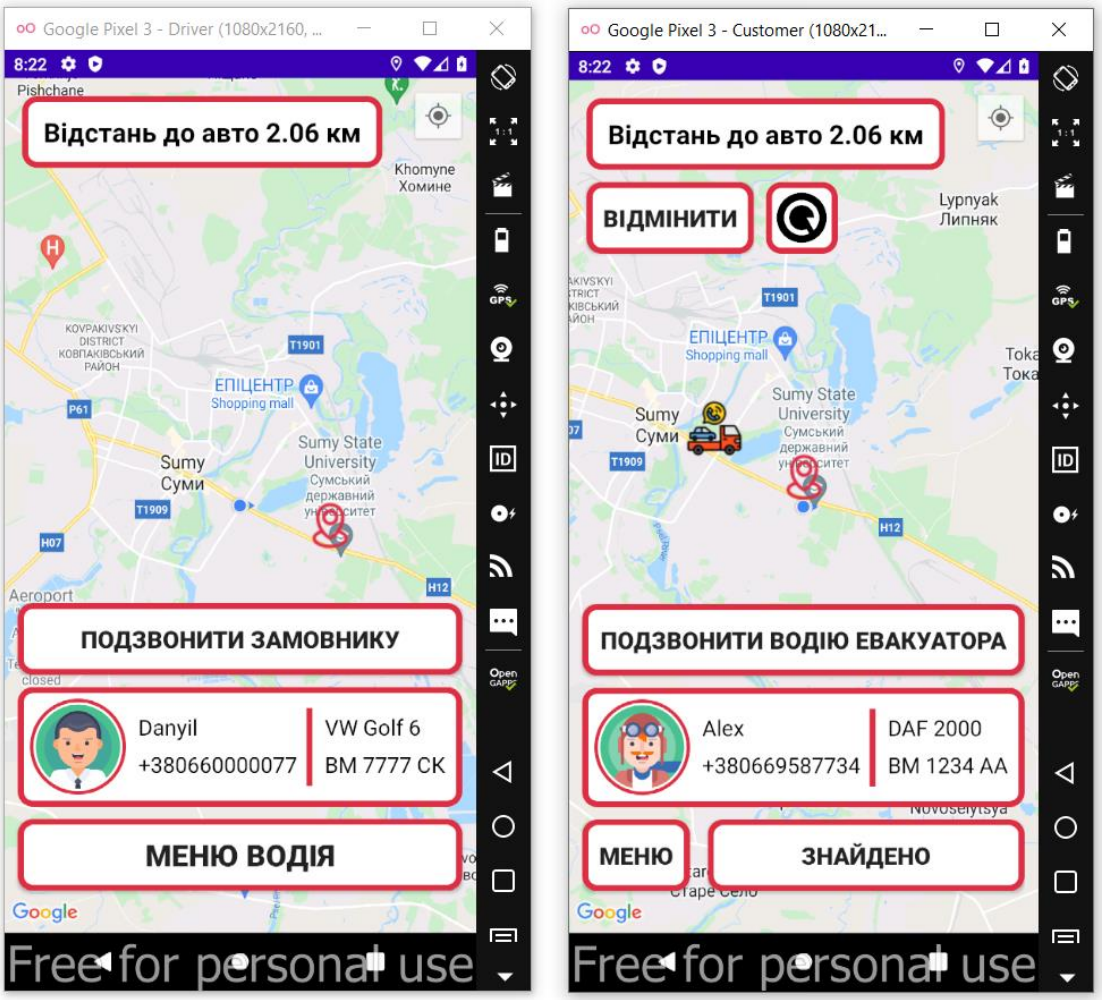

Рисунок 3.20 – Головні екрани клієнта та водія евакуатора після сформування замовлення

На рисунку 3.20 можна побачити, що головне вікно замовника, або ж клієнта видозмінилося, так як з’явився інформаційний блок, який містить

дані про водія, що повинен приїхати, текстове поле із відстанню між клієнтом та працівником. Також, за допомогою кнопок тепер виникла можливість відмінити поточне замовлення, перевантажити екран з метою відслідковування пересування водія евакуатора у реальному часі, і, якщо є така необхідність, подзвонити людині, що виконує поточне замовлення.

Майже так само видозмінився і екран водія. З'явилася можливість передзвонити клієнту, наприклад, для уточнення деталей замовлення. Варто зазначити, що можливість відмінити замовлення є тільки у замовника.

Як водій евакуатора, так і клієнт під час виконання замовлення не можуть перейти на вкладку меню, де вони могли б змінити особисті дані, вийти з аккаунту або видалити його.

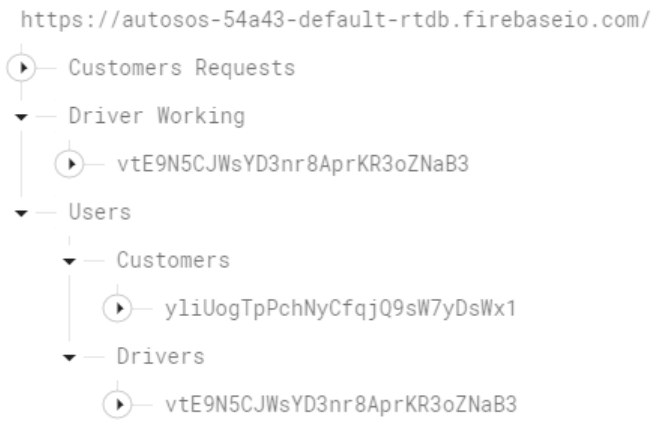

Рисунок 3.21 – Стан бази даних під час виконання замовлення

На рисунку 3.21 можна побачити стан сформованої раніше бази даних під час активної фази виконання замовлення. Так, вузол «Available Drivers» було автоматично видалено, так як єдиний, так би мовити, представник, а саме водій евакуатора був переміщений у щойно створений вузол «Driver Working», де його унікальний ідентифікатор було з'єднано з ідентифікатором клієнта, утворивши таким чином відповідний логічний ланцюг.

Натиснувши, наприклад, на кнопку «Відмінити» на головному екрані клієнта, як у випадку водія евакуатора, так і замовника, додаток автоматично

повернетеся на головний екран, і у водія знову з’явиться можливість приймати нові замовлення, а у клієнта їх відповідно формувати.

База даних повернеться у вихідне положення, а вузли «Customer Requests» та «Driver Working» будуть автоматично видалені.

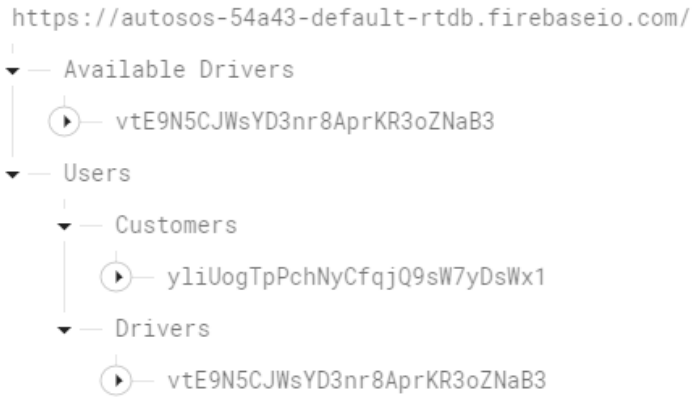

Рисунок 3.22 – Стан бази даних після скасування замовлення

Після цього було змодельовані процеси, які повинні виникнути після успішного виконання замовлення. Для цього потрібно зробити місцезнаходження як водія, так і клієнта однаковим.

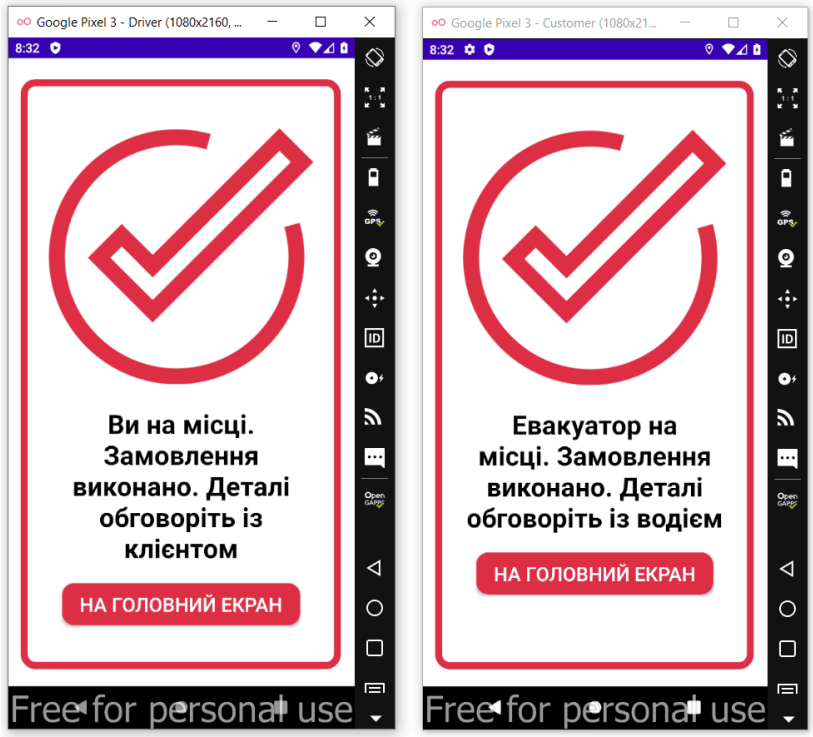

Рисунок 3.23 – Повідомлення про успішне виконання замовлення

Можна побачити, що коли дистанція між водієм евакуатора та клієнтом буде дорівнювати нулю, на екранах обох з’являться відповідні повідомлення про завершення, а база даних, згідно рисунку 3.23, повернеться у вихідне початкове положення, що дозволить водію працювати далі, а клієнту, наприклад, замовити послуги евакуатора ще раз.

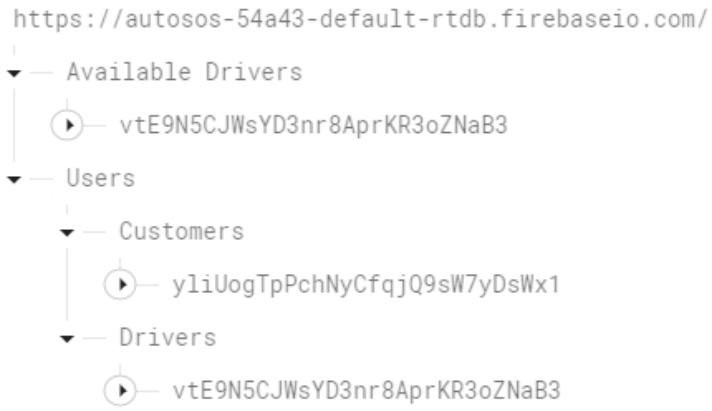

Рисунок 3.23 – Стан БД після успішного виконання замовлення

Результати видалення аккаунтів додатку «AutoSOS», а саме як реагує на це створена база даних та вікно авторизації користувачів у застосунку Firebase Console наведено на рисунку 3.24 відповідно.

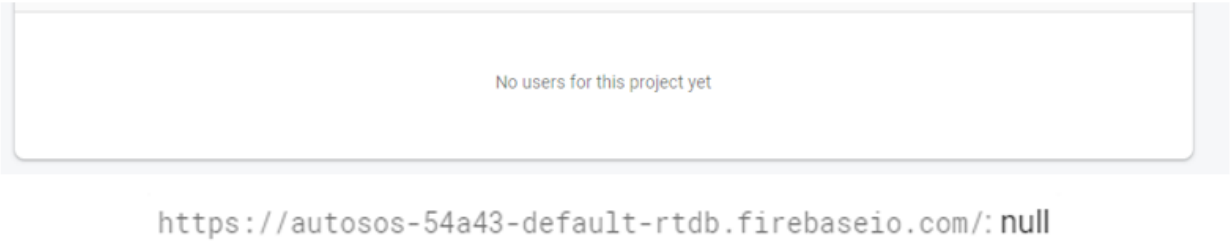

Рисунок 3.24 – Стан БД та вікна авторизації після видалення наявних користувачів

На рисунку 3.24 можна побачити, що вікно авторизації повністю порожнє, як і відповідна база даних. Це нормально і говорить лише про правильність роботи розробленого функціоналу, так як до процедури видалення у системі було зареєстровано усього 2 користувачі: клієнт (замовник) та водій евакуатора відповідно.

Отже, можна побачити, що процес опису основних блоків мобільного додатку шляхом моделювання ситуації, яка б в повній мірі показала функціонал та можливості додатку, успішно наведено.

Будь-який мобільний додаток постійно підтримується і вдосконалюється за рахунок додавання нового функціоналу, зміни інтерфейсу, розширення можливостей користувачів тощо. Мобільний застосунок «AutoSOS» не є винятком, і, також, потребуватиме подальших оновлень.

Так, зараз, це, так званий MVP, або мінімально життєздатний продукт, який може вже працювати на ринку, але потребує оновлень, що включатимуть розробку нового функціоналу та розширення можливостей як клієнтів, так і водіїв евакуаторів. Потрібно навести подальші зміни, які будуть реалізовані у майбутніх версіях додатку відповідно:

- приблизний час виконання замовлення;
- режим оплати готівкою або банківською картою, формування орієнтованої вартості замовлення;
- рейтинг замовника на основі його дій у минулому;
- система торгів вартості замовлення (замовник або система пропонує базову ціну з можливістю підвищувати її);
- під час виконання замовлення, відстеження маршруту пересування евакуатора із вже евакуйованим автомобілем.

Третій розділ успішно сформовано, як і відповідно основну частину даної кваліфікаційної роботи бакалавра.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи бакалавра було створено мобільний додаток «AutoSOS» для реалізації роботи евакуаторної служби, який відповідає як сформованому раніше UI-дизайну, так і поставленому технічному завданню відповідно.

Застосунок, що було створено, містить у собі основний функціонал та зручний і легкий у використанні інтерфейс, що дозволяє одному користувачу у ролі клієнта, або ж замовника, викликати евакуатор на допомогу, а іншому користувачу у ролі водія цього евакуатора виконати це замовлення та допомогти клієнту в аварійній ситуації.

Також, була створена система реєстрації нових користувачів та їх подальшої авторизації. Особисті дані як водіїв, так і клієнтів, що були необхідні для коректного функціонування додатку, розмішені у базі даних, яка має можливість працювати із запитами у реальному часі та швидко опрацьовувати їх.

Задля отримання результатів правильності роботи додатку, разом із описом його основних блоків було і відтворено процес його використання, де і було протестовано коректність роботи додатка загалом та окремо його основних модулів і блоків.

Також, було наведено список майбутніх оновлень, які будуть реалізовані у наступних версіях даного застосунку.

СПИСОК ЛІТЕРАТУРИ

1. Dawn Griffiths, David Griffiths. Head First Android Development: A Brain-Friendly Guide // O'Reilly Media; 1st edition, 2015. – 734 p.

2. Neil Smyth. Android Studio Bumble Bee Essentials – Java Edition: Developing Android Apps Using Android Studio and Java // Payload Media, 2022 – 1213 p.

3. Steven Branson, Chad Shoppa. UX/UI Design: Introduction Guide to Intuitive Design and User-Friendly Experience // 2020 – 150 p.

4. Gerardus Blokdyk. Google APIs A Complete Guide // 5STARCooks, 2020 – 240 p.

5. Alex Petrov. Database Internals: A Deep Dive into How Distributed Data Systems Work // O'Reilly Media; 1st edition, 2019 – 370 p.

6. Camilus Raynaldo. Android UI Design with XML // Kindle Edition, 2012 – 225 p.

7. J. Paul Cardle. Android App Development in Android Studio: Java + Android Edition for Beginners // Kindle Edition, 2017 – 202 p.

8. Evangelos Petroutsos. Google Maps: Power Tools for Maximizing the API // McGraw Hill; 1st edition, 2017 – 706 p.

9. Майкл Портер. Конкурентна стратегія. Техніки аналізу галузей і конкурентів // Наш Формат, 2020 – 424 с.

