# 

# **Вступ**

У наш час ідея дуже популярною є ідея шерінгу. Щодня все менше і менше людей схильні купувати щось нове. Люди схильні користуватись чимось в дуже короткий термін, а потім позбавлятися цього за непотрібністю. Це відбувається через те, що молодь любить спробувати щось нове, або використовувати якійсь пристрої, які потрібні дуже рідко. І головна проблема таких змін – це дороговизна цих пристроїв.

На реалізацію однієї з таких проблем і направлений даний програмний продукт. Цей проект покликаний максимально спростити і доступ до одного з таких дорогих пристроїв – дронів.

Критерії успіху даної системи базуються на тому, що система зможе допомогти користувачу уникнути покупки такого дорогого пристрою як дрон.

Основною бізнес-метою є реалізація робочого сервісу, який дозволить зберігати персональний профіль користувача зі всією інформацією у хмарі та зробити цей сервіс крос-платформеним.

Сервіс повинен мати багаторівневу архітектуру, бути багатошаровим і масштабуватися, на клієнтському рівні мати веб-клієнт і мобільний додаток, а також розумний пристрій.

Таким чином, метою курсової роботи є розробка веб сервісу, що приймає запити від користувачів про оренду дрону, відображає на загальній мапі найближчих дронів та надає доступ до керування дрону користувачу Також робота націлена на дослідження принципів та інструментів для розробки програм з використанням API для взаємодії з даними.

**1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ**

* 1. Аналіз області

У сучасному процесі глобалізації все більшу роль грають орендні агентства. У зв'язку з цим усе більш актуальним стає питання автоматизації обслуговування клієнтів таких фірм.

Абсолютних аналогів даної системи немає. Одними з часткових аналогів є «Getmancar», «Mobilecar», але це сервіси оренди автомобілей за допомогою мобільного пристрою (див. рис. 1.1).

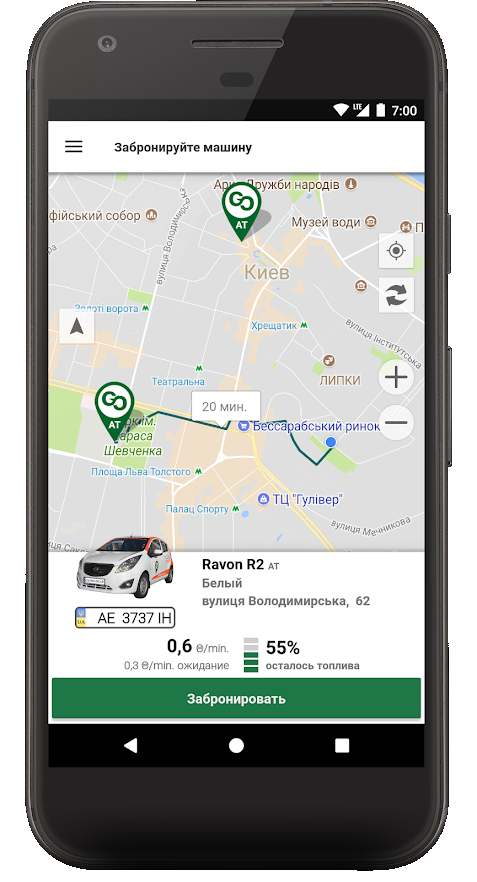


Рисунок 1.1 – Додаток «Getmancar» на мобільному пристрої

Основна проблема цієї предметної області полягає у величезному об'ємі інформації (інформація про клієнтів, коптерів, телеметрія з коптерів), що зберігається, і необхідності швидкого пошуку необхідної її частини. Зберігання її в паперовому виді вимагає великого простору і повністю позбавляє можливості швидкого доступу. Також проблемою є те, що інформація ділиться на окремі модулі: на інформацію про коптери, клієнтів та платіжні чеки. При необхідності отримання інформації про копт ери і про клієнтів які орендували цей коптер довелося б працювати з двома масивами різнотипної інформації [1]. Вочевидь, що для зберігання тисяч коптерів, клієнтів, чеків і т.д. необхідна інформаційна система, яка буде зберігати велику кількість даних та представляти швидкий та зручний доступ до них. Щоб спростити роботу користувача і надати дійсно потрібні йому функції, система повинна буде використовувати базу даних і надавати зручне відображення даних про коптери; швидкий пошук, фільтрацію і сортування потрібної інформації; отримання найбільш потрібної статистики і можливість формування чеків для друку.

Вимоги до програми:

а) можливість додавання нових коптерів, редагування існуючих (зміна технічних характеристик, фірми) і видалення;

б) фільтрація і пошук коптерів по назві, вартості;

в) можливість отримання інформації про коптер;

г) оформлення оренди при узгодженням з клієнтом;

д) оформлення чека до виконаної оренди;

е) отримання електронного листа з чеком.

Виходячи з вимог, можна скласти список об’єктів, інформацію по яких необхідно зберігати в базі даних інформаційної системи. Об'єктами в цій системі будуть: тур, чек, туристичний ваучер.

Фірми: ID номер у базі; назва; код країни;

Чеки: ID номер у базі; номер оренди, дата створення, загальна сума.

Знижки: ID номер у базі; тип знижки; кількість у процентах;

Коптери: ID номер у базі, статус, широта, довгота, вартість за хвилину, ID фірми, максимальна швидкість, максимальна висота польоту, тип управління, тип коптеру.

Клієнти: ID номер у базі; Прізвище; Ім’я; Пароль; Стать; Дата народження; Паспортні данні; Номер телефону; Email; Знижка.

Чеки: ID номер у базі; ID путівки; ID клієнту; кількість замовлених путівок; загальна вартість; дата продажу;

Країни: код країни; назва.

Оренди: ID номер у базі; ID номер клієнта; ID номер коптера; дата початку.

* 1. Опис функціональної структури системи

В умовах великої конкуренції в області оренд транспортних засобів чи електронних пристроїв одну з головних ролей грає швидкість реєстрації, пошуку й оформлення оренди. Для вирішення даного завдання програмний продукт надасть користувачу можливість швидкої оренди коптеру за допомогою зручного інтерфейсу користувача. Для клієнтів, які постійно користуються послугами агентств з питань оренди електронних пристроїв, гостро стоїть питання про фінансові витрати на оренди пристроїв. Для цього система буде автоматично відстежувати постійних клієнтів і вносити їх до списку клієнтів, яким надається знижка. Щоб прискорити і автоматизувати роботу працівника агентства з оренди, даний програмний продукт буде автоматично робити розсилку повідомлень щодо завершення оренди і отримання чеку.

Таким чином, дана інформаційна система буде значно збільшувати швидкість роботи агентства за рахунок значної автоматизації рутинної роботи.

* 1. Опис інформаційних потреб користувачів

На основі аналізу предметної області можна виділити наступні інформаційні потреби користувачів в даній системі.

Для клієнтів агентств:

а) вхід в програму;

б) перегляд інформації про коптери;

в) перегляд статистики, щодо зайнятих та вільних коптерів;

г) оренда коптерів;

д) оформлення чеку.

Для робітників агентств:

а) всі потреби клієнтів;

б) зміна інформації у коптерах;

в) додавання даних в систему;

г) видалення даних з системи;

д) перегляд інформації про клієнтів.

* 1. Опис існуючого документообігу в ПЗ

Документообіг предметної області «Агентство з оренди коптерів»

складається з наступних документів:

а) чек, який містить інформації про коптер та клієнта.

