Міністерство освіти і науки України

Одеський національний політехнічний університет

Інститут комп'ютерних систем

Кафедра інформаційних систем

КУРСОВА РОБОТА

з дисципліни “ Веб – технології та веб-дизайн ”

Тема “ Пошукова-новігаційна система „Gator”: клієнтська частина та представлення інформації для бекенду ”

ВИКОНАЛА:

Студентка групи ЛАІ-195

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

Шакун Т.Л. (підпис)

ПЕРЕВІРИВ:

Керівник \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис)

Одеса 2021

Міністерство освіти і науки України

Одеський національний політехнічний університет

Інститут комп'ютерних систем

Кафедра інформаційних систем

ЗАВДАННЯ

НА КУРСОВУ РОБОТУ

Шакун Таїсії Леонідівни

1. Тема роботи: Пошукова-новігаційна система „Gator”

2. Термін здачі студентом закінченої роботи 9.12.2021

3. Початкові дані до проекту (роботи):

Реалізувати веб-додаток

4. Зміст розрахунково-пояснювальної записки (перелік питань, яки належить розробити):

В курсовій роботі розглядається процес створення програмного продукту Gator на основних етапах: визначення вимог до програмного продукту, планування процесів розробки, проектування, конструювання, верифікація, розгортання та валідація програмного продукту.

Завдання видано \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Завдання прийнято до виконання \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Анотація

В курсовій роботі розглядається процес створення програмного продукту Gator на основних етапах: визначення вимог до програмного продукту, планування процесів розробки, проектування, конструювання, верифікація, розгортання та валідація програмного продукту.

Даний проект повинен виконувати агрегацію можливих даних для переміщення з місця на місце по місту.

Робота виконувалась в команді з декількох учасників: Шакун Таїсії та Нікітіна Павло

Тому у розділах «Проектування», «Конструювання» та  «Верифікація» детальніше описано лише одну частину з урахуванням планів проведених робіт з розділу з описом особливостей конструювання:

програмних модулів в інструментальному середовищі Visual Studia з використанням фреймворку Asp.net та мови програмування C# та JS

Поточну версію пояснювальної записки до результатів роботи розміщено на GitHub-репозиторії за адресою: https://github.com/TaisiaShakun/Gator

Назва проекту Gatorv3

Аннотация

В курсовой работе рассматривается процесс создания программного продукта Gator на основных этапах: определение требований к программному продукту, планирование процессов разработки, проектирование, конструирование, верификация, развертывание и валидация программного продукта.

Этот проект должен производить агрегацию возможных данных для перемещения с места на место по городу.

Работа выполнялась в команде из нескольких участников: Шакун Таисии и Никитина Павел

Поэтому в разделах «Проектирование», «Конструирование» и «Верификации» подробнее описана только одна часть с учетом планов проведенных работ с описанием особенностей конструирования:

• программных модулей в инструментальном среде Visual Studia с использованием фреймворка Asp.net и языка программирования C# и JS

Текущая версия пояснительной записки к результатам работы размещена на GitHub-репозитории по адресу: <https://github.com/TaisiaShakun/Gator>

Summary

The course work considers the process of creating a Gator software product at the main stages: defining the requirements for the software product, planning the processes of development, design, construction, verification, deployment and validation of the software product.

This project should aggregate possible data to move from place to place in the city.

The work was performed in a team of several participants: Shakun Taisiya and Nikitina Pavlo

Therefore, in the sections "Design", "Design" and "Verification" is described in more detail only one part, taking into account the plans of the work with a description of the features of design:

• software modules in the Visual Studio toolkit using the Asp.net framework and the C # and JS programming languages

The current version of the explanatory note to the results of the work is posted on the GitHub repository at: https://github.com/TaisiaShakun/Gator

[1 Анализ предметной области и выбор инструментов разработки 7](#_Toc89970926)

[1.1 Огляд аналогів 7](#_Toc89970927)

[1.2 Вибір инструментів розроботки 8](#_Toc89970928)

[1.2.1 Вибір мови програмування 8](#_Toc89970929)

[1.2.2 Вибір середи розробки 8](#_Toc89970930)

[1.2.3 Вибір фреймворків и бібліотек 9](#_Toc89970931)

[2 Проектирование приложения или сервиса 10](#_Toc89970932)

[2.1 Входные, выходные и управляющие потоки приложения) 10](#_Toc89970933)

[2.2 Функциональные и нефункциональные требования к разработке 12](#_Toc89970934)

