2

Міністерство освіти та науки України

Національний технічний університет України «КПІ» ім. Ігоря Сікорського

Факультет інформатики та обчислювальної техніки

Кафедра технічної кібернетики

**Курсовий робота**

з дисципліни «Проектування мікропроцесорних систем»

на тему

«Іграшка «Радіокеруємий автомобіль»»

|  |  |
| --- | --- |
| **Керівник:** доц. Долина В. Г.  «Захист дозволено»  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2019 р.  Захищено з оцінкою  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **/** підпис керівника / | **Виконав:**  студент IV курсу  групи ЗПІ-зп61  Макіян С.А (№ ЗПІ-зп6115) |

**Київ – 2019**

**МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**

**ІМЕНІ ІГОРЯ СІКОРСЬКОГО»**

**ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ**

**КАФЕДРА АВТОМАТИКИ ТА УПРАВЛІННЯ У ТЕХНІЧНИХ СИСТЕМАХ**

Дисципліна Проектування мікропроцесорних систем

СпеціальністьІнженерія програмного забезпечення

Курс 4 Група ЗПІ-зп61 Семестр V

**ЗАВДАННЯ**

На курсовий проект (роботу) студента

Макіяна Смбата Артуровича

1. **Тема проекту (роботи):** Іграшка «Радіокеруємий автомобіль»
2. **Термін здачі студентом завершеного проекту (роботи) :** 16.01.2019.
3. **Вхідні дані до проекту (роботи):** c++*,Java*
4. **Зміст розрахунково–пояснювальної записки (перелік питань, які підлягають розробці):**
   1. *Постановка задачі;*
   2. *Ієрархія задач, що вирішуються;*
   3. *Аналіз функціоналу;*
   4. *Інструкція користувачу;*
   5. *Інструкція програмісту;*
   6. *Графічні матеріали.*
5. **Дата отримання завдання:** 02.11.2018.

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Назва етапу курсового проекту (роботи) | Строк виконання етапів проекту (роботи) | Примітки |
| 1 | Отримання завдання для курсового проект | 02.11.2018 |  |
| 2 | Вивчення предметної області | 10.11.2018 |  |
| 3 | Проектування архітектури модуля | 20.11.2018 |  |
| 4 | Розробка програмного продукту | 10.12.2018 |  |
| 5 | Проектування та розробка інтерфейсу | 20.12.2018 |  |
| 6 | Налагодження та тестування ПП | 01.01.2019 |  |
| 7 | Виправлення виявлених помилок та недоліків | 5.01.2019 |  |
| 8 | Оформлення документації | 25.05.2018 |  |
| 9 | Здача курсового проекту | 07.06.2018 |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Студент** |  |  | Макіян Смбат Артурович |
|  | (підпис) |  | (прізвище, ім’я, по батькові) |

|  |  |  |  |
| --- | --- | --- | --- |
| **Керівник** |  |  | Долина Віктор Георгійович |
|  | (підпис) |  | (прізвище, ім’я, по батькові) |

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2019 р

Зміст

[Вступ 6](#_Toc516113382)

[1. АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ 7](#_Toc516113383)

[1.1 Визначення дійових осіб у предметній галузі 7](#_Toc516113384)

[1.2 Постановка задачі 7](#_Toc516113385)

[2. РОЗРОБКА ПРОГРАМНОЇ СИСТЕМИ 8](#_Toc516113386)

[2.1 Аналіз вимог 8](#_Toc516113387)

[2.1.1 Перегляд існуючих контактів 8](#_Toc516113388)

[2.1.3 Додавання нових телефони нових контактів 8](#_Toc516113389)

[2.2 Архітектура програмного забезпечення 9](#_Toc516113390)

[3. ВИБІР ПРОГРАМНОГО СЕРЕДОВИЩА 11](#_Toc516113391)

[3.1 Опис та обґрунтування вибору програмного середовища 11](#_Toc516113392)

[3.2 База даних 15](#_Toc516113393)

[3.3 Інформаційне забезпечення веб-додатку 16](#_Toc516113394)

[4 АНАЛІЗ МОЖЛИВИХ ВИКОРИСТАНЬ 21](#_Toc516113395)

[4.1 Виклик, загрузка, системні вимоги 21](#_Toc516113396)

[4.2 Графічний інтерфейс користувача 21](#_Toc516113397)

[**Висновки** 21](#_Toc516113398)

[Перелік використаних джерел 22](#_Toc516113399)

## Вступ

Веб-додаток "Телефона книга"

*Актуальність теми* полягає у тому, що на сьогоднішній день користування телефонною книгою у людей старшого покоління йде через паперові блокноти, а для малих підприємств внутрішні дані про контакти робітників по більшості збережені в файлах формату Word або Excel, що робить складним збереження та інтегрування цих даних до інших сервісів, що має на увазі збереження цих даних в базі даних та надавання API для їх використовування. Хоча аналог телефоної книги є в ресурсах Google, використовування цього рішення для корпорації має свою ціну, що може буде задорого для малих форм підприємств, домогосподарств тощо. Дивлячись на те що потенційні користувачі додатку мають складнощі в роботі з комп’ютером, необхідно як можливо простіший інтерфейс.

*Метою курсової роботи* є створення програмного рішення, що дозволить в онлайн режимі додавати контактні дані, редагувати та передивлятись результат, в можливістю надавання API для сторонніх ресурсів (мобільних додатків, внутришніх веб-ресурсів тощо).

Завдання курсової роботи обумовлене її метою:

* створення бази даних для контактів
* створення зручного та зрозумілого інтерфейсу програмного продукту для користувачів;
* ознайомлення потенційних користувачів з можливостями даного веб-сайту та веб-сервісу;

# АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

Під час аналізу предметної галузі були розглянуті існуючі задачі та розроблена інформаційна структура (концептуальна модель). Аналізуються дійові особи, та зв’язки між ними, визначаються вимоги та потреби.

## 1.1 Визначення дійових осіб у предметній галузі

Система додатку, що розробляється має чітко визначені суб’єкти. Основним основною дійовою особою предметної галузі є користувач додатку. Функціональні можливості програми забезпечують користувача зручним та зрозумілим способом вирішення задачі перегляду та редагування телефонної книги.

## 1.2 Постановка задачі

**Виходячи з поставленої мети та процесами реалізації своїх ролей дійовими особами можна окреслити наступні задачі курсового проекту.**

**Управлінням веб-додатком має керувати адміністратор веб-додатку, який має найширші права. Для більш безпечного зберігання даних за захисту їх від хакерських атак доречно відокремити роль адміністратора від функцій веб додатку, надавши доступ на ризиковане редагування бази даних тільки на локальній машині.**

**Всі користувачі додатку можуть без реєстрації відкрити веб-додаток та передивлятись телефону книгу та добавляти контакти**

**Веб-додаток має мати простий та прозорий інтерфейс дивлячись на те, що додаток орієнтований на користувачів котрим складно вправлятись з комп’ютером.**

# 2. РОЗРОБКА ПРОГРАМНОЇ СИСТЕМИ

# 2.1 Аналіз вимог

Для якісного вирішення поставлених задач курсового проекту необхідно, що б представлене рішення поєднувало в собі динамічний веб-сайт, на сторінках якого відображається інформація про надані контакти, та веб-сервіс, який використовуватиметься клієнтськими додатками, такими як десктопними додатками та мобільними додатками.