1.5 Опис обмежень цілісності

У даній галузі присутні такі обмеження щодо ідентифікації об'єктів:

а) кожна коптер має свою унікальну назву;

б) кожен клієнт має унікальний ID;

в) кожний чек має унікальний ID;

г) кожна країна має унікальний код;

д) кожна знижка має унікальний ID;

е) кожний фірма має унікальний ID;

ж) кожна оренда має унікальний ID.

Також даної предметної області притаманні такі обмеження щодо відносин між об'єктами:

а) кожен клієнт має одну чи не однієї оренди;

б) напрям в фірма може мати одну, багато, чи не одного коптерів;

в) кожен клієнт має одну, чи не однієї знижки;

г) одна оренда має один чек;

д) одна оренда має один коптер;

ж) одна фірма має одну країну.

**2 ПОСТАНОВКА ЗАДАЧИ**

Метою курсової роботи є реалізація робочого сервісу для оренди дронів різного типу і спрощення рутинної праці шляхом використання автоматизації процесу. Сервіс потрібен реалізовувати основні функції, які задані аналізом предметної області. Крім того, він повинен мати багаторівневу архітектуру, бути багатошаровим і мати можливість масштабуватися; на клієнтському рівні повинен бути веб-клієнт і мобільний додаток.

2.1 Основний функціонал системи

а) введення, зберігання, видалення і редагування інформації про коптери;

б) пошук, сортування (за ім’ям) та фільтрація коптерів, користувачів та фірм серед загальної кількості;

в) введення, зберігання, видалення і редагування інформації про фірми;

г) відображення на загальній мапі обраного коптера;

д) автоматичне знаходження найближчої зарядної станції для коптера;

е) відсилання телеметрії з коптера (координати);

ж) автоматичне оформлення чеку, який містить основну інформацію про коптер і клієнта і який можна роздрукувати;

и) автоматичне відправлення чеку на електрону адресу клієнта при завершенні оренди;

2.2 Допущення та залежності

Для коректної роботи усіх частин проекту наведений список допущень:

* користувачі системи мають пристрій з GPS системою;
* користувачі системи мають пристрій з доступом до Інтернету;
* користувачі зацікавлені в короткотривалій оренді дронів.

Також існують наступні залежності:

* додаток буде орієнтуватися на версії Android 4.0 і вище;
* додаток буде орієнтуватися на новітні версії браузерів (IE 9+, Chrome 45+, Safari 4.0+).

2.3 Релізи

Розробка даного програмного продукту була поділена на декілька релізів.

Необхідний функціонал для першого релізу:

а) реєстрація користувача ( з веб-сайту);

б) авторизація в системі (з веб-сайту);

в) можливість додавання нових коптерів, брендів (з веб-сайту);

г) можливість пошуку коптера або бренда за назвою (з веб-сайту);

д) можливість перегляду дронів на зашальній мапі (з мобільного додатку та веб-сайту);

ж) можливість перегляду основної інформації про коптер або бренд (з мобільного додатку та веб-сайту);

з) можливість замовити оренду коптера (з мобільного додатку та веб-сайту).

к) можливість отримувати електронний лист при завершенні оренди (з мобільного додатку та веб-сайту);

Реалізація наступних релізів:

л) можливість зареєструватися (з мобільного додатку);

м) можливість авторизуватися (з мобільного додатку);

п) можливість додавати нові коптери (з мобільного додатку);

р) можливість аналізувати телеметрію з коптуру (веб-сайт).

2.4 Користувацькі обмеження

Для коректного використання даного продукту та викреслення нерозуміння з боку користувачів, є деякі користувацькі обмеження:

а) користувач повинен пам’ятати логін та пароль;

б) користувач повинен авторизуватися для користування системою.

2.5 Бізнес - потреби та пріоритетність

Даний продукт може залучити користувачів та нових розробників своєю унікальністю та актуальною функціональністю (таблиця 2.1). Ця таблиця демонстує основних зацікавлених осіб та їх інтереси у даному проекті.

Таблиця 2.1 – Особи та їх інтереси

|  |  |  |
| --- | --- | --- |
| Інтереси | Обмеження | Зацікавлена ​​особа |
| Управління персональним орендованим дроном. | Бюджет | Користувач |
| Розробка масштабується. Система, що відповідає встановленим критеріям якості. | Терміни реалізації та бюджет | Розробник |
| Своєчасне отримання готового продукту з документацією. | Нема | Власник продукту |

Час і бюджет - головні обмеження розробника, таким чином потрібно відокремити пріоритетні показники для того, щоб акцентувати свою увагу на них в процесі розробки і бути більш стійким до стресових ситуацій, пов'язаних з нестачею ресурсів.

2.6. Середа оточення

Так як предметною областю даного веб-орієнтованого сервісу не передбачено використання певної СУБД для зберігання і роботи з інформацією системи була обрана СУБД Microsoft SQL Server 2017.

Даний веб-орієнтований сервіс буде розподілений на наступні части:

а)  серверна частина – платформа Microsoft .Net Core 2.2, мова програмування C#;

б) клієнтська частина – платформа Angular 7, мова програмування TypeScript, мова розмітки HTML 5, мова стилю CSS 3;

в) мобільна частина – платформа Android, мова програмування Kotlin;

г) емулятор IoT пристрою – платформа Microsoft .Net Core 2.2, мова програмування C#, розміщення Microsoft Azure Hub.

Сервіс буде розроблена в середовищі Microsoft Visual Studio 2017 [2].

У зв'язку з вибором засобів розробки, виникають обмеження з вибором операційної системи для роботи з системою. Користувачі повинні мати комп'ютер під управлінням ОС Microsoft Windows XP і вище, браузер (IE 9+, Chrome 45+, Safari 4.0+), мобільний додаток із встановленою версією ОС Android не менш ніж 4.0.0. Для функціонування емулятору або серверною частиною користувач не повинен встановлювати додаткове програмне забезпечення, тому що ці елементи розгорнуті у платформі хмарних обчислень Microsoft Azure.

1. **АРХІТЕКТУРА ПРОЕКТУ**

3.1 Архітектура

Архітектура клієнт-сервер (див. рис. 3.1) є одним із архітектурних шаблонів програмного забезпечення та є домінуючою концепцією у створенні розподілених мережних застосунків і передбачає взаємодію та обмін даними між ними. Вона передбачає такі основні компоненти:

* набір серверів, які надають інформацію або інші послуги програмам, які звертаються до них;
* набір клієнтів, які використовують сервіси, що надаються серверами;
* мережа, яка забезпечує взаємодію між клієнтами та серверами

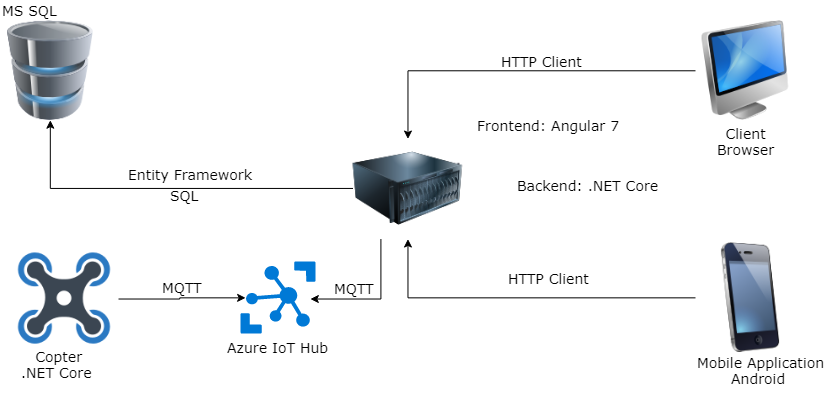


Рисунок 3.1 – Архітектура проекта

На рисунку 3.1 можна побачити взаємодію між різними компонентами системи. Так, для отримання результату з веб-браузеру користувача, front-end посилає запит на back-end сервер. Back-end сервер в свою чергу валідує дані та звертається до бази даних, далі в зворотньому напрямку компоненти передають результат до користувача.

