Міністерство освіти і науки України
Український державний університет науки і технологій

Факультет «Комп'ютерні технології і системи»

Кафедра «Комп'ютерні інформаційні технології»

# Пояснювальна записка

до кваліфікаційної роботи бакалавра

на тему: «Розробка програмного забезпечення для генерації дорожніх знаків»
за освітньою програмою: «Інженерія програмного забезпечення»
зі спеціальності: «121 Інженерія програмного забезпечення»
Виконав: студент групи «ПЗ1911»

_____ /Максим КРИЖАНОВСЬКИЙ/
(підпис студента) (Ім'я ПРІЗВИЩЕ)

Керівник: _____ /доц. Вадим АНДРЮЩЕНКО/
(підпис) (посада, Ім'я ПРІЗВИЩЕ)

Нормоконтролер: _____ /доц. Світлана БОЛКОВА/
(підпис) (посада, Ім'я ПРІЗВИЩЕ)

Консультанти:
(назва розділу) (підпис) (посада, Ім'я ПРІЗВИЩЕ)

Засвідчую, що у цій роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Дніпро – 2023 рік

Ministry of Education and Science of Ukraine
Ukrainian State University of Science and Technologies

Faculty «Computer technologies and systems»
Department «Computer information technology»

# Explanatory Note

to Bachelor's Thesis

on the topic: «Development of software for road signs generation»
according to educational curriculum «Software engineering»
in the Speciality: «121 Software engineering»

Done by the student of the group PZ1911: /Maksym KRYZHANOVSKYI/

Scientific Supervisor: /Vadym ANDRIUSHCHENKO/

Normative controller: /Svitlana VOLKOVA/

Dnipro – 2023

Міністерство освіти і науки України
Український державний університет науки і технологій

Факультет: «Комп'ютерні технології і системи»
Кафедра: «Комп'ютерні інформаційні технології»
Рівень вищої освіти: бакалавр
Освітня програма: «Інженерія програмного забезпечення»
Спеціальність: «121 Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри КІТ
_____/Вадим ГОРЯЧКІН/
(підпис)
Дата_____

# ЗАВДАННЯ

на кваліфікаційну роботу бакалавра
студенту Крижановському Максиму Ігоровичу
1. Тема роботи: «Розробка програмного забезпечення для генерації дорожніх знаків»
Керівник роботи: Андрющенко Вадим Олександрович, доцент
затверджені наказом № 1209 ст від 07.12.2022
2. Строк подання студентом роботи: __.__.202_ р.
3. Вихідні дані до роботи: _____
_____
4. Пояснювальна записка, зміст (питання, для опрацювання):
Для отримання якісної програми треба зібрати вимоги та ознайомитися з предметною областю, знайти та оглянути аналоги і визначити яку програму потрібно розробити.
Спроектувати систему, яка буде виконувати зазначену задачу, визначити сутності необхідні для реалізації програмного додатка, розробити інтерфейс користувача.
Обрати мову програмування на котрій буде реалізовано додаток та розробити програму, що реалізує проект системи.
Протестувати та налагодити розроблену програму.
Висновки щодо роботи.

5. Перелік графічного матеріалу:
Маршрутне орієнтування на дорогах України: системний путівник, зображення дорожніх знаків, знімки процесу виконання додатку.

# КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів кваліфікаційної роботи | Строк виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Збір та аналіз вимог | 15.02.23 | |
| 2 | Зовнішнє проектування | 01.03.23 | |
| 3 | Внутрішнє проектування | 15.03.23 | |
| 4 | Розробка алгоритмів | 01.04.23 | |
| 5 | Розробка програмного забезпечення | 01.05.23 | |
| 6 | Тестування програмного забезпечення | 20.05.23 | |
| 7 | Подання кваліфікаційної роботи до кафедри | 10.06.23 | |
| 8 | Захист кваліфікаційної роботи на засіданні Екзаменаційної комісії | 26.06.23 | |

Студент _____ _Максим КРИЖАНОВСЬКИЙ_
(підпис) (Ім'я ПРІЗВИЩЕ)

Керівник роботи _____ _доц. Вадим АНДРЮЩЕНКО_
(підпис) (Ім'я ПРІЗВИЩЕ)

## РЕФЕРАТ

Пояснювальна записка складається з 8 розділів:

– вступ – у даному розділі описується сутність розробки, її актуальність. Складається з 1 сторінки;

– збір вимог до програмного забезпечення – у цьому розділі описуються аналоги програми та література стосовно даної предметної області, а також проводиться опитування зацікавлених сторін для формування найбільш повної картини вимог для цієї програми. Складається з 10 сторінок;