Інтрефейс динамічного веб-майту має бути зрозумілий для користувача сайтом, надавати в повному обсязі необхідну інформацію, мати доступ до сторінок сайту згідно ролі користувача.

**Веб-сервіс має вимагати від клієнтських частин авторизацію і надавати в якості відповіді інформацію в форматі JSON.**

### 2.1.1 Перегляд існуючих контактів

Дійова особа: користувач з наданою роллю користувача.

Попередні умови: відсутні.

Вихідні умови: користувачу відображаються всі контакти котри були надані до бази телефоної книги.

Нормальний напрямок виконання:

1. Користувач авторизується у веб-додатку;
2. Ідентифікується відповідно до ролі;
3. Користувачу відкривається доступ до меню особистого кабінету;
4. В особистому кабінеті користувач вибирає тип рахунку.
5. Система відображає результати вибору;

Альтернативні напрямки виконання: відсутні.

### 2.1.3 Додавання нових телефони нових контактів

Дійова особа: користувач.

Попередні умови: відсутні.

Вихідні умови: створення нових записів контактів в базі даних.

Нормальний напрямок виконання:

1. Користувач відкріває список існуючих контактів;
2. Користувач відкріває з цієї сторінки форму додавання нового контакту;
3. Користувач зберігає контат на веб формі;
4. Система зберігає новий рахунок в базі даних;

Альтернативні напрямки виконання:

1. Користувач відкріває список існуючих контактів;
2. Користувач відкріває з цієї сторінки форму додавання нового контакту;
3. Користувач не зберігає контакт;
4. Перехід на сторінку з переглядом контактів.

## 2.2 Архітектура програмного забезпечення

Розроблюване програмне рішення є веб-додатком, доступ користувачів до якого виконується за допомогою браузеру або десктопного чи мобільного клієнта. Користувач може завантажити з сайту собі на локальний комп'ютер дестопний додаток і використовувати його як клієнта веб-сервісу, отримуючи ту ж функціональність і інформацію, яку він може отримати через фронтенд веб-додатку. Також, користувач може завантажити з сайту та встановити на своєму мобільному пристрої мобільний додаток і використовувати його як клієнта веб-сервісу, отримуючи ту ж функціональність і інформацію, яку він може отримати через фронтенд веб-додатку.

Архітектура програмного забезпечення є трьохрівневою:

Це база даних, реалізована на MS SQL Server, бекенд, реалізовний на ASP.NET CORE 2, в якому взаємодія з базою даних здійснюється через Entity Framework Core. База даних була побудована за допомогою методу code first. Тобто, спочатку були створені сутності в проекті веб застосунку, а потім використовуючи Add-Migration та Update-Database, створена база даних. Всі реляційні зв'язки між таблицями, створення ключових полів, індексів в таблицях та інші налаштування бази даних, виконуються автоматично фреймоворком Entity Framework Core на основі створених класів моделей та коду ініціалізації.

Використовуючи цю ж технологію в подальшій розробці проекту, можна змінювати структуру бази даних для потреб проекту.

Backend реалізовано на основі ASP.NET CORE 2, застосовано як MVC 5 (controls), так і Web API з застосуванням RESTful Architecture. В проектах ASP.NET CORE 2 MVC та Web API вже об'єднані в одну технологію під управлінням одного базового контролу.

Фронтенд реалізовано частково на товстому клієнті ASP.NET CORE 2 з використанням мови Razor, фреймворка CSS - Bootstrap, та JavaScript - jQuery. Представлення реалізоване через файли view (.cshtml).

Також, фронтенд реалізовано частково як single page application, з використанням в якості фроненду файлу JavaScript та оновленням сторінки з застосуванням технології Ajax та jQuery. Лістинг файлу JavaScript як окремого фронтенду наведено в додатку № 2 до пояснювальної записки.

В якості додатку до веб-додатку, розроблено десктопний додаток під операційну систему Windows 10, та буде створено мобільні додатки під різні мобільні пристрої, такі як смартфони, планшети. Архитектуру для сторонніх клієнтів представлено на Рисунку 2.2.



Рис. 2.2 Архітектура тонкого клієнту

# ВИБІР ПРОГРАМНОГО СЕРЕДОВИЩА

## 3.1 Опис та обґрунтування вибору програмного середовища

Програмний модуль вибору дисциплін це комплекс з двох частин: backend’у та frontend’у. Перша частина реалізує сприйняття, перевірки та обробки даних, їх трансформації до вигляду запитів до БД, а також надає дані для передачі у другу частину в універсальному вигляді JSON – файлу. Backend частину було реалізовано у програмному середовищі MICROSOFT VISUAL STUDIO 2013 з використанням мови програмування C#. Також було використано СУБД SQL SERVER 2012 та додаток MySQL Workbench.

Microsoft Visual Studio  — серія продуктів фірми Майкрософт, які включають інтегроване середовище розробки програмного забезпечення та ряд інших інструментальних засобів. Ці продукти дозволяють розробляти як консольні програми, так і програми з графічним інтерфейсом, в тому числі з підтримкою технології Windows Forms, а також веб-сайти, веб-застосунки, веб-служби як в рідному, так і в керованому кодах для всіх платформ, що підтримуються Microsoft Windows, Windows Mobile, Windows Phone, Windows CE, .NET Framework, .NET Compact Framework та Microsoft Silverlight. Дана серія продуктів гарно себе зарекомендувала своїм багатим функціоналом та умовною безкоштовністю використання.

SQL Server - це комплексне інтегроване, закінчене рішення обробки даних, що надає всім користувачам організації найбільш безпечну, надійну і продуктивну платформу для даних підприємства та програм BI. SQL Server 2012 надає як IT-професіоналам, так і інформаційним працівникам знайомий інструментарій, знижуючи складність створення, розгортання, управління та використання додатків обробки та аналізу даних підприємства на цілому ряді платформ, від мобільних пристроїв до систем зберігання даних масштабу підприємства. Володіючи широким набором можливостей і підтримуючи спільну роботу з існуючими системами і здатністю автоматизувати рутинні завдання, SQL Server 2012 є закінченим рішенням управління даними для підприємства будь-якого масштабу [10].

С# представляє мову програмування, призначену для розробки різноманітних корпоративних додатків, що виконуються в середовищі .NET Framework [4]. Мова C#, що є розвитком мов C і C++, є простим, сучасним, універсальна і об'єктно-орієнтованим рішенням значного кола задач. Код C# компілюється як керований код, що дозволяє користуватися службами загального мовного середовища виконання. До числа таких служб входять взаємодія мов, збірка сміття, розширена безпека та підтримка розширеного відстеження версій.