Для більш детального аналізу архітектури, побудуємо діаграму розсортування (див. рис. 3.2)

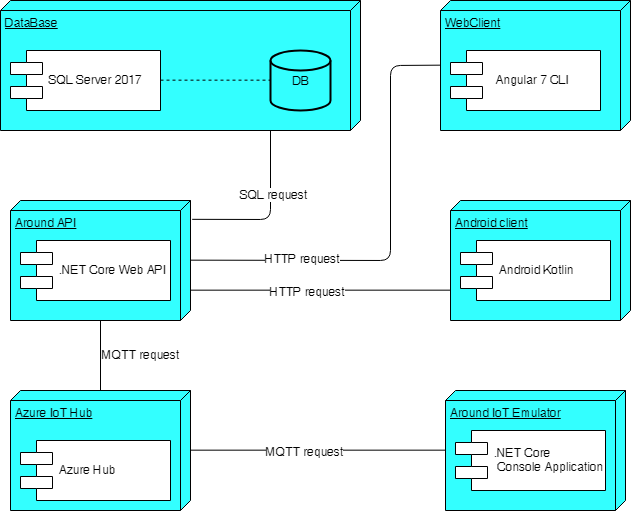


Рисунок 3.2 – Діграма розсортування

При роботі з базою даних було використано реляційну базу даних MSSQL та SQL Server 2017.

В якості клієнта може використовувати як web-сайт, так і мобільний додаток (Android) для виконання оренди коптеру, що включає запит на замовлення. Користувач для роботи з системою повинен авторизуватися. Цей процес включає хешування, за допомогою інтегрованою функції мови C#, паролю клієнта, направлення запиту на пошук користувача у базі. У тому разі якщо користувач існує в базі, йому видається унікальний JWT токен, який має час життя (5 хвилин) і який прикріплюється до кожного запиту, щоб сервер був впевнений, що запит робить авторизований користувач.

При роботі з коптером використовується спеціальна захищена черга у Azure IoT Hub та спеціальний протокол спілкування MQTT.

3.2 Будування UML діаграми

У даній системі існує три виду акторів – «Guest» – неавторизований користувач, авторизований користувач – «Authorized client» і адміністратор – «Administrator». Неавторизований користувач має можливість авторизуватись або зареєструватися у системі. Авторизований користувач має всі можливості ролі «Неавторизований користувач». Крім того він може продивлятися інформацію про коптери; отримувати статистику про зайняті коптери; знаходити коптери за допомогою фільтрів; отримувати знижку; продивлятися місцезнаходження коптеру на загальній мапі; замовляти оренду коптеру; отримувати чек. Останній актор – адміністратор має можливості ролі «Авторизований користувач» крім того, може редагувати коптери та інші сутності,

На основі цього складемо USE CASE діаграму запропонованої системи (див рис. 3.2).

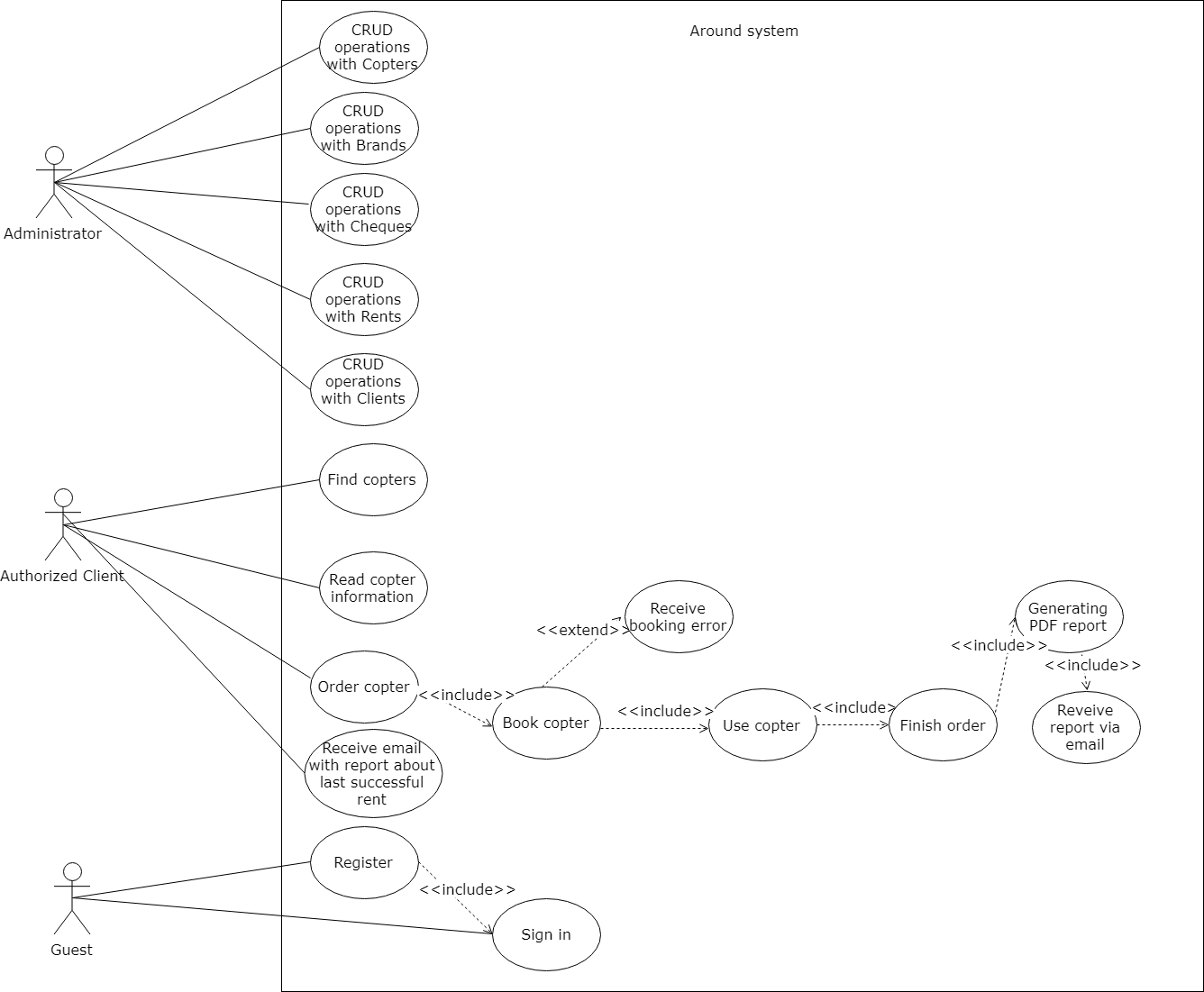


Рисунок 3.2 – UseCase діаграма

3.3 Побудова ER-діаграми

Побудова ER-діаграми є одним з важливих етапів проектування бази даних. Вона дозволяє схематично побачити модель майбутньої бази даних, яка буде побудована на основі цієї діаграми. На основі розробленої концептуальної моделі можна визначити таблиці, з котрих складатиметься база даних. Сутностями є: клієнт, коптер, чек і оренда. Виходячи з функціонала, який має надавати інформаційна система на етапі аналізу предметної області було отримано список даних, які необхідно зберігати в базі даних. Об'єднавши суті з даними можна побудувати модель бази даних у вигляді ER-діаграми, яка демонструє ставлення і зв'язку між таблицями в реляційної базі даних [3].