[2.4 Идентификация архетипа приложения 14](#_Toc89970935)

[2.5 Представление пользовательского интерфейса (UI View) 14](#_Toc89970936)

[2.6 Логическое представление (Logical View) 15](#_Toc89970937)

[2.7 Представление процессов (Process View) 16](#_Toc89970938)

[2.7.1 Диаграмма деятельности 16](#_Toc89970939)

[2.7.2 Диаграмма последовательности 17](#_Toc89970940)

[3 Реализация приложения или сервиса 18](#_Toc89970941)

[3.1 Структура проекта 18](#_Toc89970942)

[3.2 Описание кода приложения или сервиса 18](#_Toc89970943)

[3.3 Функциональное тестирование 22](#_Toc89970944)

[Висновки 25](#_Toc89970945)

## 1 Анализ предметной области и выбор инструментов разработки

## 1.1 Огляд аналогів

Для огляду аналогів ми використовували три варіанта це Google Maps – великий веб сервіс з баготьма можливостями, але без можливості бачити більшість приватних варіантів переміщення таких як: таксі, електроні самокати, велосипеди та інші.

Розглядаючи сервіс 838 ми маємо зворотну ситуацію ми маємо можливість побачити як дістатися з місця на місце на таксі але не маємо можливості бачити як дійти пішки чи на тролейбусі

“ТРАНСПОРТ ОДЕССЫ” – цей сервіс більш ліпший коли те вже визначився з варіантом руху, бо він окрім маршруту державного транспорту не може показати, але може показати де він зараз. На момент написання курсової дана функція вимкнута із-за проблем сервісу.

|  |  |  |
| --- | --- | --- |
| Назва | Вартість | Коментар |
| Google Maps | Безкоштовно | не має ціни на часні види транспорту |
| 838 | Безкоштовно | не має нічого окрім таксі |
| ТРАНСПОРТ ОДЕССЫ | Безкоштовно | Відсутність часного виду транспорту |

## 1.2 Вибір инструментів розроботки

### 1.2.1 Вибір мови програмування

У випадку вибору мови програмування, основний упор треба робити на мову з якою буде пов'язане наше API і мати більше прикладів для вивчення та успішного тестування. В цьому випадку можуть розглядатися мови, як Python, PHP, JavaScript. Всі ці мови є гарними у використанні для роботи з запитами. Але так як для візуалізації карти був обраний сервіс MapBox і вся документація основна на мові Javascript тому для зручності реалізації та тестування був обраний на frontend мову Javascript c поміссю Asp.net для зручнішого з'єднання С# що був обраний для BeckEnd у зв'язку з більш поглибленими у ньому знань.

### 1.2.2 Вибір середи розробки

В цьому пункту розглянемо три варіанта середи розробки, що можуть використовуватися у разі розробки на мові С#, що викорустовується на BeckEnd програмного забезпечення, або на мові Jscript, що викорустовується на FrontEnd, тому інші середи розробки що підтримують інші мови программування розглядати немає сенсу.

Для розробки даного ПЗ була обрана середовище розробки така як Visual Studia у зв'язку з тим, що вона забезпечує сучасний функціонал, а також підтримує можливість програмування на всіх мовах використаних для створення ПЗ.

|  |  |
| --- | --- |
| [NetBeans](https://netbeans.org/) | Не підтримує JavaScript |
| Visual Studio Code | Має більш скорочений функціонал ніж Visual Studio |
| Atom | Не підтримує С# |

Для підтвердження мовленого вище розглянемо аналоги. Наприклад [NetBeans](https://netbeans.org/) має добрий функціонал, але не має підтримки програмування на мові JavaScript.

Visual Studio Code можно мовити скоротшена версія Visual Studio має більшість можливостей Visual Studio, але не всі.

Atom у свою чергу має підтримку JS але не має підтримки C#.

### 1.2.3 Вибір фреймворків и бібліотек

При виборі Фреймворків ми керувалися знанням що, ліпшим фреймворком для написання веб-додатку на мові С# є Asp.net

Також для використання обміну даними між сервером та представленням використовуєтся фреймворк Jquery

# 2 Проектирование приложения или сервиса

## 2.1 Входные, выходные и управляющие потоки приложения)

Вхідні потоки

|  |  |
| --- | --- |
| Ідентифікатор  функції  (назва) | Засіб INPUT-потоку |
| FR1.1 | * стандартна комп`ютерна клавіатура; * 2/3-кнопочний маніпулятор типу "миша"; * сенсорний екран (Touchscreen, Touchpad, Multi-touch); |
| FR3.1.1 | * стандартна комп`ютерна клавіатура; * 2/3-кнопочний маніпулятор типу "миша"; * сенсорний екран (Touchscreen, Touchpad, Multi-touch); |
| FR3.2.1 | * стандартна комп`ютерна клавіатура; * 2/3-кнопочний маніпулятор типу "миша"; * сенсорний екран (Touchscreen, Touchpad, Multi-touch); |