Підтримка мови Visual C# охоплює шаблони проектів, конструктори, сторінки властивостей, майстри коду, модель об'єктів та інші засоби середовища розробки.

Frontend частина отримує дані в JSON - форматі, здійснює обробку та виводить їх у зручному для користувачів вигляді. Вона реалізована з використанням HTML5 , CSS3 та jQuery. Мова розмітки HTML5 та CSS3 необхідні програмування візуальної частини, що відображається у браузері користувача. jQuery - це JavaScript-фреймворк, який забезпечує обробку отриманого від backend JSON - файлу. За його допомогою формуються дані для заповнення таблиць вибору дисциплін у браузері зі зручним представленням для студента, викладача та методиста.

Фреймворк працює зі сторінкою HTML, що містить додаткові атрибути і пов'язує області вводу або виводу сторінки з моделлю, яка є звичайними змінними JavaScript. Значення цих змінних задаються вручну або отримуються зі статичних або динамічних JSON-даних.

jQuery спроектований з переконанням, що декларативне програмування найкраще пасує для побудови інтерфейсів користувача та опису програмних компонентів, в той час як імперативне програмування пасує для опису бізнес-логіки. Фреймворк адаптує та розширює традиційний HTML, щоб забезпечити двосторонню прив'язку даних для динамічного контенту, що дозволяє автоматично синхронізувати модель та вид. У результаті jQuery робить простішим та швидшим маніпуляції з DOM-ом сторінки.

HTML [3] (англ. HyperText Markup Language — мова розмітки гіпертекстових документів) — стандартна мова розмітки веб-сторінок в Інтернеті. Більшість веб-сторінок створюються за допомогою мови HTML (або XHTML). Документ HTML оброблюється браузером та відтворюється на екрані у звичному для людини вигляді. HTML є похідною мовою від SGML, успадкувавши від неї визначення типу документу та ідеологію структурної розмітки тексту.

HTML разом із каскадними таблицями стилів та вбудованими скриптами — це три основні технології побудови веб-сторінок.

Каскадні таблиці стилів — спеціальна мова, що використовується для опису сторінок, написаних мовами розмітки даних.

Найчастіше CSS [1] використовують для візуальної презентації сторінок, написаних HTML та XHTML, але формат CSS може застосовуватися до інших видів XML-документів.

Специфікації CSS були створені та розвиваються Консорціумом Всесвітньої мережі.

CSS має різні рівні та профілі. Наступний рівень CSS створюється на основі попередніх, додаючи нову функціональність або розширюючи вже наявні функції. Рівні позначаються як CSS1, CSS2 та CSS3. Профілі — сукупність правил CSS одного або більше рівнів, створені для окремих типів пристроїв або інтерфейсів. Наприклад, існують профілі CSS для принтерів, мобільних пристроїв тощо.

CSS (каскадна або блочна верстка) прийшла на заміну табличній верстці веб-сторінок. Головна перевага блочної верстки — розділення змісту сторінки (даних) та їхньої візуальної презентації.

CSS використовується авторами та відвідувачами веб-сторінок, щоб визначити кольори, шрифти, верстку та інші аспекти вигляду сторінки. Одна з головних переваг — можливість розділити зміст сторінки (або контент, наповнення, зазвичай HTML, XML або подібна мова розмітки) від вигляду документу (що описується в CSS).

Таке розділення може покращити сприйняття та доступність контенту, забезпечити більшу гнучкість та контроль за відображенням контенту в різних умовах, зробити контент більш структурованим та простим, прибрати повтори тощо. CSS також дозволяє адаптувати контент до різних умов відображення (на екрані монітора, мобільного пристрою (КПК), у роздрукованому вигляді, на екрані телевізора, пристроях з підтримкою шрифту Брайля або голосових браузерах та ін.). Один і той самий HTML або XML документ може бути відображений по-різному залежно від використаного CSS. Інформація про стиль для усього сайту або його частин може міститися в одному .css-файлі, що дозволяє швидко робити зміни в дизайні та презентації сторінок;

Різна інформація про стилі для різних типів користувачів: наприклад великий розмір шрифту для користувачів з послабленим зором, стилі для виводу сторінки на принтер, стиль для мобільних пристроїв;

Сторінки зменшуються в об'ємі та стають більш структурованими, оскільки інформація про стилі відділена від тексту та має певні правила застосування і сторінка побудована з урахуванням їх;

Прискорення завантаження сторінок і зменшення обсягів інформації, що передається, навантаження на сервер та канал передачі. Досягається за рахунок того, що сучасні браузери здатні кешувати (запам'ятовувати) інформацію про стилі і використовувати для всіх сторінок, а не завантажувати для кожної.

Для кращого відображення змісту веб- сторінок був використаний адаптивний веб-дизайн — це дизайн веб-сторінок, що забезпечує оптимальне відображення та взаємодію сайту з користувачем незалежно від роздільної здатності та формату пристрою, з якого здійснюється перегляд сторінки.

Метою адаптивного веб-дизайну є практичне відображення інформації та зручна навігація на всіх пристроях із доступом до інтернету (від стаціонарних ПК до мобільних телефонів). За технологією адаптивного веб-дизайну не потрібно створювати окремі версії веб-сайту. Один сайт може працювати на всьому спектрі пристроїв.

Вимоги до робочої станції, відносно невеликі, адже розроблений модуль взаємодіє з користувачем за допомогою веб - браузеру. Мінімальні вимоги:

процесор — 2 GHz для 32 і 64 бітної платформи;

ОЗУ — 2 Гб;

необхідний обсяг вільного дискового простору — не потребується (лише для інсталяції браузеру);

операційна система — Windows 10;

системні компоненти — веб-браузер з підтримкою NET Framework 4.7.; MS SQL 2014.

## 3.2 База даних

База даних (БД) — впорядкований набір логічно взаємопов'язаних даних, що використовується спільно, та призначений для задоволення інформаційних потреб користувачів. У технічному розумінні включно з системою керування БД.

Головним завданням БД є гарантоване збереження значних обсягів інформації та надання доступу до неї користувачеві або ж прикладній програмі. Таким чином БД складається з двох частин — збереженої інформації, та системи управління нею. З метою забезпечення ефективності доступу записи даних організовують як множину фактів (елементів даних).

В сучасному спектрі баз даних існують декілька принципово різних реалізацій (наприклад, в Енциклопедії технологій баз даних визначаються понад 50 видів БД).

В рамках даного курсового проекту використовується реляційна база даних Microsoft SQL Server 2014. Реляційна модель орієнтована на організацію даних у вигляді двовимірних таблиць. Кожна реляційна таблиця являє собою двовимірний масив і має наступні властивості:

* кожен елемент таблиці - один елемент даних;
* всі осередки в стовпчику таблиці однорідні, тобто всі елементи в стовпчику мають однаковий тип (числовий, символьний тощо);
* кожен стовпчик має унікальне ім'я;
* однакові рядки в таблиці відсутні;
* порядок проходження рядків і стовпчиків може бути довільним.