Етапи побудови ER-діаграми:

а) на основі концептуальної моделі визначені сутності, які будуть присутні у базі даних;

б) визначено відносини між отриманими сутностями (1:1; 1: М; М: М);

в) на основі інформації про об'єкти даної інформаційної системи, отриманих на етапі аналізу предметної області, визначаємо атрибути для даних про сутності;

г) проаналізувавши дані, що зберігаються визначено, що деякі поля таблиць можуть мати значення NULL. У таблиці «Clients» зберігаються дані про клієнтів агентства з оренди техніки. Кожен клієнт має право не повідомляти, контактний номер телефону, але тоді клієнт може не отримати інформацію щодо знижок.

В результаті ER-моделювання отримана діаграма, що демонструє схему бази даних і відносини між таблицями в базі даних (див. рис 3.3).

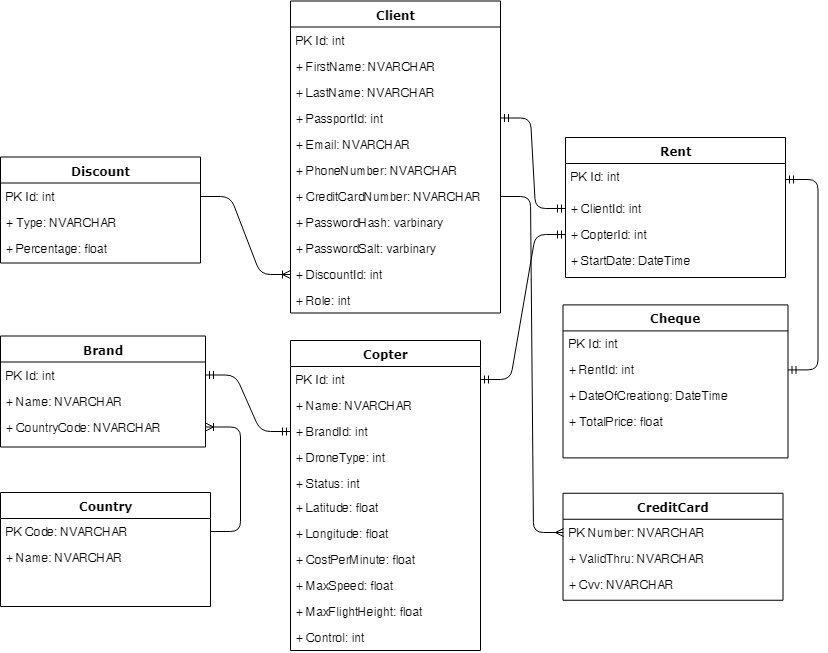


Рисунок 3.3 – ER діаграма бази даних

На основі цієї інформації можна приступати до створення файлу бази даних [4].

3.4. Побудова схеми реляційної бази даних в третій нормальної формі

На основі побудованої ER-діаграм побудуємо схема реляційної бази даних. Всі сутності, які були в концептуальній моделі та ER-діаграмі, становлять таблиці бази даних.. У результаті отримаємо схему бази даних (див. рис 3.3).

Необхідно довести, що база даних знаходиться у 3НФ. Перш ніж приступити до отримання таблиць в 3НФ наведемо визначення [5].

Відношення знаходиться у третій нормальній формі тоді і тільки тоді, коли воно знаходиться в другій нормальній формі і жоден неключовий атрибут не є транзитивно залежним від її первинного ключа. Транзитивна залежність — це залежність між неключовими ат­рибутами.

Отже переглянемо таблицю Clients(Клієнти). Ключом у неї є поле Id. При дальнішому розгляді можна зрозуміти, що всі наступні поля, а саме: FirstName (Ім’я), LastName (Прізвище), Sex (Стать), DateOfBorn (Дата народження), Passport(Паспортні дані), PhoneNumber (Номер телефону), Email (Електронная адресса), DiscountId(Код знижки), UserLevel (Рівень туриста) залежать лише від поля Id, тобто не мають транзитивну залежну. Як результат, виконується основна умова приведення до третьої нормальної форми.

Наступна таблиця Discount (Знижки). Ключом у неї є поле Id. Поля Type (Тип знижки), Percantages(Кількість у процентах) є незалежними між собою. Тобто всі поля таблиці Знижки залежні тількі від ключа. Отже, виконується основна умова 3НФ.

Таблиця Cheque (Платіжний чек) має ключ Id. Розглянемо інші поля: RentId (Код оренди), DateOfCreation (Дата створення), TotalPrice (Загальная вартість). При аналізі зрозуміло, що жодне з полів цієї таблиці не має залежності від не ключових атрибутів, тобто таблиця знаходиться у 3НФ.

Проаналізуємо таблицю Countries (Країни). Кючом цієї таблиці є поле Code. Єдине поле , окрім ключа, є поле Country отже таблиця автоматично знаходиться у третій нормальній формі.

Розглянемо таблицю Copters (Коптери). Ключом є поле Id. Розглянемо наступні поля Name (Ім’я), Status (Статус), Latitude (Широта), Longitude(Довгота), CostPerMinute (Вартість за хвилину), BrandId (Номер фірми), MaxSpeed (Максимальна швидкість), MaxHeight (Максимальна висота), Control (Тип управління), DroneType (Тип коптера). При розгляданні можна сказати, що всі поля залежать лише від ключа таблиці, тобто виконується основна умова 3НФ.

Роглянемо таблицю Rent (Оренди). Ключем цієї таблиці є поле Id. Проаналізуємо поля: ClientId (Номер клієнту), CopterId (номер коптеру), StartTime(Дата початку). Можна сказати, що жоден не ключовий атрибут не має транзитивної залежності. Отже Таблиця знаходиться у 3НФ.

Проаналізуємо таблицю Brands. Ключом таблиці є поле Id. Проаналізуємо інші поля Name (Назва) та CountryCode (Код країни). Можна сказати, що жоден не ключовий атрибут не має транзитивної залежності. Отже таблиця автоматично знаходиться у 3НФ.

Отже, якщо кожна з таблиць бази даних знаходиться у третій нормальній формі, то і вся таблиця знаходиться у третій нормальній формі. В результаті приведення таблиць бази даних до 3НФ була доведена правильність побудованої моделі бази даних на етапі побудови ER-діаграми. Таким чином, отримана точна модель бази даних, і можна приступати до створення файлу бази даних.

# **4 Кодування**

## 4.1 Back-end сервер

У якості платформи була обрана платформа .NET Core 2.2 мови розробки бекенду було обрано C#. Завдяки даному вибору розробка API була швидкою, а код вийшов компактним і відмінно читаним. Крім того, має можливість дуже шкидко й гнучко масштабуватись. Серверна частина побудована за патерном Factory. У якості IDE був обраний Visual Studio, оскільки він повністю задовольняє мої потреби як розробника і є найкращим в своїй сфері.

Сервер системи створювався на платформі .NET Core 2.2.. Була створена база даних на основі підходу Code First, а також були написані контролери. Для доступу до даних використовувався Entity Framework Core.

Крім того весь код побудований з використанням патерну Dependency Injection (паттерн в якому використовується принцип низької залежності компонентів один від одного).