Вхідні потоки дозволяють контактувати з веб додатком користувачеві, тому так як наш додаток націлен на використування як дома так і наулиці можо буде користуватися як і засобами вводу з компютера так і з мобільного пристрою

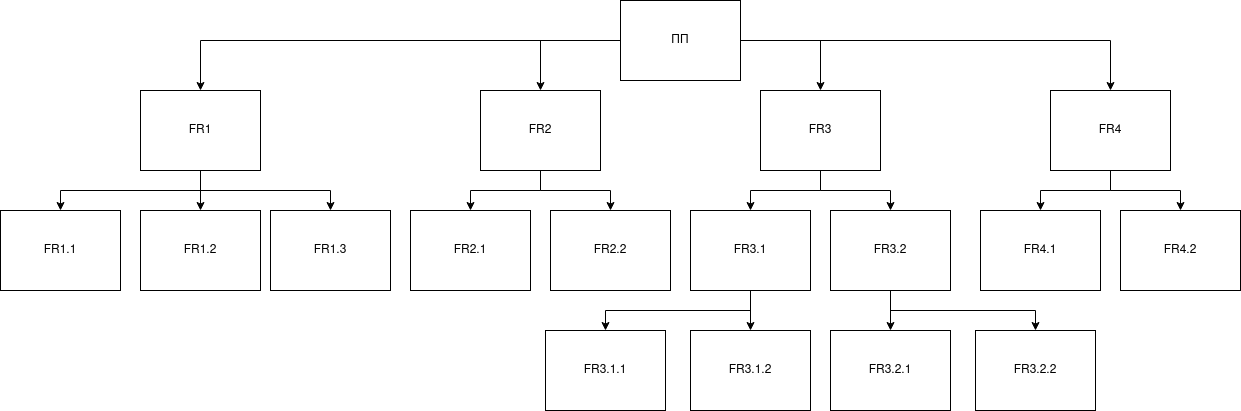
Вихідні - потоки

|  |  |
| --- | --- |
| Ідентифікатор функції (назва) | Засіб OUTPUT - потоку |
| FR1.1 | графічний інтерфейс |
| FR1.2 | графічний інтерфейс |
| FR1.3 | графічний інтерфейс |
| FR2.1 | графічний інтерфейс |
| FR2.2 | графічний інтерфейс |
| FR3.1.1 | графічний інтерфейс |
| FR3.1.2 | графічний інтерфейс |
| FR3.2.1 | графічний інтерфейс |
| FR3.2.2 | графічний інтерфейс |
| FR4.1 | графічний інтерфейс |
| FR4.2 | графічний інтерфейс |