10. Авраменко В.С., Авраменко А.С. Проектування інформаційних систем, навчальний посібник // Черкаси: Черкаський національний університет ім. Б. Хмельницького, 2017 – 434 с.

11. Перевозчикова О.Л. Інформаційні системи і структури даних // Києво-Могилянська академія, 2017 – 288 с.

12. Полторак В.А., Тараненко І.В., Красовська О.Ю. Маркетингові дослідження // Центр навчальної літератури, 2017 – 342 с.

13. Музичук С. Сучасний орфографічний словник української мови (60 000 слів) // Кристал Бук, 2015

14. Інформаційна стаття «Топ-20 найпопулярніших українських мобільних застосунків» журналу «The Page».

Посилання: https://thepage.ua/ua/business/top-20-najpopulyarnishih-ukrayinskih-mobilnih-zastosunkiv.

15. Інформаційна стаття на тему «Що таке UX/UI-дизайн і як потрапити до цієї професії».

Посилання: https://skillbox.ru/media/design/ux_ui_dizayn_chto_eto_takoe/.

16. Інформаційна стаття на тему «UX/UI-дизайнер: від нуля до профі». Посилання: https://thecentralacademy.com/blog/ux-ui-designer-from-the-scratch-to-the-professional/.

17. Інформаційна стаття на тему «Що таке Firebase?». Посилання: https://avada-media.ua/ua/services/firebase/.

ДОДАТКИ

У даному розділі наведено вміст усіх додатків, сформованих для викладення відповідних блоків програмної реалізації, необхідної для коректного функціонування мобільного застосунку.

Додаток А

У даному додатку наведено зміст системного файлу «AndroidManifest» формату XML, де розташовані дозволи програми тощо.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.example.autosos">

    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.CALL_PHONE" />

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.AutoSOS"
        tools:targetApi="31">
        <activity
            android:name=".DriverOrderResultActivity"
            android:exported="false" />
        <activity
            android:name=".CustomerOrderResultActivity"
            android:exported="false" />
        <activity
            android:name=".CustomerMenuActivity"
            android:exported="false" />
        <activity
            android:name=".CustomerInfoRegistrationActivity"
            android:exported="false" />
        <activity
            android:name=".AboutActivity"
            android:exported="false" />
        <activity
            android:name=".SiteActivity"
            android:exported="false" />
        <activity
            android:name=".DriverInfoRegistrationActivity"
            android:exported="false" />
```

```
        <activity
            android:name=".CustomerRegistrationActivity"
            android:exported="false" />
        <activity
            android:name=".DriverRegistrationActivity"
            android:exported="false" />
        <activity
            android:name=".CustomerWelcomeActivity"
            android:exported="false" />
        <activity
            android:name=".DriverWelcomeActivity"
            android:exported="false" />
        <activity
            android:name=".RoleSelectionActivity"
            android:exported="false" />
        <activity
            android:name=".DriverMenuActivity"
            android:exported="false" />

        <meta-data
            android:name="com.google.android.geo.API_KEY"
            android:value="AIzaSyC0KaspzY1IW5lapVvbL3HLusOo-Bnvs1U" />

        <activity
            android:name=".CustomerMapsActivity"
            android:exported="false"
            android:label="@string/title_activity_customer_maps" />
        <activity
            android:name=".DriverMapsActivity"
            android:exported="false"
            android:label="@string/title_activity_customer_maps" />
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Додаток Б

У даному додатку наведено зміст системного файлу «build.gradle», де розташовані загальні налаштування та відображено перелік бібліотек, що були використані.

```
plugins {
    id 'com.android.application'
    id 'com.google.gms.google-services'
    id 'com.google.android.libraries.mapsplatform.secrets-gradle-plugin'
}

android {
    compileSdk 32

    defaultConfig {
        applicationId "com.example.autosos"
        minSdk 23
        targetSdk 32
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles    getDefaultProguardFile('proguard-android-optimize.txt'),
'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
    buildFeatures {
        viewBinding true
    }
}

dependencies {

    implementation 'com.squareup.picasso:picasso:2.71828'

    implementation 'de.hdodenhof:circleimageview:3.1.0'
    implementation 'androidx.cardview:cardview:1.0.0'

    implementation 'androidx.appcompat:appcompat:1.4.2'
    implementation 'com.google.android.material:material:1.6.1'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.4'
    implementation 'com.google.android.gms:play-services-maps:18.0.2'
    implementation 'com.google.android.gms:play-services-location:19.0.1'

    implementation 'com.firebaseui:firebase-ui-auth:7.2.0'
    implementation 'com.google.firebase:firebase-auth:21.0.5'
```

```
    implementation 'com.google.firebase:firebase-database:20.0.5'
    implementation 'com.google.firebase:firebase-storage:20.0.1'
    implementation platform('com.google.firebase:firebase-bom:30.1.0')
    implementation 'com.firebase:geofire-android:3.2.0'

    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.3'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'
    implementation platform('com.google.firebase:firebase-bom:30.1.0')
}
```

Додаток В

У даному додатку наведено зміст файлів типу Java, в яких відтворено увесь функціонал додатку відповідно. Перелік сформовано в алфавітному порядку.

Клас «AboutActivity»

```
package com.example.autosos;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class AboutActivity extends AppCompatActivity {

    Button aboutBackMenuButton;

    private String getType;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_about);

        getType = getIntent().getStringExtra("type");

        aboutBackMenuButton = (Button) findViewById(R.id.aboutBackMenuButton);

        aboutBackMenuButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent backMenuIntent;
                if (getType.equals("Customer")) {
                    backMenuIntent = new Intent(AboutActivity.this,
CustomerMenuActivity.class);
                    startActivity(backMenuIntent);
                }
                else if (getType.equals("Driver")) {
                    backMenuIntent = new Intent(AboutActivity.this,
DriverMenuActivity.class);
                    startActivity(backMenuIntent);
                }
            }
        });
   }
}
```

Клас «CustomerInfoRegistrationActivity»

```
package com.example.autosos;
```

```
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import java.util.HashMap;

public class CustomerInfoRegistrationActivity extends AppCompatActivity {

    EditText customerInfoName, customerInfoPhone, customerInfoCarName,
customerInfoCarNumber;
    Button customerCreateButton;

    private FirebaseAuth mAuth;
    private DatabaseReference databaseReference;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_customer_info_registration);

        customerInfoName = (EditText) findViewById(R.id.customerInfoName);
        customerInfoPhone = (EditText) findViewById(R.id.customerInfoPhone);
        customerInfoCarName = (EditText) findViewById(R.id.customerInfoCarName);
        customerInfoCarNumber = (EditText) findViewById(R.id.customerInfoCarNumber);
        customerCreateButton = (Button) findViewById(R.id.customerCreateButton);

        mAuth = FirebaseAuth.getInstance();
        databaseReference =
FirebaseDatabase.getInstance().getReference().child("Users").child("Customers");

        customerCreateButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                validateAndSaveInfo();
            }
        });
    }

    private void validateAndSaveInfo() {
        if (TextUtils.isEmpty(customerInfoName.getText().toString())) {
            Toast.makeText(this, "Заповніть поле з Вашим іменем",
Toast.LENGTH_SHORT).show();
        }
        else if (TextUtils.isEmpty(customerInfoPhone.getText().toString())) {
            Toast.makeText(this, "Заповніть поле з номером телефону",
Toast.LENGTH_SHORT).show();
        }
        else if (TextUtils.isEmpty(customerInfoCarName.getText().toString())) {
            Toast.makeText(this, "Заповніть поле інформації про Ваше авто",
Toast.LENGTH_SHORT).show();
        }
        else if (TextUtils.isEmpty(customerInfoCarNumber.getText().toString())) {
```

```
            Toast.makeText(this, "Заповніть поле інформації про Ваш номер авто",
Toast.LENGTH_SHORT).show();
        }
        else {
            HashMap<String, Object> userMap = new HashMap<>();
            userMap.put("UserID", mAuth.getCurrentUser().getUid());
            userMap.put("Name", customerInfoName.getText().toString());
            userMap.put("Phone", customerInfoPhone.getText().toString());
            userMap.put("CarName", customerInfoCarName.getText().toString());
            userMap.put("CarNumber", customerInfoCarNumber.getText().toString());
            userMap.put("Online Status", true);


databaseReference.child(mAuth.getCurrentUser().getUid()).updateChildren(userMap);
            startActivity(new Intent(CustomerInfoRegistrationActivity.this,
CustomerMapsActivity.class));
        }
    }
}
```