Для прикладу наведемо код програмної реалізації моделі «Коптер»:

public class Copter

{

public int Id { get; set; }

public string Name { get; set; }

public Status Status { get; set; }

public double Latitude { get; set; }

public double Longitude { get; set; }

public double CostPerMinute { get; set; }

public int BrandId { get; set; }

public double MaxSpeed { get; set; }

Продовження лістингу коду:

public double MaxFlightHeight { get; set; }

public Control Control { get; set; }

public DroneType DroneType { get; set; }

public virtual Brand Brand { get; set; }

public virtual List<Rent> Rents { get; set; }

}

Даний код буде автоматично оброблений і на основі нього буде створено таблицю в базі даних. Аби це сталося необхідно спочатку створити та реалізувати клас DronesharingContext (додати усі сутності, які ми хочемо бачити у базі даних). Далі потрібно створити клас DesignTimeDbContextFactory і реалізувати інтерфейс IDesignTimeDbContextFactory<DronesharingContext>:

public class DesignTimeDbContextFactory : IDesignTimeDbContextFactory<DronesharingContext>

{

public DronesharingContext CreateDbContext(string[] args)

{

IConfigurationRoot configuration = new ConfigurationBuilder()

.SetBasePath(Directory.GetCurrentDirectory())

.AddJsonFile("appsettings.json")

.Build();

var builder = new DbContextOptionsBuilder<DronesharingContext>();

var connectionString = configuration.GetConnectionString("DefaultConnection");

builder.UseSqlServer(connectionString);

return new DronesharingContext(builder.Options);

}

}

Також наведемо приклад коду RentController,а саме код замовлення оренди коптеру:

[HttpPost]

[Route("create")]

public async Task<IActionResult> CreateRent([FromBody] RentAggregate rentDto)

{

var copter = \_copterRepository.Get(rentDto.CopterId).Result;

if (copter.Status != Status.Ordered)

{

var rent = Rent.CreateFromDto(rentDto);

rent = await \_rentRepository.Create(rent);

await \_hub.StartUsingCopter(rent);

\_copterRepository.UpdateStatus(rentDto.CopterId);

return Ok("Success");

}

return BadRequest();

}

В даному прикладі користувач відправляє на сервер свій ідентифікатор та ідентифікатор коптеру, який він хоче орендувати. Після цього система перевіряє, чи не зайнятий коптер, якщо дрон вільний, то йому відсилається запит, щоб заблокувати його від спроб інших користувачів забронювати цей коптер. Теперь користувач може використовувати дрон за своїм поглядом. Для тестування різних видів запитів я використовував дві утілити – Swagger UI та Postman.

Однією з головних особливостей серверної частини є функція генерацію PDF звіту щодо завершеної оренди. Коли користувач завершує оренду через веб-клієнт або мобільний додаток, сервер генерує спеціальний документ – чек, який користувач отримає на свою електронну скриньку, яка була вказана при реєстрації.

4.2 Front-end клієнт

Веб-додаток було написано за допомогою технології Angular 7 та мови програмування TypeScript. Я обрал платформу Angular через те, що це проект двох прогресивних корпорацій – Google та Microsoft. У останній час зв’язка .NET Core і Angular стала дуже міцною. Це можна пояснити сильною активністю двох великих ком’юніті – розробників .NET та Angular. Кількість issue(пропозицій) на офіційних GitHub репозиторіях обох фірм зростається кожного дня. Це говорить про те, що йде активна модифікація обох платформ. Саме це сприяло моєму вибору цієї міцної зв’язки. В якості середи розробки виступала Visual Studio Code.

Наведемо приклад коду – сервіс, що відповідає за лотримання даних з серверної частини та розпакування ії у потрібні елементи для роботи на стороні клієнту:

@Injectable()

export class CopterService {

constructor(private http: HttpClient) { }

baseUrl = 'http://localhost:55555/api/Copter/';

getCopters() {

return this.http.get<CopterDto[]>(this.baseUrl);

}

getCopterById(id: number) {

return this.http.get<CopterDto>(this.baseUrl + '/' + id);

}

createCopter(copter: CopterAggregate) {

return this.http.post(this.baseUrl, copter,

{

headers: new HttpHeaders({ 'Content-Type': 'application/json' }),

responseType: 'text'

})

.subscribe(data => console.log('Works!'));

}

Продовження лістингу коду:

updateCopter(copter: Copter) {

return this.http.put(this.baseUrl + '/' + copter.id,

copter);

}

deleteCopter(id: number) {

return this.http.delete(this.baseUrl + '/' + id);

}

}

Цей код демонструє CRUD методи для сутності Copter. Як можна побачити при виконанні POST запиту ми обов’язково повинні виконати функцію subscribe, якщо вона не буде викликана, то сервер не отримає необхідну модель і не зможе обробити запит.

Для методів, що потребують від користувача бути авторизованим, створений спеціальний клас-провайдер – TokenInterceptor, що перехоплює вихідні HTTP запити, та прописує їм у заголовку Authorization значення JWT токену авторизації користувача.

Одна з найважливіших особливостей цього проекту – це інтеграція із Google API. Завдяки Google Maps клієнт може подивитися, де зараз знаходиться коптер , який він хоче орендувати (див. рис. 4.1)

Можна побачити, що візуальна частина побудована із використанням бібліотеки стилів Angular Material. Ця бібліотека стилів є рекомендованою для Angular проектів, тому що не потребує великих змін у файлах стилів.