Microsoft SQL Server 2014 — комерційна система керування базами даних, що розповсюджується корпорацією Microsoft. Мова, що використовується для запитів — Transact-SQL, створена спільно Microsoft та Sybase. Transact-SQL є реалізацією стандарту ANSI / ISO щодо структурованої мови запитів SQL із розширеннями. Використовується як для невеликих і середніх за розміром баз даних, так і для великих баз даних масштабу підприємства. Багато років вдало конкурує з іншими системами керування базами даних.

SQL Server 2014 включає низку вдосконалень для роботи з критичними бізнес-застосунками і бізнес-аналітикою як в традиційних дата-центрах, так і в приватних, публічних і гібридних хмарних середовищах. Серед нових можливостей SQL Server 2014 виділяються SQL Server AlwaysOn (рішення підтримки високого рівня доступності даних та аварійного відновлення), xVelocity (технологія збільшення продуктивності сховищ даних та програм бізнес-аналітики), нові рішення в області візуалізації PowerPivot і PowerView для створення звітів і аналітичних програм з Excel і SharePoint, покращені інструменти для інтеграції даних і управління ними, включаючи SQL Server Data Quality Services і Master Data Services.

## 

## 3.3 Інформаційне забезпечення веб-додатку

**3.3.1 Концептуальний рівень проектування**

Концептуальний рівень представлення моделі бази даних відображає абстрактне уявлення даних предметної області в структурах, що відображають атрибутивний складу кожного елемента (об'єкта) моделі і особливості зв'язків між ними.

Одним з основних варіантів концептуального представлення моделі бази даних є модель в нотації Пітера Чена, запропонована в 1976 р .На той момент це подання давало можливість розробнику вказати всі необхідні компоненти майбутньої бази даних і розглядалося на об'єднаному концептуально-логічному рівні уявлення бази даних.

Для сучасних баз даних, які включають значно більшу кількість елементів структури, ніж це було представлено в 1976 р, модель Пітера Чена може бути віднесена тільки до концептуального рівня, відображаючи основні елементи предметної області з використанням наступної термінології: сутність (об'єкт), зв'язок, атрибут, тип зв'язку, потужність (кардинальність) зв'язку, відносини.

Тут варто врахувати, що термін "Відношення" в нотації Пітера Чена має дещо відмінний зміст по порівняння з цим же терміном в реляційній алгебрі, де під ставленням розумілося будь-яке представлення даних. Так як нотація Пітера Чена означає що сутності відображуються у вигляді прямокутнків, зв'язки у вигляді ромбів. Якщо сутність бере участь у відносинах, вони пов'язані линією. Якщо відносини не є обов'язковими, то лінія пунктирна. Атрибути позначаються в вигляді овалів і пов'язані з однією сутністю або зв'язком. Овал похідних атрибутів зображується пунктирним контуром.

На рівні моделі термін "Відношення" з реляційної алгебри уточнюється шляхом виділення в окремий термінологічний елемент "Сутність" відносин, що описують конкретний об'єкт предметної області з фіксованим набором атрибутів, а терміном "Відношення" називаються уявлення, одержувані при взаємодії пов'язаних сутностей.

Концептуальний рівень проектування характеризується тим, що необхідно відтворити схему, яка б представляла собою елементи даних та відношень між ними.

Для реалізації веб-додатку було виділено наступні сутності:

**Користувач (User)** - особа, яка не зареєстрована на сайті як користувач веб-додатку, в тому числі веб-сервісу.

**Роль (Role) -** формально, права доступу до сторінок та сервісів веб-додатку, які надаються користувачу. Базові ролі є адміністратор (Administrator) та користувач (User).

**Роль користувача (UserRole) -** права доступу до сторінок та сервісів веб-додатку, які надані користувачу.

**Візитна карточка (PhoneCard) –** контакти (ПІБ, телефон, адресса, дата народження) що користувач додав до бази даних.

Отримана в результаті роботи ER (EntityRelation) діаграма.

Модель «сутність-зв'язок» (ER-модель) (англ. Entity-relationshipmodel або entity-relationshipdiagram) — модель даних, яка дозволяє описувати концептуальні схеми за допомогою узагальнених конструкцій блоків. ER-модель — це мета-модель даних, тобто засіб опису моделей даних. Існує ряд моделей для представлення знань, але одним з найзручніших інструментів уніфікованого представлення даних, незалежного від програмного забезпечення що його реалізує, є модель «сутність-зв'язок». Важливим є той факт, що з моделі «сутність-зв'язок» можуть бути породжені всі існуючі моделі даних (ієрархічна, мережева, реляційна, об'єктна), тому вона є найзагальнішою.

Модель сутність-зв'язок є результатом систематичного процесу, який описує та визначає деяку предметну область. Вона не визначає сам процес, а лише візуалізує його.

Дані представлені у вигляді компонентів (сутностей), які пов'язані між собою певними зв'язками, які виражають залежності і вимоги між ними, такі як: одна будівля може бути розділена на нуль або більше квартир, але одна квартира може бути розташована лише в одній будівлі.

Сутності можуть мати різні властивості (атрибути), які характеризують їх. Діаграми, створені для представлення цих сутностей, атрибутів і зв'язків графічно, називають сутність-зв'язок діаграмами.

ER-модель зазвичай реалізується в вигляді баз даних. У разі реляційної бази даних, в якій зберігаються дані в таблицях, кожен рядок кожної таблиці являє собою один екземпляр сутності.

Деякі поля даних в цих таблицях вказують на індекси в інших таблицях. Такі поля є покажчиками фізичної реалізації зв'язків між сутностями.

Діаграма ER-моделі даного проекту представлена на рисунку 3.1.

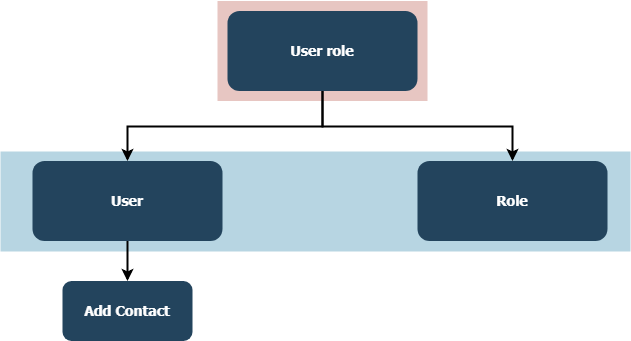


Рисунок 3.1 – Діаграма ER

**3.3.2 Логічний рівень представлення даних**

Логічний рівень представлення моделі бази даних є розширенням концептуальної моделі і доповнюється комплексом додаткових елементів, а саме: типи даних, обмеження, умовчання і т.д. Логічний рівень є більш повним поданням моделі бази даних, але порівняно з концептуальним поданням і формується з метою перевести моделі з площини предметної області в площину майбутньої бази даних.

Під час розробки веб-додатку була створена база даних (БД) PhoneCards, з єдиною таблицею PhoneCards.