Клас «CustomerMapsActivity»

```
package com.example.autosos;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;
import androidx.fragment.app.FragmentActivity;

import android.Manifest;
import android.annotation.SuppressLint;
import android.content.Context;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.Location;
import android.net.Uri;
import android.os.Bundle;
import android.view.Gravity;
import android.view.View;
import android.widget.Button;
import android.widget.RelativeLayout;
import android.widget.TextView;
import android.widget.Toast;

import com.firebase.geofire.GeoFire;
import com.firebase.geofire.GeoLocation;
import com.firebase.geofire.GeoQuery;
import com.firebase.geofire.GeoQueryEventListener;
import com.google.android.gms.common.ConnectionResult;
import com.google.android.gms.common.api.GoogleApiClient;
import com.google.android.gms.location.LocationRequest;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
```

```
import com.google.android.gms.maps.model.Marker;
import com.google.android.gms.maps.model.MarkerOptions;
import com.example.autosos.databinding.ActivityCustomerMapsBinding;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

import java.util.HashMap;
import java.util.List;

public class CustomerMapsActivity extends FragmentActivity implements
OnMapReadyCallback,
        GoogleApiClient.ConnectionCallbacks,
        GoogleApiClient.OnConnectionFailedListener,
        com.google.android.gms.location.LocationListener {

    private GoogleMap mMap;
    private ActivityCustomerMapsBinding binding;

    TextView locationEditText;
    TextView driverMapsName, driverMapsPhoneNumber, driverMapsCarInfo,
driverMapsCarNumber;
    RelativeLayout driverInfoRelativeLayout, customerUpdateDriverPosition;
    Button customerMenuButton, callDriverButton, calCurrentDriverButton,
customerCancelOrderButton;

    GoogleApiClient googleApiClient;
    Location lastLocation;
    LocationRequest locationRequest;
    Marker driverMarker, pickUpMarker;

    private String customerID;
    private LatLng customerPosition;
    private int radius = 1;
    private Boolean driverFound = false, requestType = false;

    private String driverFoundID;
    private DatabaseReference customerDatabaseRef;
    private DatabaseReference driversAvailableRef;

    private DatabaseReference driversLocationRef;
    private DatabaseReference driversRef;

    private FirebaseAuth mAuth;
    private FirebaseUser currentUser;
    private Boolean currentLogOutDriverStatus;

    private ValueEventListener driverLocationRefListener;
    GeoQuery geoQuery;

    private Boolean currentUserOrderingStatus = false;

    public static String menuCustomerLatitude, menuCustomerLongitude;
    public static Location menuCurrentCustomerCoordinates;
    public static String currentDriverPhoneNumber;
```

```
    private String cancelOrderDriverID;
    private LatLng DriverLatLng;

    @SuppressLint("DefaultLocale")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        binding = ActivityCustomerMapsBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        locationEditText = (TextView) findViewById(R.id.locationEditText);
        driverMapsName = (TextView) findViewById(R.id.driverMapsName);
        driverMapsPhoneNumber = (TextView) findViewById(R.id.driverMapsPhoneNumber);
        driverMapsCarInfo = (TextView) findViewById(R.id.driverMapsCarInfo);
        driverMapsCarNumber = (TextView) findViewById(R.id.driverMapsCarNumber);
        driverInfoRelativeLayout = (RelativeLayout)
findViewById(R.id.driverInfoRelativeLayout);
        driverInfoRelativeLayout.setVisibility(View.INVISIBLE);
        calCurrentDriverButton = (Button) findViewById(R.id.calCurrentDriverButton);
        calCurrentDriverButton.setVisibility(View.INVISIBLE);
        callDriverButton = (Button) findViewById(R.id.callDriverButton);
        customerMenuButton = (Button) findViewById(R.id.customerMenuButton);
        customerCancelOrderButton = (Button)
findViewById(R.id.customerCancelOrderButton);
        customerCancelOrderButton.setVisibility(View.INVISIBLE);

        mAuth = FirebaseAuth.getInstance();
        currentUser = mAuth.getCurrentUser();

        customerID = FirebaseAuth.getInstance().getCurrentUser().getUid();
        customerDatabaseRef =
FirebaseDatabase.getInstance().getReference().child("Customers Requests");
        driversAvailableRef =
FirebaseDatabase.getInstance().getReference().child("Available Drivers");
        driversLocationRef =
FirebaseDatabase.getInstance().getReference().child("Driver Working");

        customerUpdateDriverPosition = (RelativeLayout)
findViewById(R.id.customerUpdateDriverPosition);
        customerUpdateDriverPosition.setVisibility(View.INVISIBLE);

        SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager()
                .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);

        callDriverButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (requestType) {
                    requestType = false;
                    GeoFire geofire = new GeoFire(customerDatabaseRef);
                    geofire.removeLocation(customerID);
                    if (pickUpMarker != null) {
                        pickUpMarker.remove();
                    }
                    if (driverMarker != null) {
                        driverMarker.remove();
```

```
                    }

                    callDriverButton.setText("Викликати евакуатор");
                    if (driverFound != null) {
                        driversRef = FirebaseDatabase.getInstance().getReference()

.child("Users").child("Drivers").child(driverFoundID).child("CustomerRideID");
                        driversRef.removeValue();
                        driverFoundID = null;
                    }

                    driverFound = false;
                    radius = 1;
                } else {
                    requestType = true;

                    GeoFire geofire = new GeoFire(customerDatabaseRef);
                    geofire.setLocation(customerID, new
GeoLocation(lastLocation.getLatitude(), lastLocation.getLongitude()));

                    customerPosition = new LatLng(lastLocation.getLatitude(),
lastLocation.getLongitude());
                    pickUpMarker = mMap.addMarker(new
MarkerOptions().position(customerPosition).title("Я тут")

.icon(BitmapDescriptorFactory.fromResource(R.drawable.location_image)));

                    callDriverButton.setText("Пошук...");
                    getNearbyDrivers();
                }
            }
        });

        CheckCurrentUserOrdering();
        customerMenuButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (currentUserOrderingStatus) {
                    Toast toast = Toast.makeText(CustomerMapsActivity.this,
"Завершіть замовлення або відмініть замовлення",
                            Toast.LENGTH_SHORT);
                    toast.setGravity(Gravity.CENTER, 0, 0);
                    toast.show();
                }
                else {
                    Intent intent = new Intent(CustomerMapsActivity.this,
CustomerMenuActivity.class);
                    startActivity(intent);
                }
            }
        });

        customerCancelOrderButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                driverInfoRelativeLayout.setVisibility(View.INVISIBLE);
                calCurrentDriverButton.setVisibility(View.INVISIBLE);
                customerCancelOrderButton.setVisibility(View.INVISIBLE);
                customerUpdateDriverPosition.setVisibility(View.INVISIBLE);
                callDriverButton.setText("Викликати евакуатор");
```

```
                locationEditText.setText("Приємного користування");
                CancelCurrentOrder();
                driverMarker.remove();
                pickUpMarker.remove();
            }
        });

        customerUpdateDriverPosition.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                getNearbyDrivers();
            }
        });

        calCurrentDriverButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                int permissionCheck =
ContextCompat.checkSelfPermission(CustomerMapsActivity.this,
                        Manifest.permission.CALL_PHONE);

                if (permissionCheck != PackageManager.PERMISSION_GRANTED) {
                    ActivityCompat.requestPermissions(
                            CustomerMapsActivity.this, new
String[]{Manifest.permission.CALL_PHONE},
                            123);
                }
                else {
                    Intent intent = new Intent(Intent.ACTION_CALL, Uri.parse("tel: "
+ currentDriverPhoneNumber));
                    startActivity(intent);
                }
            }
        });
    }

    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;
        buildGoogleApiClient();
        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED &&
ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_COARSE_LOCATION)
!= PackageManager.PERMISSION_GRANTED) {
            return;
        }
        mMap.setMyLocationEnabled(true);
    }

    @Override
    public void onConnected(@Nullable Bundle bundle) {
        locationRequest = new LocationRequest();
        locationRequest.setInterval(1000);
        locationRequest.setFastestInterval(1000);
        locationRequest.setPriority(locationRequest.PRIORITY_HIGH_ACCURACY);

        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION)
```

```
                != PackageManager.PERMISSION_GRANTED &&
ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_COARSE_LOCATION)
!= PackageManager.PERMISSION_GRANTED) {
            return;
        }
        LocationServices.FusedLocationApi.requestLocationUpdates(googleApiClient,
locationRequest, this);
    }

    @Override
    public void onConnectionSuspended(int i) {}

    @Override
    public void onConnectionFailed(@NonNull ConnectionResult connectionResult) {}

    @SuppressLint("DefaultLocale")
    @Override
    public void onLocationChanged(@NonNull Location location) {
        lastLocation = location;

        LatLng latLng = new LatLng(location.getLatitude(), location.getLongitude());
        mMap.moveCamera(CameraUpdateFactory.newLatLng(latLng));
        mMap.animateCamera(CameraUpdateFactory.zoomTo(12));

        menuCustomerLatitude = String.format("%.3f", lastLocation.getLatitude());
        menuCustomerLongitude = String.format("%.3f", lastLocation.getLongitude());
    }

    protected synchronized void buildGoogleApiClient() {
        googleApiClient = new GoogleApiClient.Builder(this)
                .addConnectionCallbacks(this)
                .addOnConnectionFailedListener(this)
                .addApi(LocationServices.API)
                .build();
        googleApiClient.connect();
    }

    @Override
    protected void onStop() {
        super.onStop();
    }

    private void getNearbyDrivers() {
        GeoFire geoFire = new GeoFire(driversAvailableRef);
        GeoQuery geoQuery = geoFire.queryAtLocation(new
GeoLocation(customerPosition.latitude, customerPosition.longitude), radius);
        geoQuery.removeAllListeners();

        geoQuery.addGeoQueryEventListener(new GeoQueryEventListener() {
            @Override
            public void onKeyEntered(String key, GeoLocation location) {
                if (!driverFound && requestType) {
                    driverFound = true;
                    final String driverFoundID = key;

                    cancelOrderDriverID = key;

                    driversRef =
FirebaseDatabase.getInstance().getReference().child("Users").child("Drivers").child(d
riverFoundID);
```

```
                    HashMap driverMap = new HashMap();
                    driverMap.put("CustomerRideID", customerID);
                    driversRef.updateChildren(driverMap);

                    driverLocationRefListener =
driversLocationRef.child(driverFoundID).child("l").
                            addValueEventListener(new ValueEventListener() {
                                @SuppressLint({"SetTextI18n", "DefaultLocale"})
                                @Override
                                public void onDataChange(@NonNull DataSnapshot
dataSnapshot) {
                                    if (dataSnapshot.exists() && requestType) {
                                        List<Object> driverLocationMap =
(List<Object>) dataSnapshot.getValue();
                                        double locationLat = 0;
                                        double locationLng = 0;

                                        callDriverButton.setText("Знайдено");


driverInfoRelativeLayout.setVisibility(View.VISIBLE);

calCurrentDriverButton.setVisibility(View.VISIBLE);

customerCancelOrderButton.setVisibility(View.VISIBLE);

customerUpdateDriverPosition.setVisibility(View.VISIBLE);
                                        DatabaseReference reference =
FirebaseDatabase.getInstance().getReference()

.child("Users").child("Drivers").child(driverFoundID);

                                        reference.addValueEventListener(new
ValueEventListener() {
                                            @Override
                                            public void onDataChange(@NonNull
DataSnapshot dataSnapshot) {
                                                if (dataSnapshot.exists() &&
dataSnapshot.getChildrenCount() > 0) {
                                                    String name =
dataSnapshot.child("Name").getValue().toString();
                                                    String phone =
dataSnapshot.child("Phone").getValue().toString();
                                                    String carName =
dataSnapshot.child("CarName").getValue().toString();
                                                    String carNumber =
dataSnapshot.child("CarNumber").getValue().toString();

                                                    driverMapsName.setText(name);

driverMapsPhoneNumber.setText(phone);

driverMapsCarInfo.setText(carName);

driverMapsCarNumber.setText(carNumber);

                                                    currentDriverPhoneNumber = phone;
                                                }
                                            }
```

```
                                        @Override
                                        public void onCancelled(@NonNull
DatabaseError databaseError) {}
                                    });

                                    if (driverLocationMap.get(0) != null) {
                                        locationLat =
Double.parseDouble(driverLocationMap.get(0).toString());
                                    }
                                    if (driverLocationMap.get(1) != null) {
                                        locationLng =
Double.parseDouble(driverLocationMap.get(1).toString());
                                    }
                                    DriverLatLng = new LatLng(locationLat,
locationLng);

                                    if (driverMarker != null) {
                                        driverMarker.remove();
                                    }

                                    Location location1 = new Location("");

location1.setLatitude(customerPosition.latitude);

location1.setLongitude(customerPosition.longitude);

                                    Location location2 = new Location("");
                                    location2.setLatitude(DriverLatLng.latitude);

location2.setLongitude(DriverLatLng.longitude);

                                    float Distance =
location1.distanceTo(location2);
                                    if (Distance < 100 && Distance != 0) {
                                        callDriverButton.setText("Майже на
місці");
                                    } else {
                                        float roadDistance = Distance / 1000;
                                        locationEditText.setText("Відстань до
авто " + String.format("%.2f", roadDistance) + " км");
                                    }

                                    driverMarker = mMap.addMarker(new
MarkerOptions().position(DriverLatLng)
                                            .title("Ваше такси
тут").icon(BitmapDescriptorFactory.fromResource(R.drawable.tow_truck_image)));

                                    if (Distance == 0) {
                                        Intent resultIntent = new
Intent(CustomerMapsActivity.this, CustomerOrderResultActivity.class);
                                        startActivity(resultIntent);
                                        CancelCurrentOrder();
                                    }
                                }
                            }

                            @Override
                            public void onCancelled(@NonNull DatabaseError
databaseError) {}
                        });
```

```
                }
            }

            @Override
            public void onKeyExited(String key) {}

            @Override
            public void onKeyMoved(String key, GeoLocation location) {}

            @Override
            public void onGeoQueryReady() {
                if (!driverFound)
                {
                    radius = radius + 1;
                    getNearbyDrivers();
                }
            }

            @Override
            public void onGeoQueryError(DatabaseError error) {}
        });
    }

    private void CheckCurrentUserOrdering() {
        String customerID = FirebaseAuth.getInstance().getCurrentUser().getUid();
        DatabaseReference checkRef =
FirebaseDatabase.getInstance().getReference().child("Customers Requests")
                .child(customerID);

        checkRef.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot snapshot) {
                if (snapshot.exists()) {
                    currentUserOrderingStatus = true;
                }
            }

            @Override
            public void onCancelled(@NonNull DatabaseError error) {}
        });
    }

    private void CancelCurrentOrder() {

        DatabaseReference workingDriversRef =
FirebaseDatabase.getInstance().getReference().child("Driver Working");
        GeoFire geoFireWorking = new GeoFire(workingDriversRef);
        geoFireWorking.removeLocation(cancelOrderDriverID);

        String customerID = FirebaseAuth.getInstance().getCurrentUser().getUid();
        DatabaseReference customerRequestRef =
FirebaseDatabase.getInstance().getReference().child("Customers Requests");
        GeoFire geoFireRequest = new GeoFire(customerRequestRef);
        geoFireRequest.removeLocation(customerID);

        Task<Void> deleteCustomerID =
FirebaseDatabase.getInstance().getReference().child("Users").child("Drivers")
                .child(cancelOrderDriverID).child("CustomerRideID").setValue(null);
```

```
        DatabaseReference availableDriversRef =
FirebaseDatabase.getInstance().getReference().child("Available Drivers");
        GeoFire geoFireAvailability = new GeoFire(availableDriversRef);
        geoFireAvailability.setLocation(cancelOrderDriverID, new
GeoLocation(DriverLatLng.latitude, DriverLatLng.longitude));

        driverFound = false;
        requestType = false;
        currentUserOrderingStatus = false;
    }
}
```

Клас «CustomerMenuActivity»

```
package com.example.autosos;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.RelativeLayout;
import android.widget.TextView;
import android.widget.Toast;

import com.firebase.geofire.GeoFire;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

import java.util.HashMap;

public class CustomerMenuActivity extends AppCompatActivity {

    EditText menuCustomerName, menuCustomerPhone, menuCustomerCarInfo,
menuCustomerCarNumber;
    ImageView closeCustomerMenuButton, saveCustomerMenuButton;
    TextView latitudeCustomerInfo, longitudeCustomerInfo;
    RelativeLayout exitCustomerButton, deleteCustomerButton, siteCustomerButton,
aboutCustomerButton;

    private FirebaseAuth mAuth;
    private DatabaseReference databaseReference, driverAvailableRef;

    private Boolean currentLogOutCustomerStatus = false;
    private Boolean currentDriverWorkingStatus = false;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

```
        setContentView(R.layout.activity_customer_menu);

        menuCustomerName = (EditText) findViewById(R.id.menuCustomerName);
        menuCustomerPhone = (EditText) findViewById(R.id.menuCustomerPhone);
        menuCustomerCarInfo = (EditText) findViewById(R.id.menuCustomerCarInfo);
        menuCustomerCarNumber = (EditText) findViewById(R.id.menuCustomerCarNumber);

        closeCustomerMenuButton = (ImageView)
findViewById(R.id.closeCustomerMenuButton);
        saveCustomerMenuButton = (ImageView)
findViewById(R.id.saveCustomerMenuButton);

        latitudeCustomerInfo = (TextView) findViewById(R.id.latitudeCustomerInfo);
        latitudeCustomerInfo.setText(CustomerMapsActivity.menuCustomerLatitude);
        longitudeCustomerInfo = (TextView) findViewById(R.id.longitudeCustomerInfo);
        longitudeCustomerInfo.setText(CustomerMapsActivity.menuCustomerLongitude);

        exitCustomerButton = (RelativeLayout) findViewById(R.id.exitCustomerButton);
        deleteCustomerButton = (RelativeLayout)
findViewById(R.id.deleteCustomerButton);
        siteCustomerButton = (RelativeLayout) findViewById(R.id.siteCustomerButton);
        aboutCustomerButton = (RelativeLayout)
findViewById(R.id.aboutCustomerButton);

        mAuth = FirebaseAuth.getInstance();

        databaseReference =
FirebaseDatabase.getInstance().getReference().child("Users").child("Customers");
        getUserInformation();
        CheckCurrentUserOrdering();

        closeCustomerMenuButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent backIntent = new Intent(CustomerMenuActivity.this,
CustomerMapsActivity.class);
                startActivity(backIntent);
            }
        });

        saveCustomerMenuButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                validateAndSaveInfo();
            }
        });

        exitCustomerButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (currentDriverWorkingStatus) {
                    Toast.makeText(CustomerMenuActivity.this, "Завершіть Ваше
замовлення", Toast.LENGTH_SHORT).show();
                }
                else {
                    currentLogOutCustomerStatus = true;
                    LogoutCustomer();
                    DisconnectCustomer();
                    mAuth.signOut();
                }
```

```
            }
        });

        deleteCustomerButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (currentDriverWorkingStatus) {
                    Toast.makeText(CustomerMenuActivity.this, "Завершіть Ваше
замовлення", Toast.LENGTH_SHORT).show();
                }
                else {
                    removeCurrentCustomer();
                    LogoutCustomer();
                }
            }
        });

        siteCustomerButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent siteIntent = new Intent(CustomerMenuActivity.this,
SiteActivity.class);
                siteIntent.putExtra("type", "Customer");
                startActivity(siteIntent);
            }
        });

        aboutCustomerButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent aboutIntent = new Intent(CustomerMenuActivity.this,
AboutActivity.class);
                aboutIntent.putExtra("type", "Customer");
                startActivity(aboutIntent);
            }
        });

    }

    private void validateAndSaveInfo() {
        if (TextUtils.isEmpty(menuCustomerName.getText().toString())) {
            Toast.makeText(this, "Заповніть поле Name", Toast.LENGTH_SHORT).show();
        }
        else if (TextUtils.isEmpty(menuCustomerPhone.getText().toString())) {
            Toast.makeText(this, "Заповніть поле Phone", Toast.LENGTH_SHORT).show();
        }
        else if (TextUtils.isEmpty(menuCustomerCarInfo.getText().toString())) {
            Toast.makeText(this, "Заповніть поле Auto", Toast.LENGTH_SHORT).show();
        }
        else if (TextUtils.isEmpty(menuCustomerCarNumber.getText().toString())) {
            Toast.makeText(this, "Заповніть поле Auto", Toast.LENGTH_SHORT).show();
        }
        else {
            HashMap<String, Object> userMap = new HashMap<>();
            userMap.put("UserID", mAuth.getCurrentUser().getUid());
            userMap.put("Name", menuCustomerName.getText().toString());
            userMap.put("Phone", menuCustomerPhone.getText().toString());
            userMap.put("CarName", menuCustomerCarInfo.getText().toString());
            userMap.put("CarNumber", menuCustomerCarNumber.getText().toString());
```

```
databaseReference.child(mAuth.getCurrentUser().getUid()).updateChildren(userMap);
            startActivity(new Intent(CustomerMenuActivity.this,
CustomerMapsActivity.class));
        }
    }

    private void getUserInformation() {


databaseReference.child(mAuth.getCurrentUser().getUid()).addValueEventListener(new
ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot snapshot) {
                if (snapshot.exists() && snapshot.getChildrenCount() > 0) {
                    String name = snapshot.child("Name").getValue().toString();
                    String phone = snapshot.child("Phone").getValue().toString();
                    String carInfo = snapshot.child("CarName").getValue().toString();
                    String carNumber =
snapshot.child("CarNumber").getValue().toString();

                    menuCustomerName.setText(name);
                    menuCustomerPhone.setText(phone);
                    menuCustomerCarInfo.setText(carInfo);
                    menuCustomerCarNumber.setText(carNumber);
                }
            }

            @Override
            public void onCancelled(@NonNull DatabaseError error) {}
        });
    }

    private void DisconnectCustomer() {
        String customerID = FirebaseAuth.getInstance().getCurrentUser().getUid();

        DatabaseReference customerDatabaseRef =
FirebaseDatabase.getInstance().getReference()
                .child("Users").child("Customers").child(customerID).child("Online
Status");
        customerDatabaseRef.setValue(false);

        DatabaseReference customerRequestRef =
FirebaseDatabase.getInstance().getReference().child("Customers Requests");
        GeoFire geoFire = new GeoFire(customerRequestRef);
        geoFire.removeLocation(customerID);
    }

    private void LogoutCustomer() {
        Intent roleSelectionIntent = new Intent(CustomerMenuActivity.this,
RoleSelectionActivity.class);
        startActivity(roleSelectionIntent);
        finish();
    }

    private void removeCurrentCustomer() {

        String customerID = FirebaseAuth.getInstance().getCurrentUser().getUid();
        Task<Void> removeReference =
FirebaseDatabase.getInstance().getReference().child("Users")
```

```
                .child("Customers").child(customerID).removeValue();

        FirebaseUser currentUser = mAuth.getCurrentUser();
        currentUser.delete();
    }

    private void CheckCurrentUserOrdering() {
        String customerID = FirebaseAuth.getInstance().getCurrentUser().getUid();
        DatabaseReference checkRef =
FirebaseDatabase.getInstance().getReference().child("Customers Requests")
                .child(customerID);

        checkRef.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot snapshot) {
                if (snapshot.exists()) {
                    currentDriverWorkingStatus = true;
                }
            }

            @Override
            public void onCancelled(@NonNull DatabaseError error) {}
        });
    }
}
```

Клас «CustomerOrderResultActivity»

```
package com.example.autosos;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class CustomerOrderResultActivity extends AppCompatActivity {

    Button customerResultButton;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_customer_order_result);

        customerResultButton = (Button) findViewById(R.id.customerResultButton);

        customerResultButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent backMapIntent = new Intent(CustomerOrderResultActivity.this,
CustomerMapsActivity.class);
                startActivity(backMapIntent);
            }
        });
    }
}
```

Клас «CustomerRegistrationActivity»