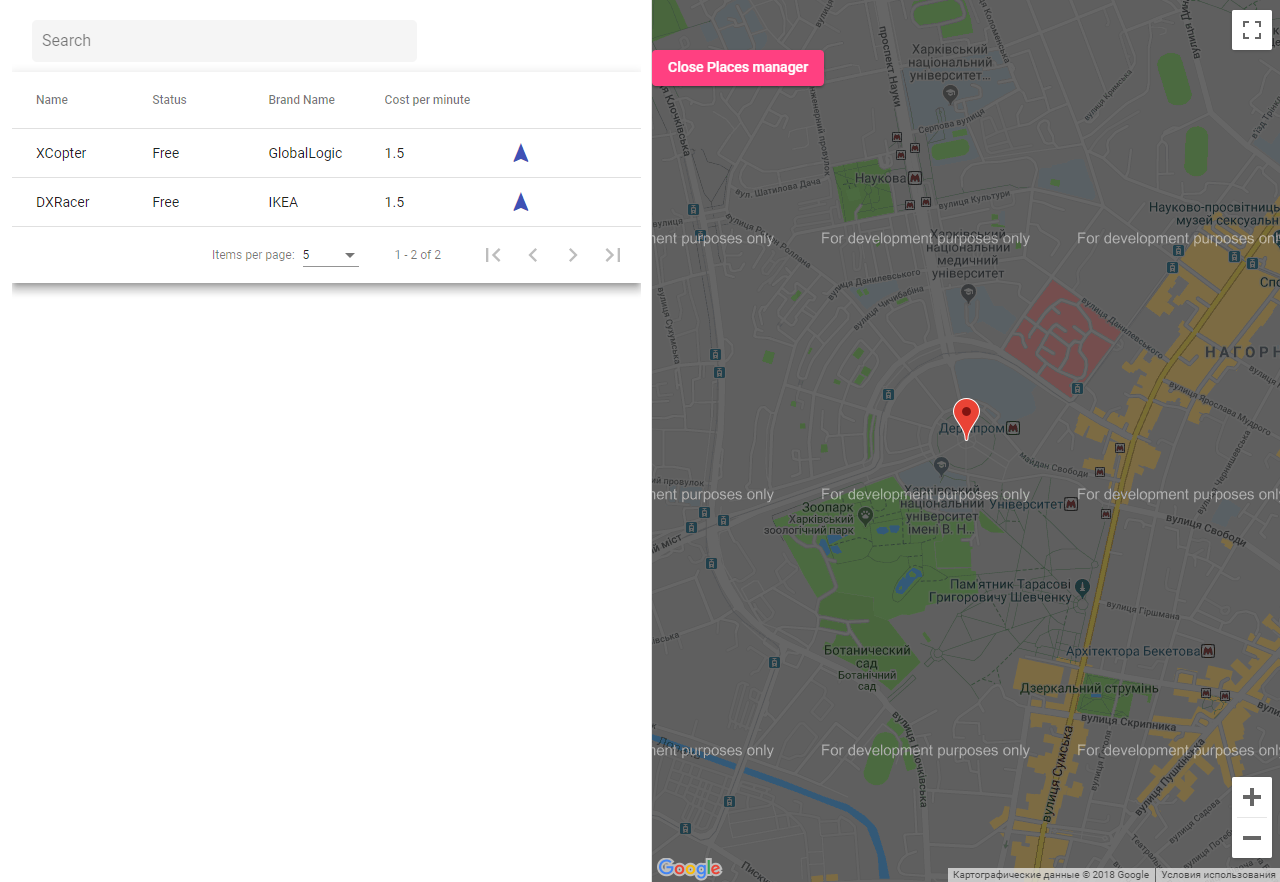


Рисунок 4.1 – Відібраження коптерів на загальній мапі

Таблиця у лівій частині екрана дозволяє користувачеві шукати потрібний коптер за ім’ям, дивитися, де він на мапі, а поті натискати кнопку оренди.

4.3 Мобільний клієнт

Мобільний додаток біло створено за допомогою мови Kotlin, Android API та IDE Android Studio, тобто, мобільний додаток є нативним додатком для платформи Android, і, таким чином, використовує всі відповідні переваги нативної розробки.

Для HTTP запитів до сервера було використано бібліотеку Retrofit2. Увесь код звернення до методів REST API сервера було інкапсульовано у сінглтон об’єкти. На рисунку наведен приклад списку доступних коптерів (див. рис. 4.2).

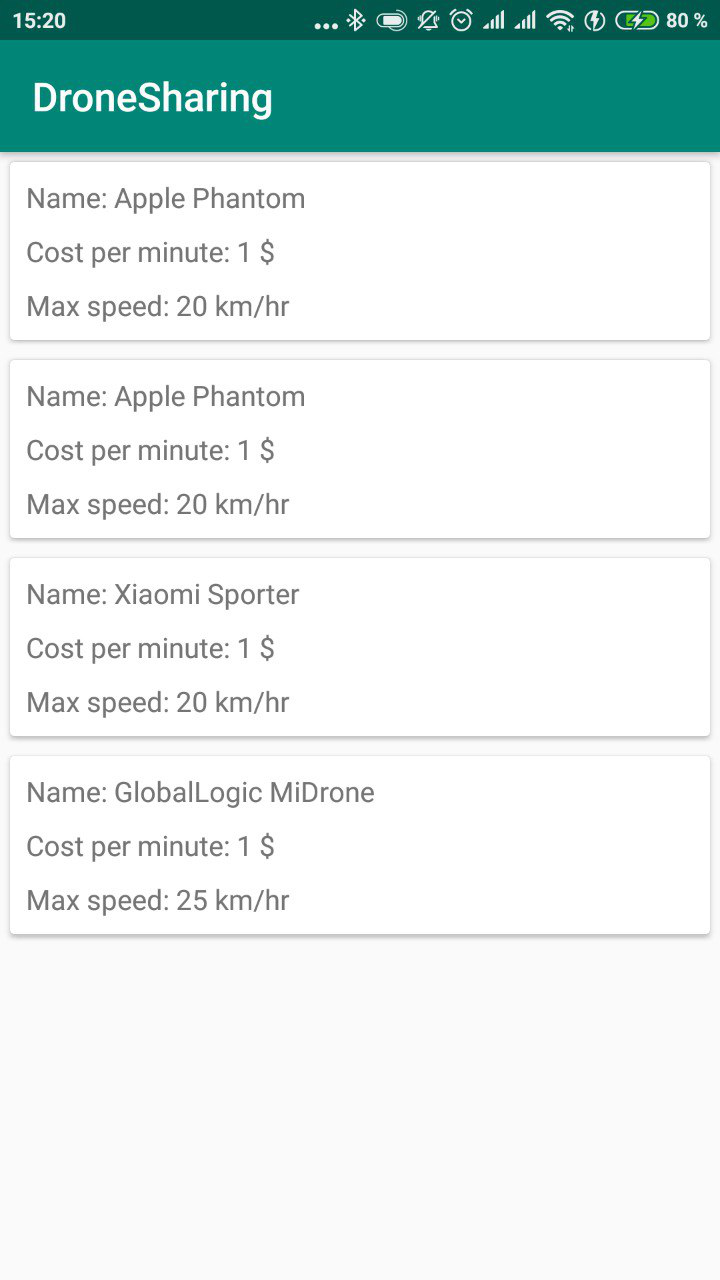


Рисунок 4.2 – Мобільний додаток

На елементи можна натиснути, щоб перейти до наступної частини – початку оренди коптеру

4.4 Розумний пристрій

В даному курсовому проекті розумний пристрій реалізований як емулятор IoT пристрою. Для демонстрації роботи мого розумного пристрою я хочу навести діаграму послідовностей (див. рив. 4.3)

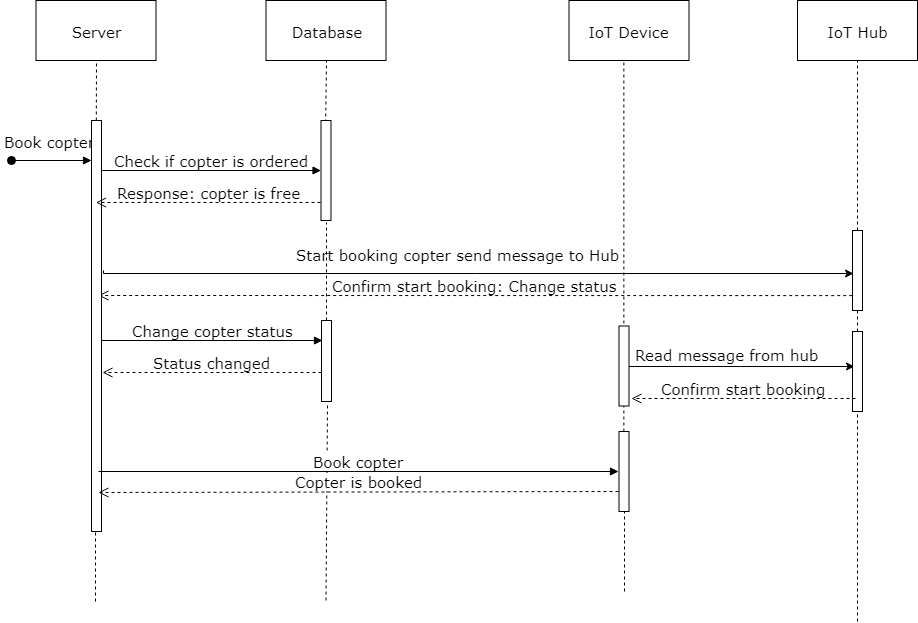
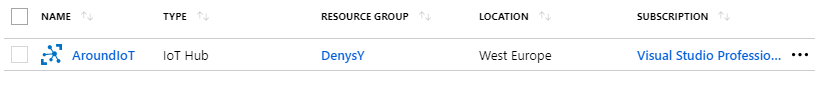


Рисунок 4.3 – Діаграма послідовностей умного приладу

Як можна побачити, у процесі замовлення оренди для коптера приймають участь 4 елемента: сервер, база даних, IoT Hub, IoT Device. Спочатку на сервер приходить запит на оренду коптера, після цього з сервера йде запрос у базу даних на перевірку, чи вільний це коптер. Якщо він вільний, то сервер відправляє повідомлення в IoT Hub, що необхідно орендувати коптер із заданим ідентифікатором. З моменту запуску коптер кожні 5 секунд (параметр задається програмно) вичитує повідомлення з абу. З того моменту, як коптер знайшов повідомлення про оренду, він стає доступним для замовника.