Таблиця PhoneCards - відповідає сутності Phone Card, опис якої наведено в таблиці 1.

Таблиця 1 – Опис таблиці «PhoneCards»

|  |  |  |  |
| --- | --- | --- | --- |
| № | Назва поля | Тип Поля | Опис поля |
| 1 | ID | int | Унікальний ідентифікатор |
| 2 | FirstName | nvarchar(MAX) | Ім’я контакту |
| 3 | LastName | nvarchar(MAX) | Призвище контакту |
| 4 | BirthDate | datetime | Дата народження |
| 5 | Address | nvarchar(MAX) | Адреса |
| 6 | PhoneNumber | nvarchar(MAX) | Номер телефону |

# 4 АНАЛІЗ МОЖЛИВИХ ВИКОРИСТАНЬ

## 4.1 Виклик, загрузка, системні вимоги

Для коректної роботи розробленого програмного продукту на машині кінцевого користувача мають бути встановлені наступні програми:

а) .NET Framework 4.7 (або новіша версія);

б) Інтернет браузер зі списку (Google Chrome 67, Mozilla FireFox 59, Opera 53, Safari 19, Microsoft Edge або пізніших модифікацій та їхніх аналогів).

## 4.2 Графічний інтерфейс користувача

Веб-додаток має меню має декілька сторінок: сторінка до внесення контактів користувачем. Сторінка демонстрація того як користуватись WebApi, та сторінка з прикладом відповідей.

Графічний інтерфейс виконано в класичній формі старих сайтів:

зверху є кнопка повернення до дому, котра знаходитися в NavBar, що додає можливості розширити навігацію для наступних версій додатку.

Також, на сторінках сайту використовуються класичні елементи управління, такі як кнопки, текстові форми, тощо. Всі елементи мають назви призначення або функціональності.

Головне вікно програми має вибір з трьох посилань: посилання на перегляд та редагування контактів, посилання на перегляд контактів, та посилання на приклад відповіді по WebAPI (див. Додаток 1).

Вікно додавання та редагування контактів відображає список контактів, з можливості детально переглянути кожний, а також з можливістю редагувати кожний (дивись Додаток 2).

Вікно редагування та додавання контактів складаються з полей котрі треба заповнити (First Name, Last Name, Birth date, Address, Phone Number) та кнопки підтвердження та відміни (дивись Додаток 3 та Додаток 4).

Якщо користувач перейде за посиланням тонкого клієнту, то він побачить вікно зі списком контактів, та фільтрацією контактів по Id (дивись Додаток 5).

Якщо користувач перейде за посилянням прикладу WebApi то він побачить вікно з прикладом відповіді від серверу, котрий можна буде використувати для роботи в сторонніх клієнтах (дивись Додаток 6).

**Висновки**

В ході виконання даного курсового проекту були спроектовані та реалізовані веб-додаток "Телефона книга" та десктопний додаток як single page application з використанням WebApi та Json відповідей від серверу, та прикладом фільтрації.