```
package com.example.autosos;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

public class CustomerRegistrationActivity extends AppCompatActivity {

    EditText registrationEmailCustomerEditText, registrationPasswordCustomerEditText;
    Button registrationCustomerButton;

    FirebaseAuth firebaseAuth;
    DatabaseReference customerDatabaseRef;
    String onlineCustomerID;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_customer_registration);

        registrationEmailCustomerEditText = (EditText)
findViewById(R.id.registrationEmailCustomerEditText);
        registrationPasswordCustomerEditText = (EditText)
findViewById(R.id.registrationPasswordCustomerEditText);
        registrationCustomerButton = (Button)
findViewById(R.id.registrationCustomerButton);

        firebaseAuth = FirebaseAuth.getInstance();

        registrationCustomerButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if
(TextUtils.isEmpty(registrationEmailCustomerEditText.getText().toString())) {
                    Toast.makeText(CustomerRegistrationActivity.this, "Заповніть поле
email-адреси",
                            Toast.LENGTH_SHORT).show();
                }
                else if
(TextUtils.isEmpty(registrationPasswordCustomerEditText.getText().toString())) {
                    Toast.makeText(CustomerRegistrationActivity.this, "Заповніть поле
з паролем",
                            Toast.LENGTH_SHORT).show();
```

```
                }
                else {
                    String email =
registrationEmailCustomerEditText.getText().toString();
                    String password =
registrationPasswordCustomerEditText.getText().toString();
                    RegistrationCustomerMethod(email, password);
                }
            }
        });
    }

    private void RegistrationCustomerMethod(String email, String password) {
        firebaseAuth.createUserWithEmailAndPassword(email,
password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful()) {

                    onlineCustomerID = firebaseAuth.getCurrentUser().getUid();
                    customerDatabaseRef =
FirebaseDatabase.getInstance().getReference()

.child("Users").child("Customers").child(onlineCustomerID);

                    Intent mainScreenIntent = new
Intent(CustomerRegistrationActivity.this, CustomerInfoRegistrationActivity.class);
                    startActivity(mainScreenIntent);

                    Toast.makeText(CustomerRegistrationActivity.this, "Реєстрація
успішна",
                            Toast.LENGTH_SHORT).show();
                }
                else {
                    Toast.makeText(CustomerRegistrationActivity.this, "Помилка
реєстрації нового користувача",
                            Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
}
```

Клас «CustomerWelcomeActivity»

```
package com.example.autosos;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
```

```
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

public class CustomerWelcomeActivity extends AppCompatActivity {

    EditText editTextCustomerEmail, editTextCustomerPassword;
    Button signInCustomerButton, roleBackCustomerButton, signUpCustomerButton;

    FirebaseAuth firebaseAuth;
    DatabaseReference customerDatabaseRef;
    String onlineCustomerID;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_customer_welcome);

        editTextCustomerEmail = (EditText) findViewById(R.id.editTextCustomerEmail);
        editTextCustomerPassword = (EditText)
findViewById(R.id.editTextCustomerPassword);
        signInCustomerButton = (Button) findViewById(R.id.signInCustomerButton);
        signUpCustomerButton = (Button) findViewById(R.id.signUpCustomerButton);
        roleBackCustomerButton = (Button) findViewById(R.id.roleBackCustomerButton);

        firebaseAuth = FirebaseAuth.getInstance();

        signInCustomerButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String email = editTextCustomerEmail.getText().toString();
                String password = editTextCustomerPassword.getText().toString();
                SignInCustomerMethod(email, password);
            }
        });

        signUpCustomerButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent registrationIntent = new Intent(CustomerWelcomeActivity.this,
CustomerRegistrationActivity.class);
                startActivity(registrationIntent);
            }
        });

        roleBackCustomerButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent roleSelectionIntent = new Intent(CustomerWelcomeActivity.this,
RoleSelectionActivity.class);
                startActivity(roleSelectionIntent);
            }
        });
    }

    private void SignInCustomerMethod(String email, String password) {
        firebaseAuth.signInWithEmailAndPassword(email,
password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
            @Override
```

```
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful()) {
                    Toast.makeText(CustomerWelcomeActivity.this, "Вхід успішний",
Toast.LENGTH_SHORT).show();

                    onlineCustomerID = firebaseAuth.getCurrentUser().getUid();
                    customerDatabaseRef =
FirebaseDatabase.getInstance().getReference()

.child("Users").child("Customers").child(onlineCustomerID).child("Online Status");
                    customerDatabaseRef.setValue(true);

                    Intent mainScreenIntent = new
Intent(CustomerWelcomeActivity.this, CustomerMapsActivity.class);
                    startActivity(mainScreenIntent);
                }
                else {
                    Toast.makeText(CustomerWelcomeActivity.this, "Помилка входу",
                            Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
}
```

Клас «DriverInfoRegistrationActivity»

```
package com.example.autosos;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

import java.util.HashMap;

public class DriverInfoRegistrationActivity extends AppCompatActivity {

    EditText driverInfoName, driverInfoPhone, driverInfoCarName, driverInfoCarNumber;
    Button driverCreateButton;

    private FirebaseAuth mAuth;
    private DatabaseReference databaseReference;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_driver_info_registration);

        driverInfoName = (EditText) findViewById(R.id.driverInfoName);
```

```
        driverInfoPhone = (EditText) findViewById(R.id.driverInfoPhone);
        driverInfoCarName = (EditText) findViewById(R.id.driverInfoCarName);
        driverInfoCarNumber = (EditText) findViewById(R.id.driverInfoCarNumber);
        driverCreateButton = (Button) findViewById(R.id.driverCreateButton);

        mAuth = FirebaseAuth.getInstance();
        databaseReference =
FirebaseDatabase.getInstance().getReference().child("Users").child("Drivers");

        driverCreateButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                validateAndSaveInfo();
            }
        });
    }

    private void validateAndSaveInfo() {
        if (TextUtils.isEmpty(driverInfoName.getText().toString())) {
            Toast.makeText(this, "Заповніть поле з Вашим іменем",
Toast.LENGTH_SHORT).show();
        }
        else if (TextUtils.isEmpty(driverInfoPhone.getText().toString())) {
            Toast.makeText(this, "Заповніть поле з номером телефону",
Toast.LENGTH_SHORT).show();
        }
        else if (TextUtils.isEmpty(driverInfoCarName.getText().toString())) {
            Toast.makeText(this, "Заповніть поле інформації про Ваш евакуатор",
Toast.LENGTH_SHORT).show();
        }
        else if (TextUtils.isEmpty(driverInfoCarNumber.getText().toString())) {
            Toast.makeText(this, "Заповніть поле інформації про Ваш номер авто",
Toast.LENGTH_SHORT).show();
        }
        else {
            HashMap<String, Object> userMap = new HashMap<>();
            userMap.put("DriverID", mAuth.getCurrentUser().getUid());
            userMap.put("Name", driverInfoName.getText().toString());
            userMap.put("Phone", driverInfoPhone.getText().toString());
            userMap.put("CarName", driverInfoCarName.getText().toString());
            userMap.put("CarNumber", driverInfoCarNumber.getText().toString());
            userMap.put("Online Status", true);


databaseReference.child(mAuth.getCurrentUser().getUid()).updateChildren(userMap);
            startActivity(new Intent(DriverInfoRegistrationActivity.this,
DriverMapsActivity.class));
        }
    }
}
```

Клас «DriverMapsActivity»

```
package com.example.autosos;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;
```

```
import androidx.fragment.app.FragmentActivity;

import android.Manifest;
import android.annotation.SuppressLint;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.Location;
import android.location.LocationListener;
import android.net.Uri;
import android.os.Bundle;
import android.view.Gravity;
import android.view.View;
import android.widget.Button;
import android.widget.RelativeLayout;
import android.widget.TextView;
import android.widget.Toast;

import com.firebase.geofire.GeoFire;
import com.firebase.geofire.GeoLocation;
import com.google.android.gms.common.ConnectionResult;
import com.google.android.gms.common.api.GoogleApiClient;
import com.google.android.gms.location.LocationRequest;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.Marker;
import com.google.android.gms.maps.model.MarkerOptions;
import com.example.autosos.databinding.ActivityDriverMapsBinding;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

import java.util.List;

public class DriverMapsActivity extends FragmentActivity implements
OnMapReadyCallback,
        GoogleApiClient.ConnectionCallbacks,
        GoogleApiClient.OnConnectionFailedListener,
        com.google.android.gms.location.LocationListener {
    private GoogleMap mMap;
    private ActivityDriverMapsBinding binding;

    GoogleApiClient googleApiClient;
    Location lastLocation;
    LocationRequest locationRequest;
    Marker driverPickUpMarker;

    private FirebaseAuth mAuth;
    private FirebaseUser currentUser;
    private Boolean currentLogOutDriverStatus = false;
    private DatabaseReference assignedCustomerRef, assignedCustomerPositionRef;
```

```
    private String driverID, customerID = "";

    private ValueEventListener assignedCustomerPositionListener;

    public static String menuDriverLatitude, menuDriverLongitude;
    public static Location menuCurrentDriverCoordinates;

    TextView locationDriverTextView, customerMapsName, customerMapsPhoneNumber,
customerMapsCarInfo, customerMapsCarNumber;
    Button driverMenuButton, calCurrentCustomerButton;
    RelativeLayout customerInfoRelativeLayout;

    private Boolean currentUserOrderingStatus = false;

    public static String currentCustomerPhoneNumber;
    public static float roadDriverCustomerDistance;

    private LatLng driverCurrentPosition;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        binding = ActivityDriverMapsBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        mAuth = FirebaseAuth.getInstance();
        currentUser = mAuth.getCurrentUser();
        driverID = mAuth.getCurrentUser().getUid();

        locationDriverTextView = (TextView)
findViewById(R.id.locationDriverTextView);
        customerMapsName = (TextView) findViewById(R.id.customerMapsName);
        customerMapsPhoneNumber = (TextView)
findViewById(R.id.customerMapsPhoneNumber);
        customerMapsCarInfo = (TextView) findViewById(R.id.customerMapsCarInfo);
        customerMapsCarNumber = (TextView) findViewById(R.id.customerMapsCarNumber);

        calCurrentCustomerButton = (Button)
findViewById(R.id.calCurrentCustomerButton);
        calCurrentCustomerButton.setVisibility(View.INVISIBLE);
        driverMenuButton = (Button) findViewById(R.id.driverMenuButton);

        customerInfoRelativeLayout = (RelativeLayout)
findViewById(R.id.customerInfoRelativeLayout);
        customerInfoRelativeLayout.setVisibility(View.INVISIBLE);

        SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager()
                .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);

        CheckCurrentUserOrdering();
        driverMenuButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (currentUserOrderingStatus) {
                    Toast toast = Toast.makeText(DriverMapsActivity.this, "Завершіть
Ваше замовлення",
                            Toast.LENGTH_SHORT);
```

```
                    toast.setGravity(Gravity.CENTER, 0, 0);
                    toast.show();
                }
                else {
                    Intent intent = new Intent(DriverMapsActivity.this,
DriverMenuActivity.class);
                    startActivity(intent);
                }
            }
        });

        calCurrentCustomerButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                int permissionCheck =
ContextCompat.checkSelfPermission(DriverMapsActivity.this,
                        Manifest.permission.CALL_PHONE);

                if (permissionCheck != PackageManager.PERMISSION_GRANTED) {
                    ActivityCompat.requestPermissions(
                            DriverMapsActivity.this, new
String[]{Manifest.permission.CALL_PHONE},
                            123);
                }
                else {
                    Intent intent = new Intent(Intent.ACTION_CALL, Uri.parse("tel: "
+ currentCustomerPhoneNumber));
                    startActivity(intent);
                }
            }
        });

        getAssignedCustomerRequest();
    }

    private void getAssignedCustomerRequest() {
        assignedCustomerRef =
FirebaseDatabase.getInstance().getReference().child("Users")
                .child("Drivers").child(driverID).child("CustomerRideID");

        assignedCustomerRef.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot snapshot) {
                if (snapshot.exists()) {
                    customerID = snapshot.getValue().toString();

                    getAssignedCustomerPosition();
                }
                else {
                    customerID = "";
                    customerInfoRelativeLayout.setVisibility(View.INVISIBLE);
                    calCurrentCustomerButton.setVisibility(View.INVISIBLE);
                    locationDriverTextView.setText("Приємного користування");
                    if (driverPickUpMarker != null) {
                        driverPickUpMarker.remove();
                    }
                    if (assignedCustomerPositionListener != null) {

assignedCustomerPositionRef.removeEventListener(assignedCustomerPositionListener);
                    }
```

```
                }
            }

            @Override
            public void onCancelled(@NonNull DatabaseError error) {

            }
        });
    }

    private void getAssignedCustomerPosition() {
        assignedCustomerPositionRef =
FirebaseDatabase.getInstance().getReference().child("Customers Requests")
                .child(customerID).child("l");

        assignedCustomerPositionListener =
assignedCustomerPositionRef.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot snapshot) {
                if (snapshot.exists()) {
                    List<Object> customerPositionMap = (List<Object>)
snapshot.getValue();
                    double locationLat =
Double.parseDouble(customerPositionMap.get(0).toString());
                    double locationLng =
Double.parseDouble(customerPositionMap.get(1).toString());

                    LatLng customerLatLng = new LatLng(locationLat, locationLng);
                    driverPickUpMarker = mMap.addMarker(new
MarkerOptions().position(customerLatLng).title("Забрати клієнта тут")

.icon(BitmapDescriptorFactory.fromResource(R.drawable.location_image)));
                    mMap.moveCamera(CameraUpdateFactory.newLatLng(customerLatLng));
                    mMap.animateCamera(CameraUpdateFactory.zoomTo(11));

                    customerInfoRelativeLayout.setVisibility(View.VISIBLE);
                    calCurrentCustomerButton.setVisibility(View.VISIBLE);
                    DatabaseReference reference =
FirebaseDatabase.getInstance().getReference()
                            .child("Users").child("Customers").child(customerID);

                    reference.addValueEventListener(new ValueEventListener() {
                        @SuppressLint({"DefaultLocale", "SetTextI18n"})
                        @Override
                        public void onDataChange(@NonNull DataSnapshot dataSnapshot)
{
                            if (dataSnapshot.exists() &&
dataSnapshot.getChildrenCount() > 0) {
                                String name =
dataSnapshot.child("Name").getValue().toString();
                                String phone =
dataSnapshot.child("Phone").getValue().toString();
                                String carName =
dataSnapshot.child("CarName").getValue().toString();
                                String carNumber =
dataSnapshot.child("CarNumber").getValue().toString();

                                customerMapsName.setText(name);
                                customerMapsPhoneNumber.setText(phone);
                                customerMapsCarInfo.setText(carName);
```

```
                                customerMapsCarNumber.setText(carNumber);

                                currentCustomerPhoneNumber = phone;
                            }
                        }

                        @Override
                        public void onCancelled(@NonNull DatabaseError databaseError)
{}
                    });

                    Location location1 = new Location("");
                    location1.setLatitude(driverCurrentPosition.latitude);
                    location1.setLongitude(driverCurrentPosition.longitude);

                    Location location2 = new Location("");
                    location2.setLatitude(customerLatLng.latitude);
                    location2.setLongitude(customerLatLng.longitude);

                    float Distance = location1.distanceTo(location2);
                    float roadDistance = Distance / 1000;
                    locationDriverTextView.setText("Відстань до авто " +
String.format("%.2f", roadDistance) + " км");


                    if (Distance == 0) {

                        Intent resultIntent = new Intent(DriverMapsActivity.this,
DriverOrderResultActivity.class);
                        startActivity(resultIntent);
                    }
                }
            }

            @Override
            public void onCancelled(@NonNull DatabaseError error) {

            }
        });
    }


    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;
        buildGoogleApiClient();
        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED &&
ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_COARSE_LOCATION)
!= PackageManager.PERMISSION_GRANTED) {
            return;
        }
        mMap.setMyLocationEnabled(true);
    }