Спочатку я створив .NET Core Console Application для отримання запросу з серверу. Потім я створив і зареєстрував новий пристрій в Azure IoT Hub. Тепер цей консольний додаток є повним IoT пристроєм, який спілкується із сервером через спеціальний протокол MQTT.(див. рис. 4.4)



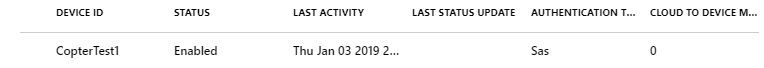


Рисунок 4.4 – Доска Azure Portal

На рисунку ми бачимо, що пристрій CopterTest1 є пристроєм ресурсу AroundIoT. Завдяки спеціальній властивості IoTHubSecret серверна частина може відправляти повідомлення у чергу на Azure. У той сами час ми знаємо, що якщо повідомлення будло відправлено в чергу, воно там буде зберігатися 30 днів (настройка за замовченням). Тепер коптер може вичитати повідомлення із абу як тільки буде онлайн.

# **ВИСНОВКИ**

В результаті роботи було розроблено веб-орієнтований сервіс для оренди коптерів. Вона складається з чотирьох частин: серверу, сайту, мобільного додатка та smart-пристрою.

Розробка велась з використанням платформи .NET Core Core 2.2 з використанням мови програмування C# для серверної частини програмної системи. Сайт було розроблено з використанням HTML5, CSS3, Angular7, стилів Angular Material та мови TypeScript. Для розробки мобільного додатку використовувалася середовище розробки Android Studio та мова програмування Kotlin. Логіка роботи емулятору smart-девайса була написана мовою програмування C# та побудовано на платформі Azure за допомогою Azure IoT Hub. В якості СКБД було обрано MS SQL Server 2017, а в якості технології доступу до даних – Entity Framework Core.

Розроблений веб-орієнтований сервіс є лише прототипом та не може в поточному стані використовуватися в реальних умовах. Перше, що необхідно доробити – замінити smart-пристрій, що емулює роботу дрона, на реального дрона, та створити відповідне програмне забезпечення для нього. Незважаючи на це, розвиток системи є досить перспективним, тому що відкриває можливість співпраці з різними компаніями, вдосконалення якості роботи дрона шляхом збору спеціальних метрік – телеметрії інтеграції аналізу даних завдяки базі Cassandra, збільшення швидкості збору інформації завдяки ElasticSearch та зростання гнучкості сервісу завдяки мікросервісній архітектурі.. Сервіс дозволяє виконувати базові функції такі, як авторизація, реєстрація, пошук коптерів, отримання електронного листа на поштову скриньку, маніпуляції з дронами у базі даних.

# **ПЕРЕЛІК ПОСИЛАНЬ**

1. Use Case – Wikipedia [Електронний ресурс] – Режим доступу до ресурсу: <https://en.wikipedia.org/wiki/Use_case> (дата звернення: 15.11.2018)
2. Боггс М. UML и Rational Rose / М. Боггс. – Москва: РГГУ, 2016. – 438 с.
3. Бен-Ган И. Microsoft SQL Server 2012. Основы T-SQL / Ицик Бен-Ган. – Москва: Эксмо, 2015. – 400 с.
4. ADO.Net Entity Framework [Електронний ресурс] – Режим доступу до ресурсу: https://ru.wikipedia.org/wiki/ADO.NET\_Entity\_Framework (дата звернення: 17.11.2018).
5. Entity Framework 6 Recipes / [Z. Hirani, L. Tenny, N. Gupta та ін.]. – New York: Apress, 2013. – 548 с.
6. Uluca D. Angular 6 for Enterprise-Ready Web Applications / Doguhan Uluca. – Бірмінгем: Packt, 2018. – 512 с.
7. Leiva A. Kotlin for Android Developers: Learn Kotlin the easy way while developing an Android App / Antonio Leiva., 2016. – 232 с.
8. Horton J. Android Programming for Beginners: Learn all the Java and Android skills you need to start making powerful mobile applications / John Horton. – Бірмінгем: Packt, 2015. – 698 с.

**ДОДАТОК А**

Електронні документи

Таблиця А.1 – Електронні документи

|  |  |  |
| --- | --- | --- |
| Зміст | Папка | Ім'я файлу |
| Пояснювальна записка до курсової роботи |  | course.doc |
| Програма, що виконується (усі файли із модулями, що виконуються, а також усі необхідні бібліотеки), а також файл setup.exe, якщо його наявність передбачена  технічним завданням | Around |  |
| Докладний проект програми з кодами і  поясненнями у середовищі проектування програмного продукту | Around |  |

# **ДОДАТОК Б**

# Специфікація ПЗ

Vision and Scope Document

for

Web-oriented service for drone rent “Around”

Version 1.0 approved

Prepared by Denys Yerusalimtsev

NURE

11.10.2018

Table of Contents

1. Business Requirements 1

1.1. Background 1

1.2. Business Opportunity 1

1.3. Business Objectives and Success Criteria 1

1.4. Customer or Market Needs 1

1.5. Business Risks 1

2. Vision of the Solution 2

2.1. Vision Statement 2

2.2. Major Features 2

2.3. Assumptions and Dependencies 2

3. Scope and Limitations 2

3.1. Scope of Initial Release 2

3.2. Scope of Subsequent Releases 2

3.3. Limitations and Exclusions 3

4. Business Context 3

4.1. Operating Environment 4

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Business Requirements

## Background

Recently, there are more and more large, profitable web and mobile projects. Stability and availability of the application and the site is an integral part of the image of any company. Convenience and quick access will increase the number of users hundreds and thousands of times. It is important for users to know how much money they spend on rental services. In addition, the problem areas for renting a device for any period were queues, searches for specialty shops, etc. If user needs to rent a device, especially a short-term, users will no longer need to search for specialty shops and stand in long queues, all functionality is transferred to a mobile device or website. in different directions. This product can be used for such types of rentals such as: rent of a special shooting equipment, rental of a cargo vehicle, rental of a vehicle. Thus, this product contributes to the spread of modern type of equipment, a copter (drone), which resolves many of problems, such as shooting photos or video materials in different types of terrain, the possibility of fast and convenient transportation using the minimum amount of human resources. Using copters as a means of transport allows you to reduce traffic on the roads and to load the airspace of cities in the right amount. In addition, the power supply of drones allows you to save the environment. Thus this product solves problems.

## Business Opportunity

There are many companies, which have similar functionality, but the main object of those projects are cars. This project remasters opinions about drones. Below table presents examples of similar projects and their costs.

|  |  |  |
| --- | --- | --- |
| Компания | Rate per min. | Rate waiting per min. |
| YouDrive | 2-3 hrn | 50коп – 1 hrn |
| Делимобиль | 3 hrn | 1 hrn |
| Car5 | 2.50 – 3 hrn | 1.50 hrn. |
| Anytime | 2 – 2.75 hrn | 50 hrn. |
| Belkacar | 3 hrn. | 0.3 – 0.5 hrn. |

As can be seen from the table, the rental of a land vehicle is not always profitable. Due to the rise in price of fuel, the price for services of this type of transport continues to grow. While the use of transport with an electric motor is becoming increasingly beneficial both economically and ecologically. The advantage of copters as a mode of transport will be low cost of fuel (electricity) and mobility (the client will not overpay for being in a “traffic jam” as a result, the speed of movement increases). For the first time profit of this project will be low or will be absent. So adding advertising blocks into first release application will covered first payment wastes. After that, 80 – 90% of profit will come from drone renting.