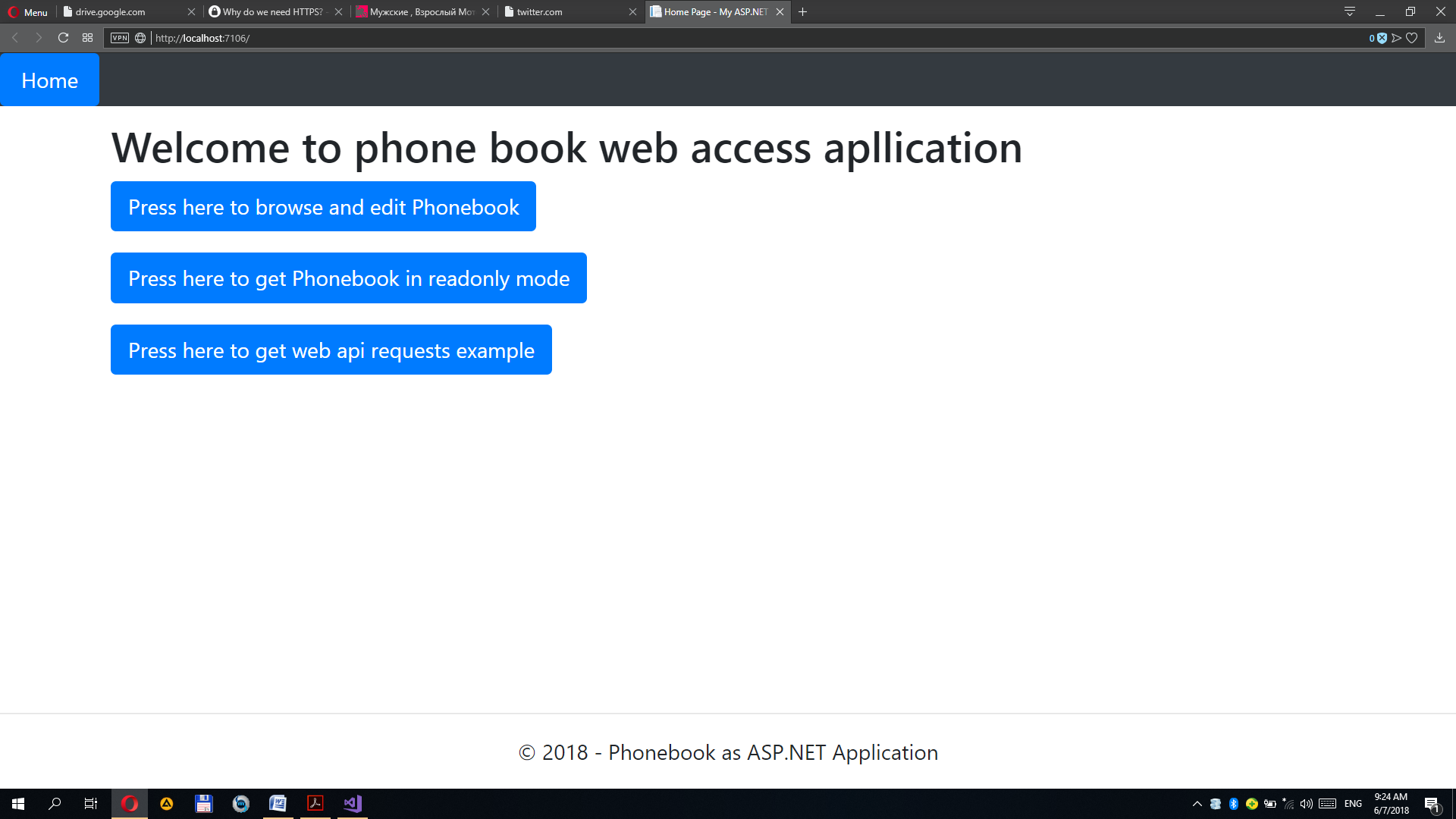
Якщо перейти за посіланням прикладу

# Перелік використаних джерел

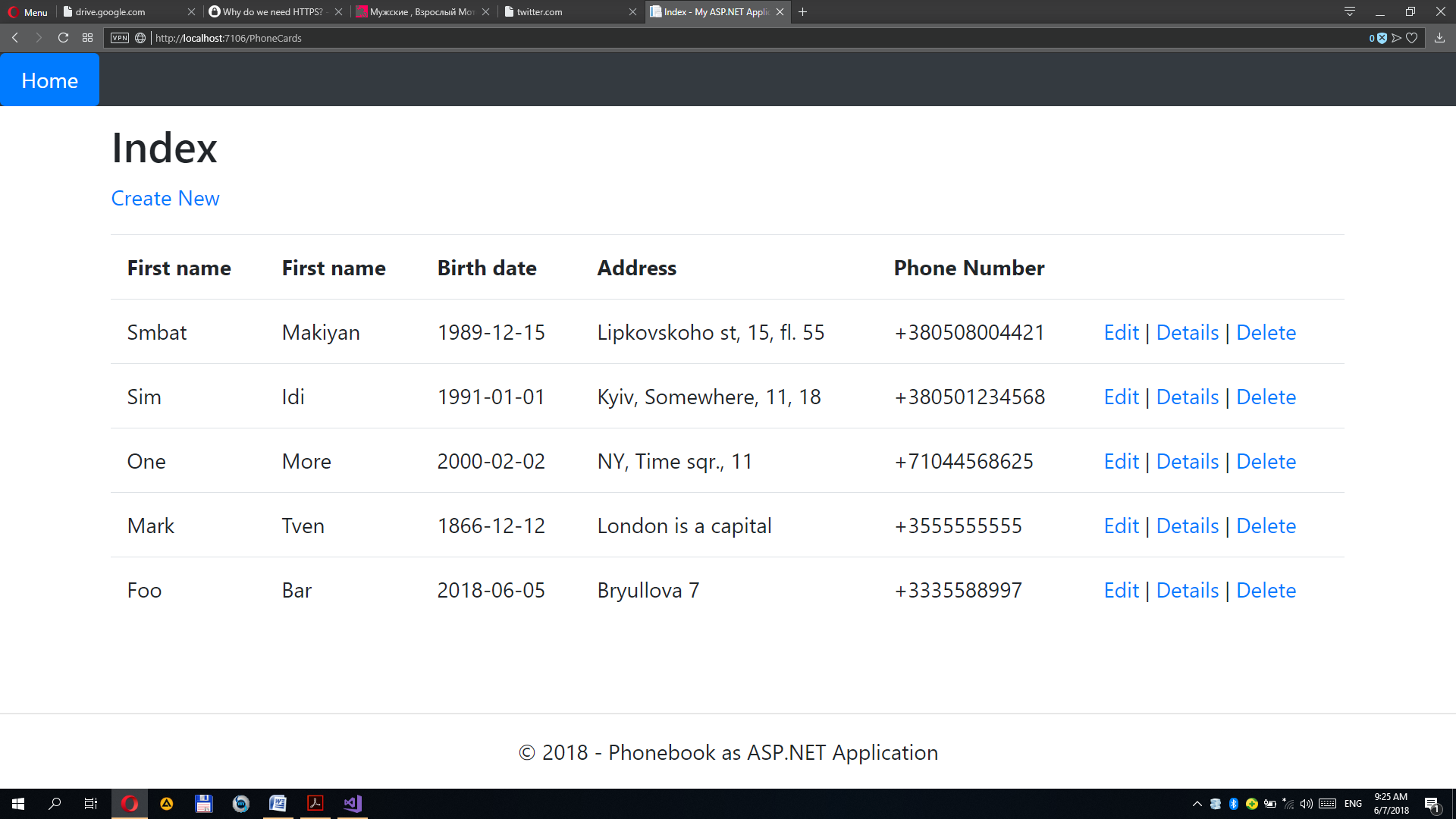
* 1. Троелсен Э. Язык программирования C# 2010 и платформа .NET 4 [Текст] / Э. Троелсен : Москв, ИЛ, 2011. – 331 с.
  2. Похлёбкин В. Национальные кухни наших народов. [Текст] / В. Похлёбкин : Центрполиграф, 2009. - 638 с.
  3. Microsoft corp. Средства и языки разработки [Електронний ресурс] – Режим доступу https://msdn.microsoft.com/ru-ru/library/aa187916.aspx
  4. Microsoft corp. Документация по Visual Studio [Електронний ресурс] – Режим доступу https://docs.microsoft.com/ru-ru/visualstudio/
  5. Ярошенко Ф. Проектное управление. Управление инновационными проектами и программами [Текст] / Ф. Ярошенко : 2011. – 268 с.
  6. Майо Д. С# Builder. Быстрый старт [Текст] / Д. Майо : Бином-Пресс, 2005. – 384 с.
  7. Мартин Г. SQL довідковий посібник. Видання 2-е. [Текст] / М. Лори, Г. Мартин : НТ Пресс, 2011. – 354 c