    @Override
    public void onConnected(@Nullable Bundle bundle) {
        locationRequest = new LocationRequest();
        locationRequest.setInterval(100000);
```

```
        locationRequest.setFastestInterval(100000);
        locationRequest.setPriority(locationRequest.PRIORITY_HIGH_ACCURACY);

        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION)
                != PackageManager.PERMISSION_GRANTED &&
ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_COARSE_LOCATION)
!= PackageManager.PERMISSION_GRANTED) {
            return;
        }
        LocationServices.FusedLocationApi.requestLocationUpdates(googleApiClient,
locationRequest, this);
    }

    @Override
    public void onConnectionSuspended(int i) {

    }

    @Override
    public void onConnectionFailed(@NonNull ConnectionResult connectionResult) {

    }

    @SuppressLint("DefaultLocale")
    @Override
    public void onLocationChanged(Location location) {
        if (getApplicationContext() != null) {
            lastLocation = location;
            LatLng latLng = new LatLng(location.getLatitude(),
location.getLongitude());
            mMap.moveCamera(CameraUpdateFactory.newLatLng(latLng));
            mMap.animateCamera(CameraUpdateFactory.zoomTo(12));

            driverCurrentPosition = new LatLng(location.getLatitude(),
location.getLongitude());

            menuCurrentDriverCoordinates = location;
            menuDriverLatitude = String.format("%.3f", location.getLatitude());
            menuDriverLongitude = String.format("%.3f", location.getLongitude());

            String userID = FirebaseAuth.getInstance().getCurrentUser().getUid();

            DatabaseReference availableDriversRef =
FirebaseDatabase.getInstance().getReference().child("Available Drivers");
            GeoFire geoFireAvailability = new GeoFire(availableDriversRef);

            DatabaseReference driverWorkingRef =
FirebaseDatabase.getInstance().getReference().child("Driver Working");
            GeoFire geoFireWorking = new GeoFire(driverWorkingRef);

            switch (customerID) {
                case "":
                    geoFireWorking.removeLocation(userID);
                    geoFireAvailability.setLocation(userID, new
GeoLocation(location.getLatitude(), location.getLongitude()));
                    break;
                default:
                    geoFireAvailability.removeLocation(userID);
```

```
                    geoFireWorking.setLocation(userID, new
GeoLocation(location.getLatitude(), location.getLongitude()));
                    break;
            }
        }
    }

    protected synchronized void buildGoogleApiClient() {
        googleApiClient = new GoogleApiClient.Builder(this)
                .addConnectionCallbacks(this)
                .addOnConnectionFailedListener(this)
                .addApi(LocationServices.API)
                .build();
        googleApiClient.connect();
    }

    @Override
    protected void onStop() {
        super.onStop();
        if(!currentLogOutDriverStatus) {
            DisconnectDriver();
        }
    }

    private void DisconnectDriver() {
        String userID = FirebaseAuth.getInstance().getCurrentUser().getUid();
        DatabaseReference availableDriversRef =
FirebaseDatabase.getInstance().getReference().child("Available Drivers");
        GeoFire geoFire = new GeoFire(availableDriversRef);
        geoFire.removeLocation(userID);
    }

    private void LogoutDriver() {
        Intent welcomeIntent = new Intent(DriverMapsActivity.this,
RoleSelectionActivity.class);
        startActivity(welcomeIntent);
        finish();
    }

    private void CheckCurrentUserOrdering() {
        String driverID = FirebaseAuth.getInstance().getCurrentUser().getUid();
        DatabaseReference checkRef =
FirebaseDatabase.getInstance().getReference().child("Driver Working")
                .child(driverID);

        checkRef.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot snapshot) {
                if (snapshot.exists()) {
                    currentUserOrderingStatus = true;
                }
            }

            @Override
            public void onCancelled(@NonNull DatabaseError error) {}
        });
    }
}
```

Клас «DriverMenuActivity»

```
package com.example.autosos;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.RelativeLayout;
import android.widget.TextView;
import android.widget.Toast;

import com.firebase.geofire.GeoFire;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
import java.util.HashMap;

public class DriverMenuActivity extends AppCompatActivity {

    EditText menuDriverName, menuDriverPhone, menuDriverCarInfo, menuDriverCarNumber;
    ImageView closeDriverMenuButton, saveDriverMenuButton;
    TextView latitudeDriverInfo, longitudeDriverInfo;
    RelativeLayout exitDriverButton, deleteDriverButton, siteDriverButton,
aboutDriverButton;

    private FirebaseAuth mAuth;
    private DatabaseReference databaseReference, driverAvailableRef;

    private Boolean currentLogOutDriverStatus = false;
    private Boolean currentDriverWorkingStatus = false;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_driver_menu);

        menuDriverName = (EditText) findViewById(R.id.menuDriverName);
        menuDriverPhone = (EditText) findViewById(R.id.menuDriverPhone);
        menuDriverCarInfo = (EditText) findViewById(R.id.menuDriverCarInfo);
        menuDriverCarNumber = (EditText) findViewById(R.id.menuDriverCarNumber);

        closeDriverMenuButton = (ImageView) findViewById(R.id.closeDriverMenuButton);
        saveDriverMenuButton = (ImageView) findViewById(R.id.saveDriverMenuButton);

        latitudeDriverInfo = (TextView) findViewById(R.id.latitudeDriverInfo);
        latitudeDriverInfo.setText(DriverMapsActivity.menuDriverLatitude);
        longitudeDriverInfo = (TextView) findViewById(R.id.longitudeDriverInfo);
```

```
        longitudeDriverInfo.setText(DriverMapsActivity.menuDriverLongitude);

        exitDriverButton = (RelativeLayout) findViewById(R.id.exitDriverButton);
        deleteDriverButton = (RelativeLayout) findViewById(R.id.deleteDriverButton);
        siteDriverButton = (RelativeLayout) findViewById(R.id.siteDriverButton);
        aboutDriverButton = (RelativeLayout) findViewById(R.id.aboutDriverButton);

        mAuth = FirebaseAuth.getInstance();

        databaseReference =
FirebaseDatabase.getInstance().getReference().child("Users").child("Drivers");
        getUserInformation();

        closeDriverMenuButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent backIntent = new Intent(DriverMenuActivity.this,
DriverMapsActivity.class);
                startActivity(backIntent);
            }
        });

        saveDriverMenuButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                    validateAndSaveInfo();
                }
        });

        exitDriverButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (currentDriverWorkingStatus) {
                    Toast.makeText(DriverMenuActivity.this, "Завершіть Ваше
замовлення", Toast.LENGTH_SHORT).show();
                }
                else {
                    currentLogOutDriverStatus = true;
                    LogoutDriver();
                    DisconnectDriver();
                    mAuth.signOut();
                }
            }
        });

        deleteDriverButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (currentDriverWorkingStatus) {
                    Toast.makeText(DriverMenuActivity.this, "Завершіть Ваше
замовлення", Toast.LENGTH_SHORT).show();
                }
                else {
                    removeCurrentDriver();
                    LogoutDriver();
                }
            }
        });

        siteDriverButton.setOnClickListener(new View.OnClickListener() {
```

```
            @Override
            public void onClick(View v) {
                Intent siteIntent = new Intent(DriverMenuActivity.this,
SiteActivity.class);
                siteIntent.putExtra("type", "Driver");
                startActivity(siteIntent);
            }
        });

        aboutDriverButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent aboutIntent = new Intent(DriverMenuActivity.this,
AboutActivity.class);
                aboutIntent.putExtra("type", "Driver");
                startActivity(aboutIntent);
            }
        });
    }

    private void validateAndSaveInfo() {
        if (TextUtils.isEmpty(menuDriverName.getText().toString())) {
            Toast.makeText(this, "Заповніть поле Name", Toast.LENGTH_SHORT).show();
        }
        else if (TextUtils.isEmpty(menuDriverPhone.getText().toString())) {
            Toast.makeText(this, "Заповніть поле Phone", Toast.LENGTH_SHORT).show();
        }
        else if (TextUtils.isEmpty(menuDriverCarInfo.getText().toString())) {
            Toast.makeText(this, "Заповніть поле Auto", Toast.LENGTH_SHORT).show();
        }
        else if (TextUtils.isEmpty(menuDriverCarNumber.getText().toString())) {
            Toast.makeText(this, "Заповніть поле Auto", Toast.LENGTH_SHORT).show();
        }
        else {
            HashMap<String, Object> userMap = new HashMap<>();
            userMap.put("DriverID", mAuth.getCurrentUser().getUid());
            userMap.put("Name", menuDriverName.getText().toString());
            userMap.put("Phone", menuDriverPhone.getText().toString());
            userMap.put("CarName", menuDriverCarInfo.getText().toString());
            userMap.put("CarNumber", menuDriverCarNumber.getText().toString());


databaseReference.child(mAuth.getCurrentUser().getUid()).updateChildren(userMap);
            startActivity(new Intent(DriverMenuActivity.this,
DriverMapsActivity.class));
        }
    }

    private void getUserInformation() {


databaseReference.child(mAuth.getCurrentUser().getUid()).addValueEventListener(new
ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot snapshot) {
                if (snapshot.exists() && snapshot.getChildrenCount() > 0) {
                    String name = snapshot.child("Name").getValue().toString();
                    String phone = snapshot.child("Phone").getValue().toString();
                    String carInfo = snapshot.child("CarName").getValue().toString();
```

```
                    String carNumber =
snapshot.child("CarNumber").getValue().toString();

                    menuDriverName.setText(name);
                    menuDriverPhone.setText(phone);
                    menuDriverCarInfo.setText(carInfo);
                    menuDriverCarNumber.setText(carNumber);
                }
            }

            @Override
            public void onCancelled(@NonNull DatabaseError error) {

            }
        });
    }

    private void DisconnectDriver() {
        String driverID = FirebaseAuth.getInstance().getCurrentUser().getUid();

        DatabaseReference driverDatabaseRef =
FirebaseDatabase.getInstance().getReference()
                .child("Users").child("Drivers").child(driverID).child("Online
Status");
        driverDatabaseRef.setValue(false);

        DatabaseReference availableDriversRef =
FirebaseDatabase.getInstance().getReference().child("Available Drivers");
        GeoFire geoFire = new GeoFire(availableDriversRef);
        geoFire.removeLocation(driverID);
    }

    private void LogoutDriver() {
        Intent roleSelectionIntent = new Intent(DriverMenuActivity.this,
RoleSelectionActivity.class);
        startActivity(roleSelectionIntent);
        finish();
    }

    private void CheckCurrentDriverWorking() {
        String driverID = FirebaseAuth.getInstance().getCurrentUser().getUid();
        DatabaseReference checkRef =
FirebaseDatabase.getInstance().getReference().child("Driver
Working").child(driverID);

        checkRef.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot snapshot) {
                if (snapshot.exists()) {
                    currentDriverWorkingStatus = true;
                }
            }

            @Override
            public void onCancelled(@NonNull DatabaseError error) {}
        });
    }

    private void removeCurrentDriver() {
```

```
        String driverID = FirebaseAuth.getInstance().getCurrentUser().getUid();
        Task<Void> removeReference =
FirebaseDatabase.getInstance().getReference().child("Users")
                .child("Drivers").child(driverID).removeValue();

        FirebaseUser currentUser = mAuth.getCurrentUser();
        currentUser.delete();
    }
}
```

Клас «DriverOrderResultActivity»

```
package com.example.autosos;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class DriverOrderResultActivity extends AppCompatActivity {

    Button driverResultButton;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_driver_order_result);

        driverResultButton = (Button) findViewById(R.id.driverResultButton);

        driverResultButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent backMapIntent = new Intent(DriverOrderResultActivity.this,
DriverMapsActivity.class);
                startActivity(backMapIntent);
            }
        });
    }
}
```

Клас «DriverRegistrationActivity»

```
package com.example.autosos;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
```

```
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

public class DriverRegistrationActivity extends AppCompatActivity {

    EditText registrationEmailDriverEditText, registrationPasswordDriverEditText;
    Button registrationDriverButton;

    FirebaseAuth firebaseAuth;
    DatabaseReference driverDatabaseRef;
    String onlineDriverID;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_driver_registration);

        registrationEmailDriverEditText = (EditText)
findViewById(R.id.registrationEmailDriverEditText);
        registrationPasswordDriverEditText = (EditText)
findViewById(R.id.registrationPasswordDriverEditText);
        registrationDriverButton = (Button)
findViewById(R.id.registrationDriverButton);

        firebaseAuth = FirebaseAuth.getInstance();

        registrationDriverButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if
(TextUtils.isEmpty(registrationEmailDriverEditText.getText().toString())) {
                    Toast.makeText(DriverRegistrationActivity.this, "Заповніть поле
email-адреси",
                            Toast.LENGTH_SHORT).show();
                }
                else if
(TextUtils.isEmpty(registrationPasswordDriverEditText.getText().toString())) {
                    Toast.makeText(DriverRegistrationActivity.this, "Заповніть поле з
паролем",
                            Toast.LENGTH_SHORT).show();
                }
                else {
                    String email =
registrationEmailDriverEditText.getText().toString();
                    String password =
registrationPasswordDriverEditText.getText().toString();
                    RegistrationDriverMethod(email, password);
                }
            }
        });
    }

    private void RegistrationDriverMethod(String email, String password) {
        firebaseAuth.createUserWithEmailAndPassword(email,
password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
            @Override
```

```
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful()) {

                    onlineDriverID = firebaseAuth.getCurrentUser().getUid();
                    driverDatabaseRef = FirebaseDatabase.getInstance().getReference()
                            .child("Users").child("Drivers").child(onlineDriverID);

                    Intent mainScreenIntent = new
Intent(DriverRegistrationActivity.this, DriverInfoRegistrationActivity.class);
                    startActivity(mainScreenIntent);

                    Toast.makeText(DriverRegistrationActivity.this, "Реєстрація
успішна",
                            Toast.LENGTH_SHORT).show();
                }
                else {
                    Toast.makeText(DriverRegistrationActivity.this, "Помилка
реєстрації нового користувача",
                            Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
}
```

Клас «DriverWelcomeActivity»

```
package com.example.autosos;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

public class DriverWelcomeActivity extends AppCompatActivity {

    EditText editTextDriverEmail, editTextDriverPassword;
    Button signInDriverButton, roleBackDriverButton, signUpDriverButton;

    FirebaseAuth firebaseAuth;
    DatabaseReference driverDatabaseRef, driverAvailableRef;
    String onlineDriverID;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_driver_welcome);
```

```
        editTextDriverEmail = (EditText) findViewById(R.id.editTextDriverEmail);
        editTextDriverPassword = (EditText)
findViewById(R.id.editTextDriverPassword);
        signInDriverButton = (Button) findViewById(R.id.signInDriverButton);
        signUpDriverButton = (Button) findViewById(R.id.signUpDriverButton);
        roleBackDriverButton = (Button) findViewById(R.id.roleBackDriverButton);

        firebaseAuth = FirebaseAuth.getInstance();

        signInDriverButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String email = editTextDriverEmail.getText().toString();
                String password = editTextDriverPassword.getText().toString();
                SignInDriverMethod(email, password);
            }
        });

        signUpDriverButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent registrationIntent = new Intent(DriverWelcomeActivity.this,
DriverRegistrationActivity.class);
                startActivity(registrationIntent);
            }
        });

        roleBackDriverButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent roleSelectionIntent = new Intent(DriverWelcomeActivity.this,
RoleSelectionActivity.class);
                startActivity(roleSelectionIntent);
            }
        });
    }

    private void SignInDriverMethod(String email, String password) {
        firebaseAuth.signInWithEmailAndPassword(email,
password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful()) {
                    Toast.makeText(DriverWelcomeActivity.this, "Вхід успішний",
Toast.LENGTH_SHORT).show();

                    onlineDriverID = firebaseAuth.getCurrentUser().getUid();
                    driverDatabaseRef = FirebaseDatabase.getInstance().getReference()

.child("Users").child("Drivers").child(onlineDriverID).child("Online Status");
                    driverDatabaseRef.setValue(true);

                    Intent mainScreenIntent = new Intent(DriverWelcomeActivity.this,
DriverMapsActivity.class);
                    startActivity(mainScreenIntent);
                }
                else {
                    Toast.makeText(DriverWelcomeActivity.this, "Помилка входу",
                            Toast.LENGTH_SHORT).show();
```