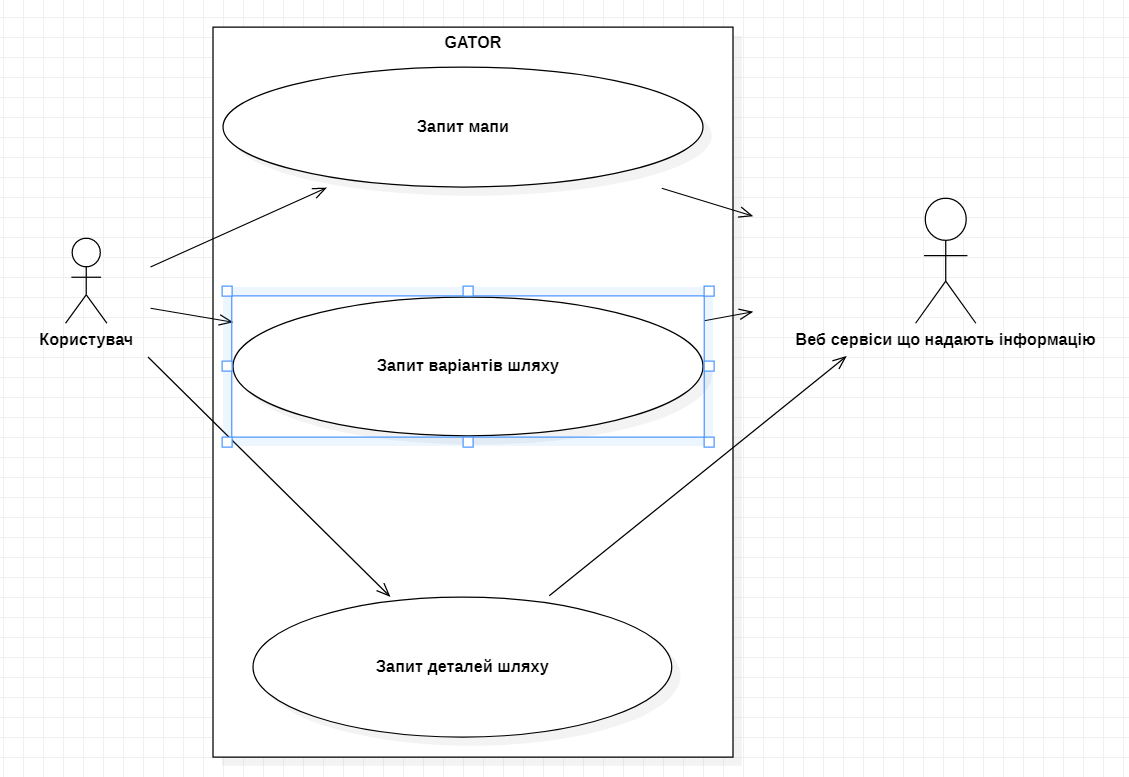
Всі вихідні потоки в даній курсовій виходять у графічний інтерфейс у зв'язку з тим, що більше і не потрібно лише відображення після обробки значень

## 2.2 Функциональные и нефункциональные требования к разработке

|  |  |
| --- | --- |
| Ідентифікатор функції  (назва) | Назва функції |
| FR1 | Запуск ПП |
| FR1.1 | Успішне завантаження ПП |
| FR1.2 | Відсутність інтернету |
| FR1.3 | Помилка сервера при завантаженні |
| FR2 | Отримання мапи |
| FR2.1 | Успішне отримання мапи |
| FR2.2 | Помилка сервісу, який надає мапу |
| FR3 | Введення координат |
| FR3.1 | Введення точки старту |
| FR3.1.1 | Успішне введення точки |
| FR3.1.2 | Точка відсутня на мапі |
| FR3.2 | Введення кінцевої точки |
| FR3.2.1 | Успішне введення точки |
| FR3.2.2 | Точка відсутня на мапі |
| FR4 | Отримання варіантів руху |
| FR4.1 | Успішне отримання ≥ 1 варіанту |
| FR4.2 | Помилка сервісів, які надають інформацію про транспортні засоби |



2.3 Диаграмма использования (Use cases diagram) приложения и ее описание



## 2.4 Идентификация архетипа приложения

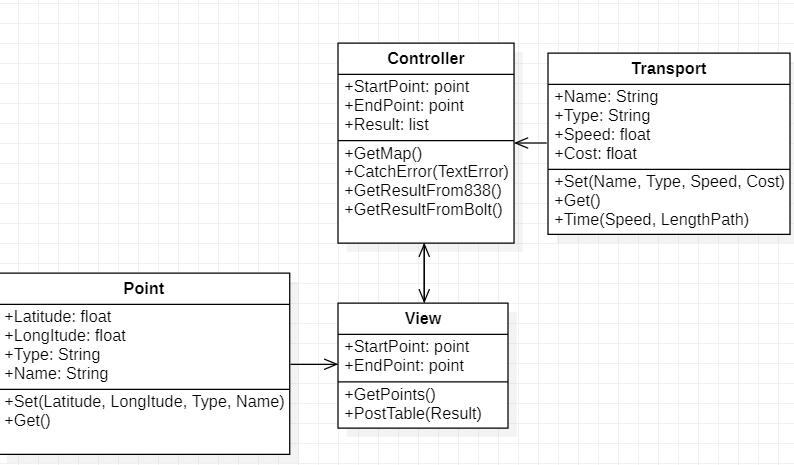
WEB Application (WA). Классическое WEB приложение, предоставляющее WWW интерфейс посредством HTML/DHTML.