## Додатки

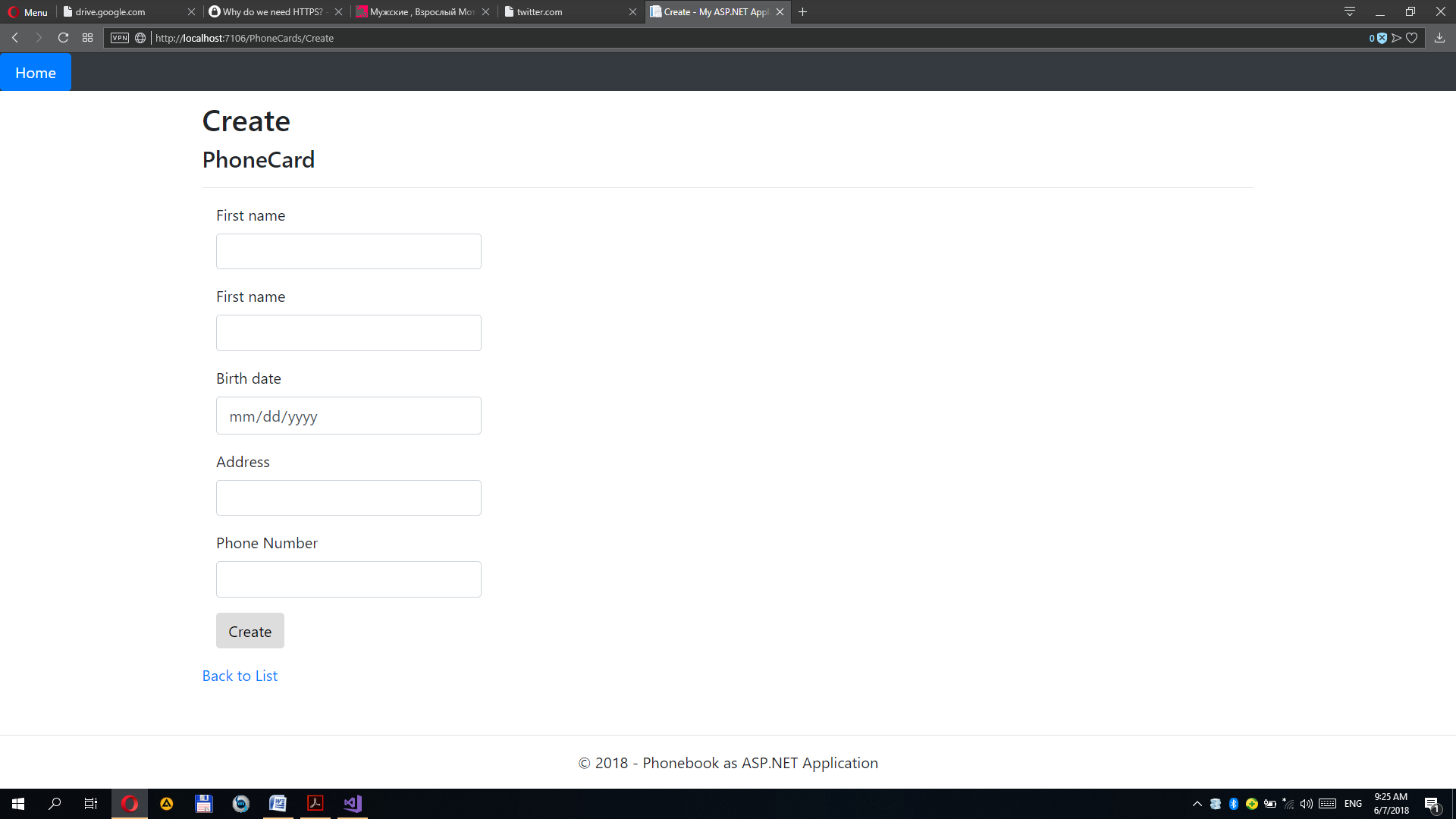
**Додаток 1 – Головне вікно**

****

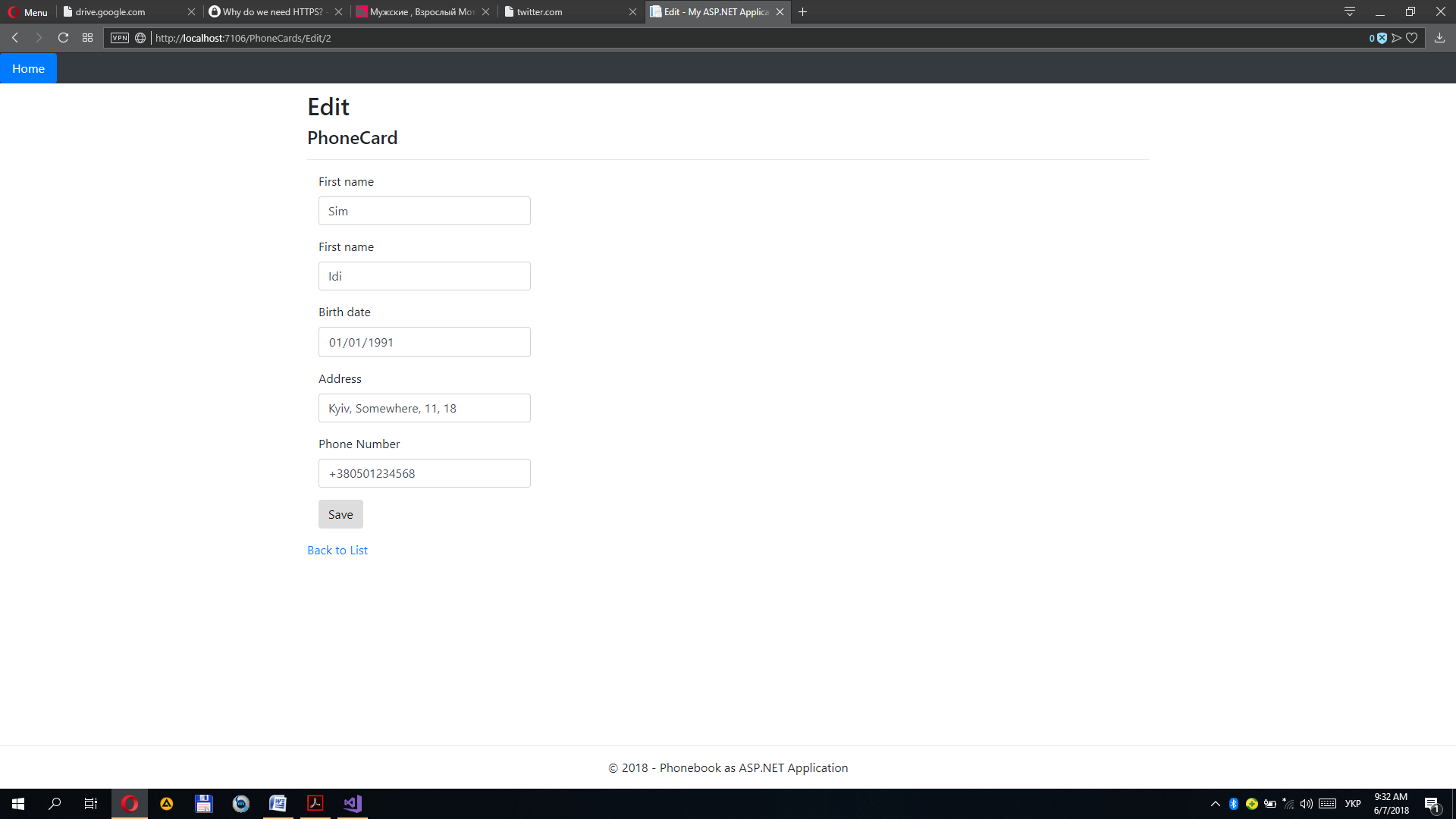
**Додаток 2 –Вікно перегляду**

****

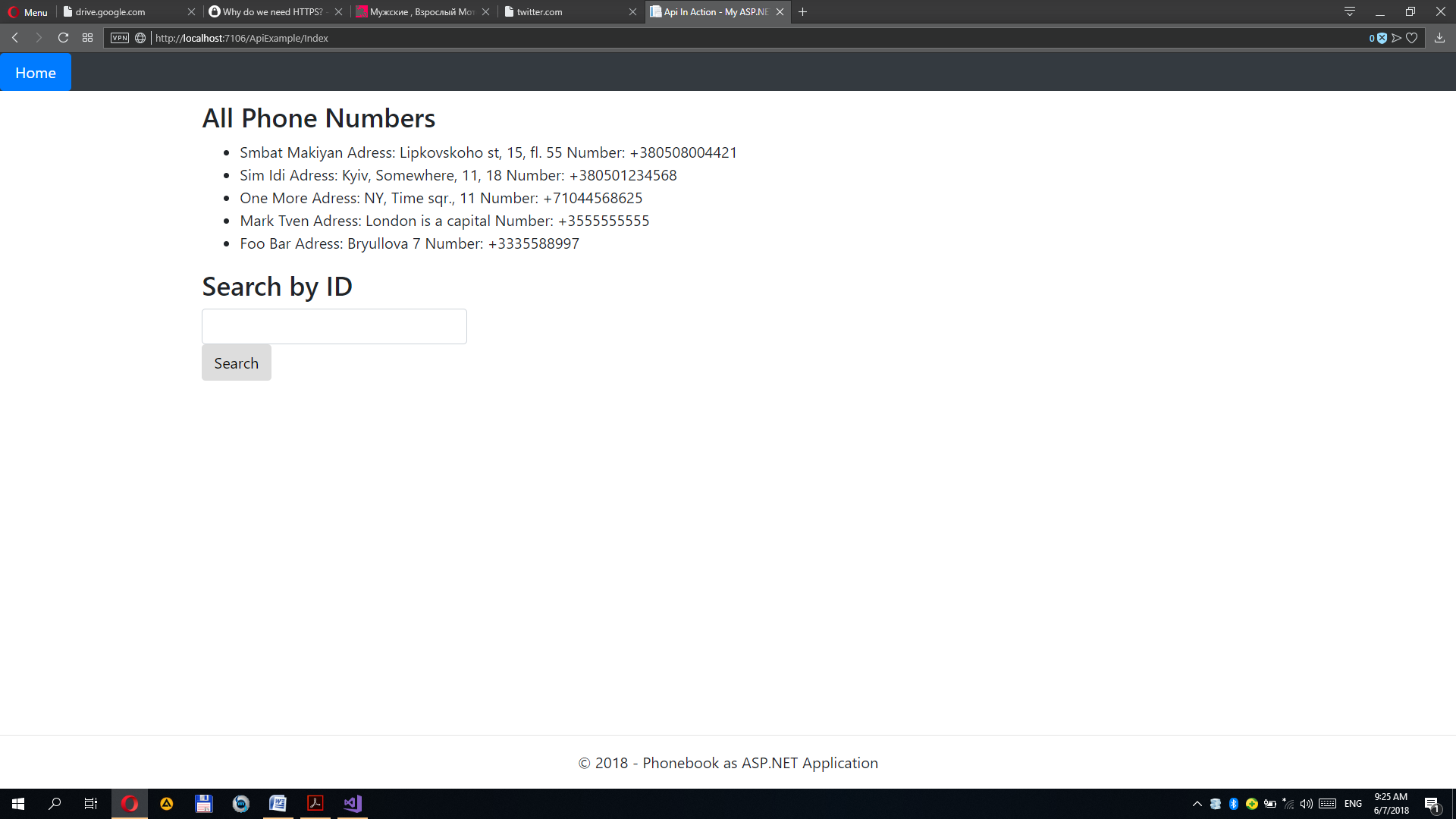