```
                }
            }
        });
    }
}
```

Клас «MainActivity»

```
package com.example.autosos;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Thread thread = new Thread() {
            @Override
            public void run() {
                super.run();
                {
                    try {
                        sleep(3000);
                    } catch (Exception e) {
                        e.printStackTrace();
                    } finally {
                        Intent roleSelectionIntent = new Intent(MainActivity.this,
RoleSelectionActivity.class);
                        startActivity(roleSelectionIntent);
                    }
                }
            }
        };
        thread.start();
    }

    @Override
    protected void onPause() {
        super.onPause();
        finish();
    }
}
```

Клас «RoleSelectionActivity»

```
package com.example.autosos;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
```

```
import android.widget.ImageView;

public class RoleSelectionActivity extends AppCompatActivity {

    ImageView roleSelectionCustomerButton, roleSelectionDriverButton;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_role_selection);

        roleSelectionCustomerButton = (ImageView)
findViewById(R.id.roleSelectionCustomerButton);
        roleSelectionDriverButton = (ImageView)
findViewById(R.id.roleSelectionDriverButton);

        roleSelectionCustomerButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent welcomeIntent = new Intent(RoleSelectionActivity.this,
CustomerWelcomeActivity.class);
                startActivity(welcomeIntent);
            }
        });

        roleSelectionDriverButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent welcomeDriverIntent = new Intent(RoleSelectionActivity.this,
DriverWelcomeActivity.class);
                startActivity(welcomeDriverIntent);
            }
        });
    }
}
```

Клас «SiteActivity»

```
package com.example.autosos;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class SiteActivity extends AppCompatActivity {

    Button siteBackMenuButton;

    private String getType;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_site);

        getType = getIntent().getStringExtra("type");
```

```
        siteBackMenuButton = (Button) findViewById(R.id.siteBackMenuButton);

        siteBackMenuButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent backMenuIntent;
                if (getType.equals("Customer")) {
                    backMenuIntent = new Intent(SiteActivity.this,
CustomerMenuActivity.class);
                    startActivity(backMenuIntent);
                }
                else if (getType.equals("Driver")) {
                    backMenuIntent = new Intent(SiteActivity.this,
DriverMenuActivity.class);
                    startActivity(backMenuIntent);
                }
            }
        });
    }
}
```

Додаток Г

У даному додатку наведено зміст файлів типу XML, в яких відтворено увесь зовнішній інтерфейс відповідно. Перелік сформовано в алфавітному порядку.

Файл «activity_about»

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/other_background"
    tools:context=".AboutActivity">

    <ImageView
        android:id="@+id/siteImage"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="35dp"
        android:layout_width="330dp"
        android:layout_height="330dp"
        android:background="@drawable/about_app_image" />

    <TextView
        android:id="@+id/siteTextView"
        android:layout_centerHorizontal="true"
        android:layout_width="320dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/siteImage"
        android:textAlignment="center"
        android:textStyle="bold"
        android:textColor="@color/black"
        android:textSize="30dp"
        android:text="Мобільний додаток AutoSOS розроблений у навчальних цілях та є
лише MVP майбутнього повноцінного продукту"/>

    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/aboutBackMenuButton"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="10dp"
        android:layout_width="150dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/siteTextView"
        android:background="@drawable/welcome_button_style"
        android:text="Назад"
        android:textAlignment="center"
        android:textColor="@color/white"
        android:textSize="22dp" />

</RelativeLayout>
```

Файл «activity_customer_info_registration»

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/customer_welcome_background"
    tools:context=".CustomerInfoRegistrationActivity">

    <TextView
        android:id="@+id/registrationTextView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="195dp"
        android:text="Ваше ім'я"
        android:textSize="20dp"
        android:textColor="@color/black" />

    <EditText
        android:id="@+id/customerInfoName"
        android:layout_width="245dp"
        android:layout_height="50dp"
        android:layout_below="@+id/registrationTextView1"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="5dp"
        android:background="@drawable/welcome_edittext_style"
        android:ems="10"
        android:hint="Ім'я користувача"
        android:inputType="text"
        android:textAlignment="center"
        android:textColor="@color/black" />

    <TextView
        android:id="@+id/registrationTextView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/customerInfoName"
        android:layout_marginTop="5dp"
        android:text="Номер телефону"
        android:textSize="20dp"
        android:textColor="@color/black" />

    <EditText
        android:id="@+id/customerInfoPhone"
        android:layout_width="245dp"
        android:layout_height="50dp"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/registrationTextView2"
        android:layout_marginTop="5dp"
        android:background="@drawable/welcome_edittext_style"
        android:ems="10"
        android:hint="+380 (__) ___-__-__"
        android:inputType="phone"
        android:textAlignment="center"
        android:textColor="@color/black" />
```

```
<TextView
    android:id="@+id/registrationTextView3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_below="@+id/customerInfoPhone"
    android:layout_marginTop="5dp"
    android:text="Марка Вашого авто"
    android:textSize="20dp"
    android:textColor="@color/black" />

<EditText
    android:id="@+id/customerInfoCarName"
    android:layout_width="245dp"
    android:layout_height="50dp"
    android:layout_centerHorizontal="true"
    android:layout_below="@+id/registrationTextView3"
    android:layout_marginTop="5dp"
    android:background="@drawable/welcome_edittext_style"
    android:ems="10"
    android:hint="Марка та модель авто"
    android:textAlignment="center"
    android:textColor="@color/black"
    android:inputType="text" />

<TextView
    android:id="@+id/registrationTextView4"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_below="@+id/customerInfoCarName"
    android:layout_marginTop="5dp"
    android:text="Номер Вашого авто"
    android:textSize="20dp"
    android:textColor="@color/black" />

<EditText
    android:id="@+id/customerInfoCarNumber"
    android:layout_width="245dp"
    android:layout_height="50dp"
    android:layout_centerHorizontal="true"
    android:layout_below="@+id/registrationTextView4"
    android:layout_marginTop="5dp"
    android:background="@drawable/welcome_edittext_style"
    android:ems="10"
    android:hint="Номер авто"
    android:textAlignment="center"
    android:textColor="@color/black"
    android:inputType="text" />

<androidx.appcompat.widget.AppCompatButton
    android:id="@+id/customerCreateButton"
    android:layout_width="245dp"
    android:layout_height="wrap_content"
    android:layout_below="@+id/customerInfoCarNumber"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="10dp"
    android:background="@drawable/welcome_button_style"
    android:textAlignment="center"
```

```
        android:textColor="@color/white"
        android:text="Створити клієнта"
        android:textSize="20dp"/>

</RelativeLayout>
```

Файл «activity_customer_maps»

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:map="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".CustomerMapsActivity" >

    <fragment
        android:id="@+id/map"
        android:name="com.google.android.gms.maps.SupportMapFragment"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/calCurrentDriverButton"
        android:layout_width="370dp"
        android:layout_height="60dp"
        android:layout_above="@+id/driverInfoRelativeLayout"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="10dp"
        android:background="@drawable/map_edittext_style"
        android:text="Подзвонити водію евакуатора"
        android:textSize="20dp"
        android:textStyle="bold" />

    <RelativeLayout
        android:id="@+id/driverInfoRelativeLayout"
        android:layout_width="370dp"
        android:layout_height="100dp"
        android:layout_above="@+id/rel2"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="10dp"
        android:background="@drawable/map_edittext_style">

        <de.hdodenhof.circleimageview.CircleImageView
            android:id="@+id/driverImage"
            android:layout_width="80dp"
            android:layout_height="80dp"
            android:layout_centerVertical="true"
            android:layout_marginLeft="10dp"
            android:src="@drawable/avatar_driver_image"
            map:civ_border_color="#dd2e44"
            map:civ_border_width="3dp" />

        <TextView
            android:id="@+id/driverMapsName"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="10dp"
            android:layout_marginTop="21dp"
            android:layout_toEndOf="@+id/driverImage"
```

```
            android:text="Username"
            android:textColor="@color/black"
            android:textSize="18dp" />

        <TextView
            android:id="@+id/driverMapsPhoneNumber"
            android:layout_width="140dp"
            android:layout_height="wrap_content"
            android:layout_below="@+id/driverMapsName"
            android:layout_marginLeft="10dp"
            android:layout_marginTop="7dp"
            android:layout_toEndOf="@+id/driverImage"
            android:text="380000000000"
            android:textColor="@color/black"
            android:textSize="18dp" />

        <View
            android:id="@+id/view"
            android:layout_width="5dp"
            android:layout_height="65dp"
            android:layout_centerVertical="true"
            android:layout_toEndOf="@+id/driverMapsPhoneNumber"
            android:background="#dd2e44" />

        <TextView
            android:id="@+id/driverMapsCarInfo"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="10dp"
            android:layout_marginTop="21dp"
            android:layout_toEndOf="@+id/view"
            android:text="Car Info"
            android:textColor="@color/black"
            android:textSize="18dp" />

        <TextView
            android:id="@+id/driverMapsCarNumber"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/driverMapsCarInfo"
            android:layout_marginLeft="10dp"
            android:layout_marginTop="7dp"
            android:layout_toEndOf="@+id/view"
            android:text="Car Number"
            android:textColor="@color/black"
            android:textSize="18dp" />

    </RelativeLayout>

    <RelativeLayout
        android:id="@+id/locationRelativeLayout"
        android:layout_width="300dp"
        android:layout_height="60dp"
        android:layout_marginLeft="15dp"
        android:layout_marginTop="15dp"
        android:background="@drawable/map_edittext_style">

        <TextView
            android:id="@+id/locationEditText"
            android:layout_width="300dp"
```

```
        android:layout_height="wrap_content"
        android:layout_centerVertical="true"
        android:text="Приємного користування"
        android:textAlignment="center"
        android:textColor="@color/black"
        android:textSize="22dp"
        android:textStyle="bold" />
</RelativeLayout>

<androidx.appcompat.widget.AppCompatButton
    android:id="@+id/customerCancelOrderButton"
    android:layout_width="140dp"
    android:layout_height="60dp"
    android:layout_below="@+id/locationRelativeLayout"
    android:layout_marginLeft="15dp"
    android:layout_marginTop="10dp"
    android:background="@drawable/map_edittext_style"
    android:text="Відмінити"
    android:textSize="20dp"
    android:textStyle="bold" />

<RelativeLayout
    android:id="@+id/customerUpdateDriverPosition"
    android:layout_width="60dp"
    android:layout_height="60dp"
    android:layout_below="@+id/locationRelativeLayout"
    android:layout_marginLeft="10dp"
    android:layout_marginTop="10dp"
    android:layout_toEndOf="@+id/customerCancelOrderButton"
    android:background="@drawable/map_edittext_style">

    <ImageView
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:layout_centerInParent="true"
        android:src="@drawable/restart_button" />

</RelativeLayout>

<RelativeLayout
    android:id="@+id/rel2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    android:layout_centerHorizontal="true">

    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/customerMenuButton"
        android:layout_width="90dp"
        android:layout_height="60dp"
        android:layout_marginBottom="35dp"
        android:background="@drawable/map_edittext_style"
        android:text="Меню"
        android:textSize="20dp"
        android:textStyle="bold" />

    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/callDriverButton"
        android:layout_width="265dp"
        android:layout_height="60dp"
```

```
            android:layout_marginLeft="15dp"
            android:layout_marginBottom="35dp"
            android:layout_toEndOf="@+id/customerMenuButton"
            android:background="@drawable/map_edittext_style"
            android:text="Викликати евакуатор"
            android:textSize="20dp"
            android:textStyle="bold" />
    </RelativeLayout>
</RelativeLayout>
```

Файл «activity_customer_menu»

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".CustomerMenuActivity">

    <com.google.android.material.appbar.AppBarLayout
        android:id="@+id/appBarCustomerMenu"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >

        <androidx.appcompat.widget.Toolbar
            android:id="@+id/toolBarCustomerMenu"
            android:background="#dd2e44"
            android:layout_width="match_parent"
            android:layout_height="60dp">

            <RelativeLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content">

                <ImageView
                    android:id="@+id/closeCustomerMenuButton"
                    android:layout_width="40dp"
                    android:layout_height="50dp"
                    android:src="@drawable/cancel_info_button"
                    app:tint="@color/white" />

                <ImageView
                    android:id="@+id/saveCustomerMenuButton"
                    android:layout_width="40dp"
                    android:layout_height="50dp"
                    android:src="@drawable/save_info_button"
                    android:layout_alignParentEnd="true"
                    android:layout_marginRight="20dp"
                    app:tint="@color/white"/>
            </RelativeLayout>

        </androidx.appcompat.widget.Toolbar>

    </com.google.android.material.appbar.AppBarLayout>

    <RelativeLayout
        android:id="@+id/menuRelativeLayout1"
        android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:layout_below="@+id/appBarCustomerMenu">

        <de.hdodenhof.circleimageview.CircleImageView
xmlns:app="http://schemas.android.com/apk/res-auto"
            android:id="@+id/avatarImage"
            android:layout_width="200dp"
            android:layout_height="250dp"
            android:layout_marginLeft="20dp"
            android:src="@drawable/avatar_customer_image"
            app:civ_border_color="#dd2e44"
            app:civ_border_width="4dp" />

        <EditText
            android:id="@+id/menuCustomerName"
            android:layout_width="150dp"
            android:layout_height="40dp"
            android:layout_marginLeft="10dp"
            android:layout_marginTop="35dp"
            android:layout_toEndOf="@+id/avatarImage"
            android:textSize="17dp" />

        <EditText
            android:id="@+id/menuCustomerPhone"
            android:layout_width="150dp"
            android:layout_height="40dp"
            android:layout_below="@id/menuCustomerName"
            android:layout_marginLeft="10dp"
            android:layout_toEndOf="@+id/avatarImage"
            android:textSize="17dp"
            android:inputType="phone"/>

        <EditText
            android:id="@+id/menuCustomerCarInfo"
            android:layout_width="150dp"
            android:layout_height="40dp"
            android:layout_toEndOf="@+id/avatarImage"
            android:layout_below="@id/menuCustomerPhone"
            android:layout_marginLeft="10dp"
            android:textSize="17dp" />

        <EditText
            android:id="@+id/menuCustomerCarNumber"
            android:layout_width="150dp"
            android:layout_height="49dp"
            android:layout_below="@id/menuCustomerCarInfo"
            android:layout_marginLeft="10dp"
            android:layout_toEndOf="@+id/avatarImage"
            android:textSize="17dp" />

    </RelativeLayout>

    <RelativeLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/menuRelativeLayout1"
        android:layout_marginTop="20dp"
        android:layout_centerHorizontal="true">

        <RelativeLayout
```

```
android:id="@+id/coordinatesRelativeLayout"
android:layout_width="220dp"
android:layout_height="180dp"
android:background="@drawable/map_edittext_style">

<TextView
    android:id="@+id/menuTextview1"
    android:layout_width="200dp"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="10dp"
    android:text="Поточні координати"
    android:textAlignment="center"
    android:textColor="@color/black"
    android:textSize="22dp"
    android:textStyle="bold" />

<View
    android:id="@+id/view"
    android:layout_width="165dp"
    android:layout_height="3dp"
    android:layout_below="@+id/menuTextview1"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="3dp"
    android:background="#dd2e44" />

<TextView
    android:id="@+id/menuTextview2"
    android:layout_width="100dp"
    android:layout_height="wrap_content"
    android:layout_below="@+id/view"
    android:layout_marginLeft="15dp"
    android:layout_marginTop="15dp"
    android:text="Широта"
    android:textAlignment="center"
    android:textColor="@color/black"
    android:textSize="20dp"
    android:textStyle="bold" />

<TextView
    android:id="@+id/latitudeCustomerInfo"
    android:layout_width="90dp"
    android:layout_height="wrap_content"
    android:layout_below="@+id/view"
    android:layout_marginTop="15dp"
    android:layout_toEndOf="@+id/menuTextview2"
    android:text="000"
    android:textAlignment="center"
    android:textColor="#dd2e44"
    android:textSize="20dp"
    android:textStyle="bold" />