Наше приложение использует класическую архитектуру веб приложения так как у нас есть клентская часть написанная на JS и HTML, который предоставляет интерфейс

## 2.5 Представление пользовательского интерфейса (UI View)

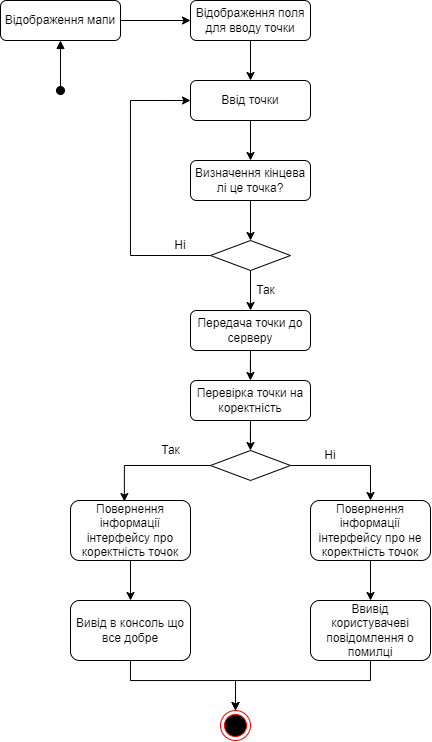
|  |  |
| --- | --- |
| Ідентифікатор функції  (назва) | Зовнішній пристрій |
| FR1.1 | Смартфон / Notebook / Desktop-персональний комп`ютер |
| FR1.2 | Смартфон / Notebook / Desktop-персональний комп`ютер |
| FR1.3 | Смартфон / Notebook / Desktop-персональний комп`ютер |
| FR2.1 | Смартфон / Notebook / Desktop-персональний комп`ютер |
| FR2.2 | Смартфон / Notebook / Desktop-персональний комп`ютер |
| FR3.1.1 | Смартфон / Notebook / Desktop-персональний комп`ютер |
| FR3.1.2 | Смартфон / Notebook / Desktop-персональний комп`ютер |
| FR3.2.1 | Смартфон / Notebook / Desktop-персональний комп`ютер |
| FR3.2.2 | Смартфон / Notebook / Desktop-персональний комп`ютер |
| FR4.1 | Смартфон / Notebook / Desktop-персональний комп`ютер |
| FR4.2 | Смартфон / Notebook / Desktop-персональний комп`ютер |

## 2.6 Логическое представление (Logical View)

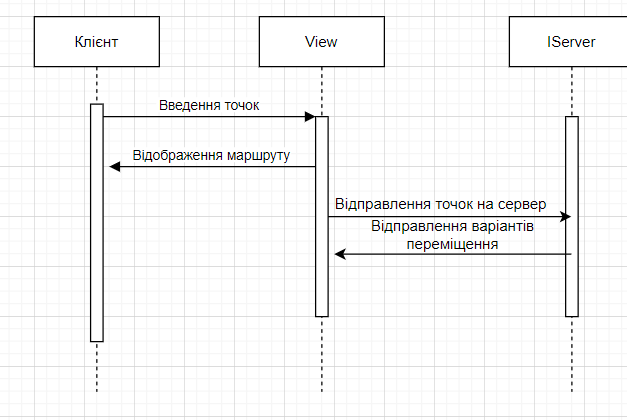


## 2.7 Представление процессов (Process View)

## 2.7.1 Диаграмма деятельности



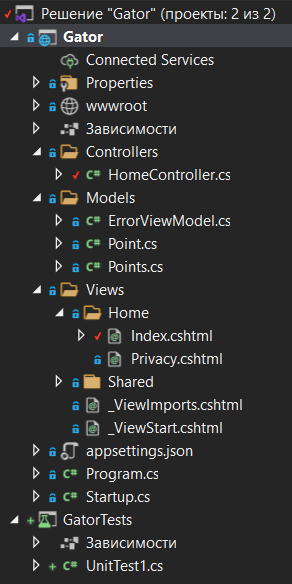
## 2.7.2 Диаграмма последовательности



Як ми бачимо на діаграммі моя частина додатка працює так, ми отримуємо точки від користувача, наш інтерфейс передає ці точки до сервер він їх оброблює та повертає варіанти перемщення

3 Реализация приложения или сервиса

## 3.1 Структура проекта



Структура проекта

Даний додаток розроблен на патерні MVC тому ми маємо три ключові моменти це View – що відповідає про відображення нашого додатку, Controller – що обрабляють отриману інформацію, та Model – що зберігають інформацію