## Business Objectives and Success Criteria

The goal of this project is to create a network of sharing drones across the country, and later expand its influence to other regions.

BO-1: This product is a service provided free-to-use model. To use the services you must have a bank card. The cost of services may vary depending on the type of leased drone.

BO-2: Attracting 100,000 regular users in 1 year after the first release.

BO-3: Achieving cash flow in the product in the amount of 500,000 hryvnias per month in 1 year after the first release.

BO-4: Expand the diversity of drones in first year after release.

BO-5: Reduce the amortization loss of drones in the first 6 months after release by adding ad units.

*SC-1: 10 thousand downloads in the first month after release*

*SC-2: 100 thousand users in first 6 months after release.*

*SC-3: 15 thousand regular users in first 6 months after the release.*

## Customer or Market Needs

*The target audience of this project can be divided into several categories. Each of them has specific needs that this product must meet.*

* *Car drivers*

This category of users will increasingly use drones as a modern means of transportation. Easy management and mobility are the main characteristics of devices that are appreciated by this audience. The target audience of this project can be divided into several categories. Each of them has specific needs that this product must include.

* Cargo delivery services

Courier companies spend a lot of money on human resources and fuel (besides, there are frequent injuries). Using drones in the field of courier delivery or shipments will reduce the number of loaders and couriers. In addition, spending on fuel and the so-called "human factor" will be reduced.

* Amator video and photo shooting

Professionals in video and photo shooting have their own drones, so this service will be relevant for beginners. In order not to spend extra money on the purchase of an expensive drone, the user can try out this craft using the rental one.

## Business Risks

RI-1: Lack of time for developing full functional (Difficulty = 0.7 Action: reduction in the amount to the functional working version)

RI-2: The problem with the initial number of drones (Difficulty = 0.7, Measures: finding an investor, reducing the initial number of drones, reducing the types of drones in the first release after release)

RI-3: Too few users will use the service, which may lead to the initial unprofitability of the product (Difficulty = 0.9, Measures: conducting promotions on social networks and other sites.

# Vision of the Solution

## Vision Statement

This system will be adapted to different types of operating systems - cross-platformint. The goal of the project is to provide quick and easy access to the drones for the purpose of viewing and renting copters 24/7. Constant profit will be provided by around the clock working, Regular online support helps user with different problems. It will be possible to view the status of the drones on the map and see the nearest free drones. The service was created to improve the process of leasing (especially short-term) of devices by more convenient search of devices. For convenient search this application has integration with Google API such as Google Maps. So user can search the nearest drone by using maps. Client also can filter displayed drones by category and read short description of every copter. After selecting specific copter user can go into rent mode and book this drone for 20 minutes for free. This time is needed for getting to place, where drone located. After quick inspection user can turn on rent function and he will see timer, which calculate rent time. If user decide to stop renting, he can use application to turning off the rent mode. This product provides cashless kind of payment. After payment maked user will received generated PDF report, which consists of main rent information: user info, drone info, rent time and total cost.

## Major Features

FE-1: Filtering drones by category.

FE-2: Search drones with a map.

FE-3: Renting mode.

FE-4 Automatically generating a PDF report.

FE-5: Sending a pre-generated report to user`s mail.

FE-6: Integration with the Google API.

FE-7: Bank payment function

## Assumptions and Dependencies

AS-1: Start-up capital is required, which will cover the need to purchase the first batch of drones.

AS-2: The connection between the mobile application and the database will be via an HTTP request.

AS-3: To work with the application, the client needs a smartphone or computer.

*DE-1: The number of users depends on whether they are satisfied with the quality of the provided drones*

*DE-2: Availability of a system that will ensure the security of payments.*

# Scope and Limitations

## Scope of Initial Release

Front end

This application is built on Angular 6 platform. The main programming language is TypeScript. Web site is representing of mobile application for browsers. The main functionality of front-end part is using Google API, such as Google Maps, Web site also provides basic validation for adding, editing and deleting drones. Site also provides convenient timer, which demonstrate, how much time has passed.

Back end

Back end part of this application is built on ,NET Core 2.1 platform. The main programming language is C# .The main advantages of this platform is cross platforming. This means that this application can be executed in different operation systems. This project can also be dockerized by Docker systems. That means, that this project will be scalable and reliable. This application will built on 3-tier architecture, and second tier – business logic will include different functions. One of the main functions is connecting to database and making different operations with entities. Registration and authorization are also provided in this application. Authorization to this service will be provided with using Bearer token or BaseAuth. So unlogged user will not see drone`s or user`s information. Besides, system provides role management. There are two roles in service: admin and user. Admin can manage base information of drones, ban users. User can see information about drones, rent them, receive cheques, pay for service. Web site will be translated into 3 languages: English, Ukrainian, Russian. Special service will create PDF reports for comfortable representing of information. This system also provides email receiving of pre-generated report.

Mobile

Mobile application is built on Android system. The main programming language is Java. One of the main functions is connecting to database and making different operations with entities. Registration and authorization are also provided in this application. Authorization to this service will be provided with using Bearer token or BaseAuth. So unlogged user will not see drone`s or user`s information. Besides, system provides role management. There are two roles in service: admin and user. Admin can manage base information of drones, ban users. User can see information about drones, rent them, receive cheques, pay for service. Mobile application will be translated into 3 languages: English, Ukrainian, Russian. Special service will create PDF reports for comfortable representing of information. This system also provides email receiving of pre-generated report.

IoT

IoT in this project plays a secure role. Small IoT device will provide access to particular drone after user turned on rent mode on mobile application or web site. Stranger person can`t gave access to drone without server verification on back end side.

|  |  |  |
| --- | --- | --- |
| Feature | Web version | Mobile version |
| FE-1 | Fully implemented. | Not implemented |
| FE-2 | Fully implemented. | Fully implemented. |
| FE-3 | Fully implemented. | Fully implemented. |
| FE-4 | Fully implemented. | Not implemented |
| FE-5 | Fully implemented. | Fully implemented. |
| FE-6 | Fully implemented. | Fully implemented. |
| FE-7 | Fully implemented. | Fully implemented. |

## 

## Scope of Subsequent Releases

Release 2

|  |  |  |
| --- | --- | --- |
| Feature | Web version | Mobile version |
| FE-1 | See prev. release | Fully implemented. |
| FE-2 | See prev. release | Fully implemented. |
| FE-3 | See prev. release | *See prev. release* |
| FE-4 | See prev. release | *See prev. release* |
| FE-5 | See prev. release | *See prev. release* |
| FE-6 | See prev. release | *See prev. release* |
| FE-7 | See prev. release | *See prev. release* |

## Limitations and Exclusions

*LI-1: The application will work on phones with Android version 4 +*

*LI-2: Users must be over 18 years old.*

*LI-3: Users must provide a photocopy of their document.*

*LI-4: Payment is cashless.*

# Business Context

## Operating Environment

It is expected that users will be from different parts of the world.

The database will be stored on the IIS server, later migration to Amazon S3 or Microsoft Azure cloud technologies is possible. The number of servers of this platform is large and they are dispersed throughout the world. Amazon / Microsoft Azure is responsible for the bandwidth of the site. Data will be available to users without interruption.

Angular 7 platform - frontend and .NET Core 2.2 platform and Microsoft SQL Server 2017 - backend were selected for website development for this service. For the IoT device-emulator Azure IoT Hub and NET Core 2.2 platform have been chosen. For the development of a mobile application, the Android platform and the Kotlin language have been chosen.