<TextView
    android:id="@+id/menuTextview3"
    android:layout_width="100dp"
    android:layout_height="wrap_content"
    android:layout_below="@+id/menuTextview2"
    android:layout_marginLeft="15dp"
    android:layout_marginTop="10dp"
    android:text="Довгота"
```

```
        android:textAlignment="center"
        android:textColor="@color/black"
        android:textSize="20dp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/longitudeCustomerInfo"
        android:layout_width="90dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/latitudeCustomerInfo"
        android:layout_marginTop="10dp"
        android:layout_toEndOf="@+id/menuTextview3"
        android:text="000"
        android:textAlignment="center"
        android:textColor="#dd2e44"
        android:textSize="20dp"
        android:textStyle="bold" />
</RelativeLayout>

<RelativeLayout
    android:id="@+id/aboutCustomerButton"
    android:layout_width="220dp"
    android:layout_height="65dp"
    android:layout_below="@id/coordinatesRelativeLayout"
    android:layout_marginTop="20dp"
    android:background="@drawable/map_edittext_style">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:text="Про додаток"
        android:textAlignment="center"
        android:textColor="@color/black"
        android:textSize="22dp"
        android:textStyle="bold" />

</RelativeLayout>

<RelativeLayout
    android:id="@+id/deleteCustomerButton"
    android:layout_width="220dp"
    android:layout_height="65dp"
    android:background="@drawable/map_edittext_style"
    android:layout_below="@id/aboutCustomerButton"
    android:layout_marginTop="20dp">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true"
        android:text="Видалити аккаунт"
        android:textAlignment="center"
        android:textColor="@color/black"
        android:textSize="22dp"
        android:textStyle="bold" />
</RelativeLayout>

<RelativeLayout
```

```
            android:id="@+id/siteCustomerButton"
            android:layout_width="130dp"
            android:layout_height="180dp"
            android:layout_marginLeft="20dp"
            android:layout_toEndOf="@+id/coordinatesRelativeLayout"
            android:background="@drawable/map_edittext_style">

            <ImageView
                android:id="@+id/siteImage"
                android:layout_width="110dp"
                android:layout_height="120dp"
                android:layout_centerHorizontal="true"
                android:layout_marginTop="20dp"
                android:src="@drawable/menu_site_button" />

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_below="@+id/siteImage"
                android:layout_centerHorizontal="true"
                android:text="Сайт додатку"
                android:textAlignment="center"
                android:textColor="@color/black"
                android:textSize="16dp"
                android:textStyle="bold" />

        </RelativeLayout>

        <RelativeLayout
            android:id="@+id/exitCustomerButton"
            android:layout_marginTop="20dp"
            android:layout_width="130dp"
            android:layout_height="150dp"
            android:background="@drawable/map_edittext_style"
            android:layout_below="@+id/siteCustomerButton"
            android:layout_toEndOf="@+id/coordinatesRelativeLayout"
            android:layout_marginLeft="20dp">

            <ImageView
                android:id="@+id/exitImage"
                android:layout_width="110dp"
                android:layout_height="150dp"
                android:layout_centerInParent="true"
                android:src="@drawable/menu_exit_button" />

        </RelativeLayout>
    </RelativeLayout>

</RelativeLayout>
```

Файл «activity_customer_order_result»

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/other_background"
```

```
    tools:context=".CustomerOrderResultActivity">

    <ImageView
        android:id="@+id/customerResultImage"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="75dp"
        android:layout_width="300dp"
        android:layout_height="300dp"
        android:background="@drawable/ready_order_image" />

    <TextView
        android:id="@+id/customerResultTextView"
        android:layout_centerHorizontal="true"
        android:layout_width="300dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/customerResultImage"
        android:layout_marginTop="20dp"
        android:textAlignment="center"
        android:textStyle="bold"
        android:textColor="@color/black"
        android:textSize="30dp"
        android:text="Евакуатор на місці. Замовлення виконано. Деталі обговоріть із
водієм"/>

    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/customerResultButton"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="20dp"
        android:layout_width="270dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/customerResultTextView"
        android:background="@drawable/welcome_button_style"
        android:text="На головний екран"
        android:textAlignment="center"
        android:textColor="@color/white"
        android:textSize="22dp" />

</RelativeLayout>
```

Файл «activity_customer_registration»

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/customer_welcome_background"
    tools:context=".CustomerRegistrationActivity">

    <TextView
        android:id="@+id/registrationTextView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="235dp"
        android:text="РЕЄСТРАЦІЯ"
        android:textSize="45dp"
        android:textColor="#dd2e44" />
```

```
<TextView
    android:id="@+id/registrationTextView2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_below="@+id/registrationTextView1"
    android:layout_marginTop="5dp"
    android:text="Email-адреса"
    android:textSize="20dp"
    android:textColor="@color/black" />

<EditText
    android:id="@+id/registrationEmailCustomerEditText"
    android:layout_width="245dp"
    android:layout_height="50dp"
    android:layout_centerHorizontal="true"
    android:layout_below="@+id/registrationTextView2"
    android:layout_marginTop="5dp"
    android:background="@drawable/welcome_edittext_style"
    android:ems="10"
    android:hint="Електронна пошта"
    android:textAlignment="center"
    android:textColor="@color/black"
    android:inputType="textEmailAddress" />

<TextView
    android:id="@+id/registrationTextView3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_below="@+id/registrationEmailCustomerEditText"
    android:layout_marginTop="5dp"
    android:text="Пароль"
    android:textSize="20dp"
    android:textColor="@color/black" />

<EditText
    android:id="@+id/registrationPasswordCustomerEditText"
    android:layout_width="245dp"
    android:layout_height="50dp"
    android:layout_centerHorizontal="true"
    android:layout_below="@+id/registrationTextView3"
    android:layout_marginTop="5dp"
    android:background="@drawable/welcome_edittext_style"
    android:ems="10"
    android:hint="**********"
    android:textAlignment="center"
    android:textColor="@color/black"
    android:inputType="textPassword" />

<androidx.appcompat.widget.AppCompatButton
    android:id="@+id/registrationCustomerButton"
    android:layout_width="245dp"
    android:layout_height="wrap_content"
    android:layout_below="@+id/registrationPasswordCustomerEditText"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="10dp"
    android:background="@drawable/welcome_button_style"
    android:textAlignment="center"
```

```
        android:textColor="@color/white"
        android:text="Продовжити"
        android:textSize="20dp"/>

</RelativeLayout>
```

Файл «activity_customer_welcome»

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/customer_welcome_background"
    tools:context=".CustomerWelcomeActivity">

    <TextView
        android:id="@+id/welcomeTextView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="220dp"
        android:text="ПРИВІТ"
        android:textSize="55dp"
        android:textColor="@color/black" />

    <TextView
        android:id="@+id/welcomeTextView2"
        android:layout_width="300dp"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/welcomeTextView1"
        android:layout_marginTop="-6dp"
        android:text="Введіть email-адресу та пароль"
        android:textAlignment="center"
        android:textSize="19dp"
        android:textColor="@color/black"
        />

    <EditText
        android:id="@+id/editTextCustomerEmail"
        android:layout_width="245dp"
        android:layout_height="50dp"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/welcomeTextView2"
        android:layout_marginTop="8dp"
        android:background="@drawable/welcome_edittext_style"
        android:ems="10"
        android:hint="Електронна пошта"
        android:inputType="textEmailAddress"
        android:textAlignment="center"
        android:textColor="@color/black" />

    <EditText
        android:id="@+id/editTextCustomerPassword"
        android:layout_width="245dp"
        android:layout_height="48dp"
        android:layout_below="@+id/editTextCustomerEmail"
```

```
    android:layout_centerHorizontal="true"
    android:layout_marginTop="10dp"
    android:background="@drawable/welcome_edittext_style"
    android:ems="10"
    android:hint="**********"
    android:inputType="textPassword"
    android:textAlignment="center"
    android:textColor="@color/black" />

<RelativeLayout
    android:id="@+id/welcomeCustomerRelativeLayout"
    android:layout_centerHorizontal="true"
    android:layout_width="245dp"
    android:layout_height="wrap_content"
    android:layout_below="@+id/editTextCustomerPassword"
    android:layout_marginTop="10dp">

    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/signInCustomerButton"
        android:layout_width="115dp"
        android:layout_height="wrap_content"
        android:background="@drawable/welcome_button_style"
        android:textAlignment="center"
        android:textColor="@color/white"
        android:text="Увійти"
        android:textSize="17dp"/>

    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/roleBackCustomerButton"
        android:layout_width="115dp"
        android:layout_height="wrap_content"
        android:layout_marginLeft="15dp"
        android:layout_toEndOf="@+id/signInCustomerButton"
        android:background="@drawable/welcome_button_style"
        android:text="Назад"
        android:textAlignment="center"
        android:textColor="@color/white"
        android:textSize="17dp" />

</RelativeLayout>

<androidx.appcompat.widget.AppCompatButton
    android:id="@+id/signUpCustomerButton"
    android:layout_width="210dp"
    android:layout_height="wrap_content"
    android:layout_below="@+id/welcomeCustomerRelativeLayout"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="27dp"
    android:background="@android:color/transparent"
    android:text="Немає аккаунту"
    android:textAlignment="center"
    android:textColor="@color/black"
    android:textSize="16dp"
    android:textStyle="bold" />

<View
    android:id="@+id/view"
    android:layout_width="165dp"
    android:layout_height="3dp"
    android:background="#dd2e44"
```

```
        android:layout_below="@+id/signUpCustomerButton"
        android:layout_marginTop="-10dp"
        android:layout_centerHorizontal="true" />

</RelativeLayout>
```

Файл «activity_driver_info_registration»

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/driver_welcome_background"
    tools:context=".DriverInfoRegistrationActivity">

    <TextView
        android:id="@+id/registrationTextView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="195dp"
        android:text="Ваше ім'я"
        android:textSize="20dp"
        android:textColor="@color/black" />

    <EditText
        android:id="@+id/driverInfoName"
        android:layout_width="245dp"
        android:layout_height="50dp"
        android:layout_below="@+id/registrationTextView1"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="5dp"
        android:background="@drawable/welcome_edittext_style"
        android:ems="10"
        android:hint="Ім'я користувача"
        android:inputType="text"
        android:textAlignment="center"
        android:textColor="@color/black" />

    <TextView
        android:id="@+id/registrationTextView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/driverInfoName"
        android:layout_marginTop="5dp"
        android:text="Номер телефону"
        android:textSize="20dp"
        android:textColor="@color/black" />

    <EditText
        android:id="@+id/driverInfoPhone"
        android:layout_width="245dp"
        android:layout_height="50dp"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/registrationTextView2"
        android:layout_marginTop="5dp"
```

```
    android:background="@drawable/welcome_edittext_style"
    android:ems="10"
    android:hint="+380 (__) ___-__-__"
    android:inputType="phone"
    android:textAlignment="center"
    android:textColor="@color/black" />

<TextView
    android:id="@+id/registrationTextView3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_below="@+id/driverInfoPhone"
    android:layout_marginTop="5dp"
    android:text="Марка евакуатора"
    android:textSize="20dp"
    android:textColor="@color/black" />

<EditText
    android:id="@+id/driverInfoCarName"
    android:layout_width="245dp"
    android:layout_height="50dp"
    android:layout_centerHorizontal="true"
    android:layout_below="@+id/registrationTextView3"
    android:layout_marginTop="5dp"
    android:background="@drawable/welcome_edittext_style"
    android:ems="10"
    android:hint="Марка та модель авто"
    android:textAlignment="center"
    android:textColor="@color/black"
    android:inputType="text" />

<TextView
    android:id="@+id/registrationTextView4"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_below="@+id/driverInfoCarName"
    android:layout_marginTop="5dp"
    android:text="Номер евакуатора"
    android:textSize="20dp"
    android:textColor="@color/black" />

<EditText
    android:id="@+id/driverInfoCarNumber"
    android:layout_width="245dp"
    android:layout_height="50dp"
    android:layout_centerHorizontal="true"
    android:layout_below="@+id/registrationTextView4"
    android:layout_marginTop="5dp"
    android:background="@drawable/welcome_edittext_style"
    android:ems="10"
    android:hint="Номер авто"
    android:textAlignment="center"
    android:textColor="@color/black"
    android:inputType="text" />

<androidx.appcompat.widget.AppCompatButton
    android:id="@+id/driverCreateButton"
    android:layout_width="245dp"
```

```
        android:layout_height="wrap_content"
        android:layout_below="@+id/driverInfoCarNumber"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="10dp"
        android:background="@drawable/welcome_button_style"
        android:textAlignment="center"
        android:textColor="@color/white"
        android:text="Створити водія"
        android:textSize="20dp"/>

</RelativeLayout>
```

Файл «activity_driver_maps»

```
<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:map="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".DriverMapsActivity" >

    <fragment
        android:id="@+id/map"
        android:name="com.google.android.gms.maps.SupportMapFragment"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/calCurrentCustomerButton"
        android:layout_width="370dp"
        android:layout_height="60dp"
        android:layout_above="@+id/customerInfoRelativeLayout"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="10dp"
        android:background="@drawable/map_edittext_style"
        android:text="Подзвонити замовнику"
        android:textSize="20dp"
        android:textStyle="bold" />

    <RelativeLayout
        android:id="@+id/locationRelativeLayout"
        android:layout_width="300dp"
        android:layout_height="60dp"
        android:layout_marginLeft="15dp"
        android:layout_marginTop="15dp"
        android:background="@drawable/map_edittext_style">

        <TextView
            android:id="@+id/locationDriverTextView"
            android:layout_width="300dp"
            android:layout_height="wrap_content"
            android:layout_centerVertical="true"
            android:text="Приємного користування"
            android:textAlignment="center"
            android:textColor="@color/black"
            android:textSize="22dp"
            android:textStyle="bold" />
```

```
</RelativeLayout>

<androidx.appcompat.widget.AppCompatButton
    android:id="@+id/driverMenuButton"
    android:layout_width="370dp"
    android:layout_height="60dp"
    android:layout_alignParentBottom="true"
    android:layout_centerHorizontal="true"
    android:layout_marginBottom="35dp"
    android:background="@drawable/map_edittext_style"
    android:text="Меню водія"
    android:textSize="25dp"
    android:textStyle="bold" />

<RelativeLayout
    android:id="@+id/customerInfoRelativeLayout"
    android:layout_width="370dp"
    android:layout_height="100dp"
    android:layout_above="@+id/driverMenuButton"
    android:layout_centerHorizontal="true"
    android:layout_marginBottom="10dp"
    android:background="@drawable/map_edittext_style">

    <de.hdodenhof.circleimageview.CircleImageView
        android:id="@+id/customerImage"
        android:layout_width="80dp"
        android:layout_height="80dp"
        android:layout_centerVertical="true"
        android:layout_marginLeft="10dp"
        android:src="@drawable/avatar_customer_image"
        map:civ_border_color="#dd2e44"
        map:civ_border_width="3dp" />

    <TextView
        android:id="@+id/customerMapsName"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="10dp"
        android:layout_marginTop="21dp"
        android:layout_toEndOf="@+id/customerImage"
        android:text="Username"
        android:textColor="@color/black"
        android:textSize="18dp" />

    <TextView
        android:id="@+id/customerMapsPhoneNumber"
        android:layout_width="140dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/customerMapsName"
        android:layout_marginLeft="10dp"
        android:layout_marginTop="7dp"
        android:layout_toEndOf="@+id/customerImage"
        android:text="380000000000"
        android:textColor="@color/black"
        android:textSize="18dp" />

    <View
        android:id="@+id/view"
        android:layout_width="5dp"
        android:layout_height="65dp"
```

```
            android:layout_centerVertical="true"
            android:layout_toEndOf="@+id/customerMapsPhoneNumber"
            android:background="#dd2e44" />

        <TextView
            android:id="@+id/customerMapsCarInfo"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="10dp"
            android:layout_marginTop="21dp"
            android:layout_toEndOf="@+id/view"
            android:text="Car Info"
            android:textColor="@color/black"
            android:textSize="18dp" />

        <TextView
            android:id="@+id/customerMapsCarNumber"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/customerMapsCarInfo"
            android:layout_marginLeft="10dp"
            android:layout_marginTop="7dp"
            android:layout_toEndOf="@+id/view"
            android:text="Car Number"
            android:textColor="@color/black"
            android:textSize="18dp" />