## 3.2 Описание кода приложения или сервиса

|  |
| --- |
| В цьому додатку як Framework викоритовувався JQuery з його допомогою відправляються запити до контролеру і його структура виглядає так  url: куди відправити  type: Якого типу выдправляємий запит, для отримання даних чи відправки  headers: Якого типу внутрішня частина із самими даними  data: Самі дані  success: Що робити при повернені позитивного відгуку нам запит  error: Що робити при повернені негативного відгуку на запит |
| В додатку використовувалось 3 класа це homeController, Points,Point де Point використовуєтся для зберегання даних точок і має лише поля для цього, Points зберігає початкову та кінцеву точку для роботи із маршрутом і також має лише два поля StartPoint – EndPoint  homeController головний клас котрий відповідає за правильну обробку даних на стороні BeckEnd та поверненя оброблених даних на FrontEnd, він має методи Index()Send()Error()CathErrors()котрі відповідають за підгрузку строниці, роботи із запитами та даними, перенаправлення у разі глобальної помилки на стороні FrontEnd до сторінки з повідомленням про це, та метод обробки помилок даних котрі будуть використовуватись для подальшого векористання та пришли від користувача |
| В даному додатку є 4 основні функції/методи це function\_click, SendPoints, Send,CatchErrors та вспоміжні, що використовуваються для стабільної роботи додатку Index(), Error()  Як мовилось раніше Index() відповідає за завантаження сторінки  Error() перенаправляє до вспоміжної сторінки при глобальній помилці, чи недоступності сторінки Index  function\_click отримує дані із натискання мищі по мапі за допомогою івентів, що має API та передачі цих даних до функції SendPoints попередньо первіряючи на обновлення точок та їх наявність  SendPoints() за допомогою раніше описаного фреймворку, відправляє запит і все, тоб то ця функція використовується лише для виклачення та формування запиту за допомогою Framework-у JQuery.  Send() отримує точки із SendPoints() та використовує функцію CatchErrors для перевірки даних, та після формування зворотньої відповіді в залежності від результату  CatchErrors() при отримані точок перевіряє їх на усі варіанти можливих похибок такі як виход за дозволені границі значень за допомогою оператору if…else та повертає відповідь про проходження тестування |
| Нижче преведени схеми алгоримів роботи основних методів    Схема1 Алгоритму методу Send    Схема2 Алгоритму методу CatchErrors    Схема3 Алгоритму методу function click |
|  |
| Схема4 алгоритму методу SendPoint |

## 3.3 Функциональное тестирование

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | FR ID | Test  Case  ID | Опис значень вхідних даних  до методу (процедури, функції) | Опис очікуваних значень  результату виконання методу | | FR1 | TC1 | Нічого не передаєтся | Відображаєтся мапа міста | | FR1 | TC2 | point1 = point(82.93057,82.93057)  point2= point(102.93057,102.93057) | Відображаєтся маршрут з point1до point2 | | FR1 | TC3 | point1 = point(82.93057,82.93057)  point2 = null | Відображаєтся точка на мапі з координатами point1 | | FR1 | TC4 | point1 = null  point2 = point(82.93057,82.93057) | Відображаєтся точка на мапі з координатами point2 | | FR1 | TC5 | point1 = null  point2 = null | Відображаєтся мапа города | |
| Для функції SendPoint   |  |  |  |  | | --- | --- | --- | --- | | FR ID | Test  Case  ID | Опис значень вхідних даних  до методу (процедури, функції) | Опис очікуваних значень  результату виконання методу | | FR2 | TC1 | point1 = null  point2 = null | Відправляєтся порожній запит на send() | | FR2 | TC2 | point1 = point(82.93057,82.93057)  point2= point(102.93057,102.93057) | Відправляєтся запит з точками point1 и point2 на send () | | FR2 | TC3 | point1 = point(82.93057,82.93057)  point2 = null | Відправляєтся запит з точкою point1 на send () | | FR2 | TC4 | point1 = null  point2 = point(82.93057,82.93057) | Відправляєтся запит з точкою point2 на send () | |

Та Unit Tests

В котрих було перевірено коректність обробки даних у методі Send, бо він є головним зі сторони сервера

Изображение выглядит как текст

Автоматически созданное описание

В юніт тестах перевірялась, як буде поводити себе метод при надані різних значень точок як коректних так і некоректних, що виходять за діапазон значень можливих параметрів точок на карті, тобто широти та довготи

Изображение выглядит как текст

Автоматически созданное описание

Результат тестування

# Висновки

Був розроблена частина FrontEnd та відправлення даних до серверу, додатка Gator, що повинен агрегувати дані про можливі шляхи із місця в місце. Це було досягнено засобами Visual Studia, мови C#, мови JavaScript, Api, що предоставляло мапу та знанями отриманими за навчання на курсі Веб технології та веб дизайн