**Додаток 3 – Вікно додавання**

****

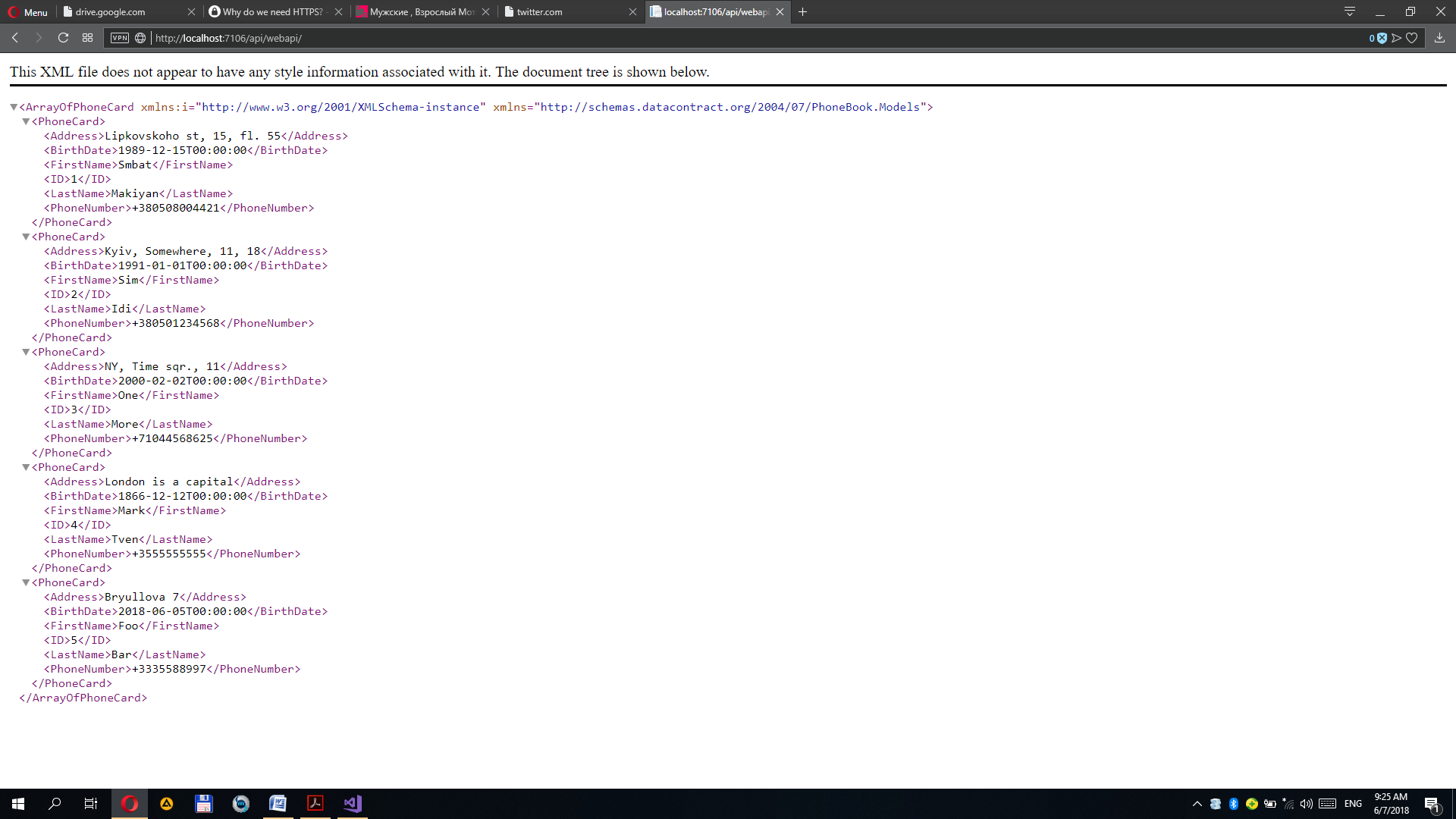
**Додаток 4 – Вікно редагування**

****

**Додаток 5 – Вікно тонкого клієнту**

****

**Додаток 6 – Приклад відповіді від WebAPI в форматі XML**

****