    </RelativeLayout>

</RelativeLayout>
```

Файл «activity_driver_menu»

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".DriverMenuActivity">

    <com.google.android.material.appbar.AppBarLayout
        android:id="@+id/appBarDriverMenu"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >

        <androidx.appcompat.widget.Toolbar
            android:id="@+id/toolBarDriverMenu"
            android:background="#dd2e44"
            android:layout_width="match_parent"
            android:layout_height="60dp">

            <RelativeLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content">

                <ImageView
                    android:id="@+id/closeDriverMenuButton"
                    android:layout_width="40dp"
```

```
                    android:layout_height="50dp"
                    android:src="@drawable/cancel_info_button"
                    app:tint="@color/white" />

                <ImageView
                    android:id="@+id/saveDriverMenuButton"
                    android:layout_width="40dp"
                    android:layout_height="50dp"
                    android:src="@drawable/save_info_button"
                    android:layout_alignParentEnd="true"
                    android:layout_marginRight="20dp"
                    app:tint="@color/white"/>
            </RelativeLayout>

        </androidx.appcompat.widget.Toolbar>

    </com.google.android.material.appbar.AppBarLayout>

    <RelativeLayout
        android:id="@+id/menuRelativeLayout1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/appBarDriverMenu">

        <de.hdodenhof.circleimageview.CircleImageView
xmlns:app="http://schemas.android.com/apk/res-auto"
            android:id="@+id/avatarImage"
            android:layout_width="200dp"
            android:layout_height="250dp"
            android:layout_marginLeft="20dp"
            android:src="@drawable/avatar_driver_image"
            app:civ_border_color="#dd2e44"
            app:civ_border_width="4dp" />

        <EditText
            android:id="@+id/menuDriverName"
            android:layout_width="150dp"
            android:layout_height="40dp"
            android:layout_marginLeft="10dp"
            android:layout_marginTop="35dp"
            android:layout_toEndOf="@+id/avatarImage"
            android:textSize="17dp" />

        <EditText
            android:id="@+id/menuDriverPhone"
            android:layout_width="150dp"
            android:layout_height="40dp"
            android:layout_below="@id/menuDriverName"
            android:layout_marginLeft="10dp"
            android:layout_toEndOf="@+id/avatarImage"
            android:textSize="17dp"
            android:inputType="phone"/>

        <EditText
            android:id="@+id/menuDriverCarInfo"
            android:layout_width="150dp"
            android:layout_height="40dp"
            android:layout_toEndOf="@+id/avatarImage"
            android:layout_below="@id/menuDriverPhone"
            android:layout_marginLeft="10dp"
```

```
            android:textSize="17dp" />

        <EditText
            android:id="@+id/menuDriverCarNumber"
            android:layout_width="150dp"
            android:layout_height="49dp"
            android:layout_below="@id/menuDriverCarInfo"
            android:layout_marginLeft="10dp"
            android:layout_toEndOf="@+id/avatarImage"
            android:textSize="17dp" />

    </RelativeLayout>

    <RelativeLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/menuRelativeLayout1"
        android:layout_marginTop="20dp"
        android:layout_centerHorizontal="true">

        <RelativeLayout
            android:id="@+id/coordinatesRelativeLayout"
            android:layout_width="220dp"
            android:layout_height="180dp"
            android:background="@drawable/map_edittext_style">

            <TextView
                android:id="@+id/menuTextview1"
                android:layout_width="200dp"
                android:layout_height="wrap_content"
                android:layout_centerHorizontal="true"
                android:layout_marginTop="10dp"
                android:text="Поточні координати"
                android:textAlignment="center"
                android:textColor="@color/black"
                android:textSize="22dp"
                android:textStyle="bold" />

            <View
                android:id="@+id/view"
                android:layout_width="165dp"
                android:layout_height="3dp"
                android:layout_below="@+id/menuTextview1"
                android:layout_centerHorizontal="true"
                android:layout_marginTop="3dp"
                android:background="#dd2e44" />

            <TextView
                android:id="@+id/menuTextview2"
                android:layout_width="100dp"
                android:layout_height="wrap_content"
                android:layout_below="@+id/view"
                android:layout_marginLeft="15dp"
                android:layout_marginTop="15dp"
                android:text="Широта"
                android:textAlignment="center"
                android:textColor="@color/black"
                android:textSize="20dp"
                android:textStyle="bold" />
```

```
    <TextView
        android:id="@+id/latitudeDriverInfo"
        android:layout_width="90dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/view"
        android:layout_marginTop="15dp"
        android:layout_toEndOf="@+id/menuTextview2"
        android:text="000"
        android:textAlignment="center"
        android:textColor="#dd2e44"
        android:textSize="20dp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/menuTextview3"
        android:layout_width="100dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/menuTextview2"
        android:layout_marginLeft="15dp"
        android:layout_marginTop="10dp"
        android:text="Довгота"
        android:textAlignment="center"
        android:textColor="@color/black"
        android:textSize="20dp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/longitudeDriverInfo"
        android:layout_width="90dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/latitudeDriverInfo"
        android:layout_marginTop="10dp"
        android:layout_toEndOf="@+id/menuTextview3"
        android:text="000"
        android:textAlignment="center"
        android:textColor="#dd2e44"
        android:textSize="20dp"
        android:textStyle="bold" />
</RelativeLayout>

<RelativeLayout
    android:id="@+id/aboutDriverButton"
    android:layout_width="220dp"
    android:layout_height="65dp"
    android:layout_below="@id/coordinatesRelativeLayout"
    android:layout_marginTop="20dp"
    android:background="@drawable/map_edittext_style">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:text="Про додаток"
        android:textAlignment="center"
        android:textColor="@color/black"
        android:textSize="22dp"
        android:textStyle="bold" />

</RelativeLayout>
```

```
<RelativeLayout
    android:id="@+id/deleteDriverButton"
    android:layout_width="220dp"
    android:layout_height="65dp"
    android:background="@drawable/map_edittext_style"
    android:layout_below="@id/aboutDriverButton"
    android:layout_marginTop="20dp">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true"
        android:text="Видалити аккаунт"
        android:textAlignment="center"
        android:textColor="@color/black"
        android:textSize="22dp"
        android:textStyle="bold" />
</RelativeLayout>

<RelativeLayout
    android:id="@+id/siteDriverButton"
    android:layout_width="130dp"
    android:layout_height="180dp"
    android:layout_marginLeft="20dp"
    android:layout_toEndOf="@+id/coordinatesRelativeLayout"
    android:background="@drawable/map_edittext_style">

    <ImageView
        android:id="@+id/siteImage"
        android:layout_width="110dp"
        android:layout_height="120dp"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="20dp"
        android:src="@drawable/menu_site_button" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/siteImage"
        android:layout_centerHorizontal="true"
        android:text="Сайт додатку"
        android:textAlignment="center"
        android:textColor="@color/black"
        android:textSize="16dp"
        android:textStyle="bold" />

</RelativeLayout>

<RelativeLayout
    android:id="@+id/exitDriverButton"
    android:layout_marginTop="20dp"
    android:layout_width="130dp"
    android:layout_height="150dp"
    android:background="@drawable/map_edittext_style"
    android:layout_below="@+id/siteDriverButton"
    android:layout_toEndOf="@+id/coordinatesRelativeLayout"
    android:layout_marginLeft="20dp">

    <ImageView
```

```
                android:id="@+id/exitImage"
                android:layout_width="110dp"
                android:layout_height="150dp"
                android:layout_centerInParent="true"
                android:src="@drawable/menu_exit_button" />

        </RelativeLayout>
    </RelativeLayout>
</RelativeLayout>
```

Файл «activity_driver_order_result»

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/other_background"
    tools:context=".DriverOrderResultActivity">

    <ImageView
        android:id="@+id/customerResultImage"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="75dp"
        android:layout_width="300dp"
        android:layout_height="300dp"
        android:background="@drawable/ready_order_image" />

    <TextView
        android:id="@+id/customerResultTextView"
        android:layout_centerHorizontal="true"
        android:layout_width="300dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/customerResultImage"
        android:layout_marginTop="20dp"
        android:textAlignment="center"
        android:textStyle="bold"
        android:textColor="@color/black"
        android:textSize="30dp"
        android:text="Ви на місці. Замовлення виконано. Деталі обговоріть із
клієнтом"/>

    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/driverResultButton"
        android:layout_width="270dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/customerResultTextView"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="20dp"
        android:background="@drawable/welcome_button_style"
        android:text="На головний екран"
        android:textAlignment="center"
        android:textColor="@color/white"
        android:textSize="22dp" />

</RelativeLayout>
```

Файл «activity_driver_registration»

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/driver_welcome_background"
    tools:context=".DriverRegistrationActivity">

    <TextView
        android:id="@+id/registrationTextView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="235dp"
        android:text="РЕЄСТРАЦІЯ"
        android:textSize="45dp"
        android:textColor="#dd2e44" />

    <TextView
        android:id="@+id/registrationTextView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/registrationTextView1"
        android:layout_marginTop="5dp"
        android:text="Email-адреса"
        android:textSize="20dp"
        android:textColor="@color/black" />

    <EditText
        android:id="@+id/registrationEmailDriverEditText"
        android:layout_width="245dp"
        android:layout_height="50dp"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/registrationTextView2"
        android:layout_marginTop="5dp"
        android:background="@drawable/welcome_edittext_style"
        android:ems="10"
        android:hint="Електронна пошта"
        android:textAlignment="center"
        android:textColor="@color/black"
        android:inputType="textEmailAddress" />

    <TextView
        android:id="@+id/registrationTextView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/registrationEmailDriverEditText"
        android:layout_marginTop="5dp"
        android:text="Пароль"
        android:textSize="20dp"
        android:textColor="@color/black" />
```

```
    <EditText
        android:id="@+id/registrationPasswordDriverEditText"
        android:layout_width="245dp"
        android:layout_height="50dp"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/registrationTextView3"
        android:layout_marginTop="5dp"
        android:background="@drawable/welcome_edittext_style"
        android:ems="10"
        android:hint="**********"
        android:textAlignment="center"
        android:textColor="@color/black"
        android:inputType="textPassword" />

    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/registrationDriverButton"
        android:layout_width="245dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/registrationPasswordDriverEditText"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="10dp"
        android:background="@drawable/welcome_button_style"
        android:textAlignment="center"
        android:textColor="@color/white"
        android:text="Продовжити"
        android:textSize="20dp"/>

</RelativeLayout>
```

Файл «activity_driver_welcome»

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/driver_welcome_background"
    tools:context=".DriverWelcomeActivity">

    <TextView
        android:id="@+id/welcomeTextView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="220dp"
        android:text="ПРИВІТ"
        android:textSize="55dp"
        android:textColor="@color/black" />

    <TextView
        android:id="@+id/welcomeTextView2"
        android:layout_width="300dp"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/welcomeTextView1"
        android:layout_marginTop="-6dp"
        android:text="Введіть email-адресу та пароль"
        android:textAlignment="center"
```

```
    android:textSize="19dp"
    android:textColor="@color/black"
    />

<EditText
    android:id="@+id/editTextDriverEmail"
    android:layout_width="245dp"
    android:layout_height="50dp"
    android:layout_centerHorizontal="true"
    android:layout_below="@+id/welcomeTextView2"
    android:layout_marginTop="8dp"
    android:background="@drawable/welcome_edittext_style"
    android:ems="10"
    android:hint="Електронна пошта"
    android:inputType="textEmailAddress"
    android:textAlignment="center"
    android:textColor="@color/black" />

<EditText
    android:id="@+id/editTextDriverPassword"
    android:layout_width="245dp"
    android:layout_height="48dp"
    android:layout_below="@+id/editTextDriverEmail"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="10dp"
    android:background="@drawable/welcome_edittext_style"
    android:ems="10"
    android:hint="**********"
    android:inputType="textPassword"
    android:textAlignment="center"
    android:textColor="@color/black" />

<RelativeLayout
    android:id="@+id/welcomeDriverRelativeLayout"
    android:layout_centerHorizontal="true"
    android:layout_width="245dp"
    android:layout_height="wrap_content"
    android:layout_below="@+id/editTextDriverPassword"
    android:layout_marginTop="10dp">

<androidx.appcompat.widget.AppCompatButton
    android:id="@+id/signInDriverButton"
    android:layout_width="115dp"
    android:layout_height="wrap_content"
    android:background="@drawable/welcome_button_style"
    android:textAlignment="center"
    android:textColor="@color/white"
    android:text="Увійти"
    android:textSize="17dp"/>

<androidx.appcompat.widget.AppCompatButton
    android:id="@+id/roleBackDriverButton"
    android:layout_width="115dp"
    android:layout_height="wrap_content"
    android:layout_toEndOf="@+id/signInDriverButton"
    android:layout_marginLeft="15dp"
    android:background="@drawable/welcome_button_style"
    android:textAlignment="center"
    android:textColor="@color/white"
    android:text="Назад"
```

```
        android:textSize="17dp"/>

    </RelativeLayout>

    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/signUpDriverButton"
        android:layout_width="210dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/welcomeDriverRelativeLayout"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="27dp"
        android:background="@android:color/transparent"
        android:text="Немає аккаунту"
        android:textAlignment="center"
        android:textColor="@color/black"
        android:textSize="16dp"
        android:textStyle="bold" />

    <View
        android:id="@+id/view"
        android:layout_width="165dp"
        android:layout_height="3dp"
        android:background="#dd2e44"
        android:layout_below="@+id/signUpDriverButton"
        android:layout_marginTop="-10dp"
        android:layout_centerHorizontal="true" />

</RelativeLayout>
```

Файл «activity_main»

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/start"
    tools:context=".MainActivity">


</androidx.constraintlayout.widget.ConstraintLayout>
```

Файл «activity_role_selection»

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/welcome"
    tools:context=".RoleSelectionActivity">

    <TextView
        android:id="@+id/roleSelectionTextView1"
        android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="220dp"
        android:text="Вибір ролі"
        android:textSize="45dp"
        android:textColor="#dd2e44" />

    <TextView
        android:id="@+id/roleSelectionTextView2"
        android:layout_width="350dp"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/roleSelectionTextView1"
        android:layout_marginTop="1dp"
        android:text="Оберіть роль для продовження авторизації"
        android:textAlignment="center"
        android:textSize="20dp"
        android:textColor="@color/black" />

    <RelativeLayout
        android:id="@+id/roleSelectionRelativeLayout"
        android:layout_width="wrap_content"
        android:layout_height="200dp"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/roleSelectionTextView2"
        android:layout_marginTop="12dp">

        <ImageView
            android:id="@+id/roleSelectionCustomerButton"
            android:layout_width="160dp"
            android:layout_height="wrap_content"
            android:background="@drawable/client_role"
            />

        <ImageView
            android:id="@+id/roleSelectionDriverButton"
            android:layout_width="160dp"
            android:layout_height="wrap_content"
            android:layout_toEndOf="@+id/roleSelectionCustomerButton"
            android:layout_marginLeft="10dp"
            android:background="@drawable/driver_role"/>
    </RelativeLayout>

</RelativeLayout>
```

Файл «activity_site»

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/other_background"
    tools:context=".SiteActivity">

    <ImageView
        android:id="@+id/siteImage"
        android:layout_centerHorizontal="true"
```

```
        android:layout_marginTop="75dp"
        android:layout_width="330dp"
        android:layout_height="330dp"
        android:background="@drawable/site_error_image" />

    <TextView
        android:id="@+id/siteTextView"
        android:layout_centerHorizontal="true"
        android:layout_width="300dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/siteImage"
        android:layout_marginTop="10dp"
        android:textAlignment="center"
        android:textStyle="bold"
        android:textColor="@color/black"
        android:textSize="30dp"
        android:text="Наразі, на жаль, сайт знаходиться в розробці та не працює"/>

    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/siteBackMenuButton"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="20dp"
        android:layout_width="150dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/siteTextView"
        android:background="@drawable/welcome_button_style"
        android:text="Назад"
        android:textAlignment="center"
        android:textColor="@color/white"
        android:textSize="22dp" />

</RelativeLayout>
```