– проектування програми та інтерфейсу – у цьому розділі проведений огляд даних на вхід та даних, які отримуємо на виході, формалізація задачі, розробка моделі проекту, приводиться опис об’єктно-орієнтованого проектування, проектування інтерфейсу користувача, ескізи форми, аналіз проекту. Складається з 18 сторінок;

– розробка програмного забезпечення – включає у себе: вибір мови програмування та розробку алгоритмів необхідних для реалізації проекту. Складається з 4 сторінок;

– тестування та налагодження програми – включає у себе: вибір стратегії тестування, опис тестів за допомогою метода «білої» скриньки. Також аналіз помилок, їх вплив на систему та вирішення проблеми. Складається з 12 сторінок;

– висновки. Складається з 1 сторінки;

– список літератури – включає у себе бібліографічний список використаної літератури. Складає 2 сторінки;

– додатки – містить текст програми.

Кількість таблиць: 21 одиниця.

Кількість рисунків: 27 одиниць.

Ключові слова: SVG, PDF, PNG, VUE, ДЗІП.

**ЗМІСТ**

## ПЕРЕЛІК УМОВНИХ ОЗНАК, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

SVG – (Scalable Vector Graphics) масштабована векторна графіка, що базується на XML та використовується для відображення двомірної векторної графіки, як статичної, так і анімованої та інтерактивної.

PDF – (Portable Document Format) формат файлу, створений для представлення двовимірних документів у незалежному від пристрою виведення та роздільної здатності вигляді.

PNG – (Portable Network Graphics) растровий формат збереження графічної інформації, що використовує стиснення без втрат;

VUE – фреймворк JavaScript, використовується для досягнення реактивності у веб-застосунках;

ДЗІП – Дорожні знаки індивідуального проектування.

# ВСТУП

Метою розробки цього проекту є перехід держави на використання вітчизняного програмного забезпечення для конструювання схем дорожніх знаків (замість російського). Також, державою були розробленні нові ДСТУ з оформлення дорожніх знаків, тому ці документи були використані при формуванні вимог до проекту. Додатковою метою цього проекту є пришвидшення механізму розробки та виготовлення схем для друку знаків, зменшення витрат коштів на їх розробку та створення схем за чіткими правилами та концептами. Для простої підтримки програмного забезпечення та його функціонування у будь-якому місці де є зв'язок з мережею інтернет та інтернет-браузер, було обранно веб-застосунок без серверной частини.

Програма має використовувати різні типи дорожніх напрямів для формування макету. Дорожні напрями будуть узяті з документів та розбиті на окремі елементи, після чого використовуватимуться для конструювання цілого знаку. Увесь знак повинен використовувати векторний формат графіки для того, щоб знак виглядав однаково у будь-якому розмірі. Також, до дорожніх напрямів можна буде додавати назву напряму та різні дорожні знаки. Для пристосування знаків до різних місцевостей та оточення треба мати декілька кольорових палітр, наразі, вони вже зазначені у ДСТУ. Для виведення та збереження цих макетів використовуватимуться формати PNG та PDF. Програма повинна мати спосіб збереження прогресу (роботи зі знаком) та спосіб продовжити роботу зі знаком, який був збережений. Для тестування нових та старих функцій програми, а також, для зручності користувача повинна бути система заготовок (заздалегідь зробленних знаків). Алгоритм програми повинен автоматично підлаштовувати розмір знаку під його вміст.

# 1 ЗБІР ТА АНАЛІЗ ВИМОГ

Аналіз вимог пов’язаний з цілями й потребами замовника. Під час аналізу вимог слід визначити:

- спосіб використання ПЗ;
- вхідні та вихідні дані;
- спосіб зображення інформації;
- наскільки продуктивним повинен бути додаток;
- надійність додатка;
- інструменти для реалізації проекта.

Тобто, за результатом виконання збору та аналізу вимог повинно вирішити, що буде виконувати програмний продукт, як це повинно виглядати та за допомогою яких інструментів реалізовано.

## 1.1 Огляд аналогів

В ході дослідження виявлено лише один аналог генератора дорожніх знаків – **Sign Maker.** [1]

Цей застосунок був знайдений під час пошуку аналогів у мережі інтернет, він є доволі простий та не виконує всі вимоги заданого проекту, але надає кілька корисних підходів та рішень для проекту.

**Sign Maker –** Застосунок на JavaScript для створення дорожніх знаків. Розроблений “Kurumi”. Дозволяє створювати дорожні знаки у стилі визначеному Федеральним управлінням автомобільних доріг США з використанням Посібника з уніфікованих засобів регулювання дорожнього руху.

На рис. 1.1. зображений дорожній знак, якій буде змінюватись впродовж введення нової інформації та форма для змінення саме цієї інформації.

На рис. 1.2. зображений кінцевий дорожній знак, який використовує введену інформацію користувача.


Рисунок 1.1 - Початковий екран

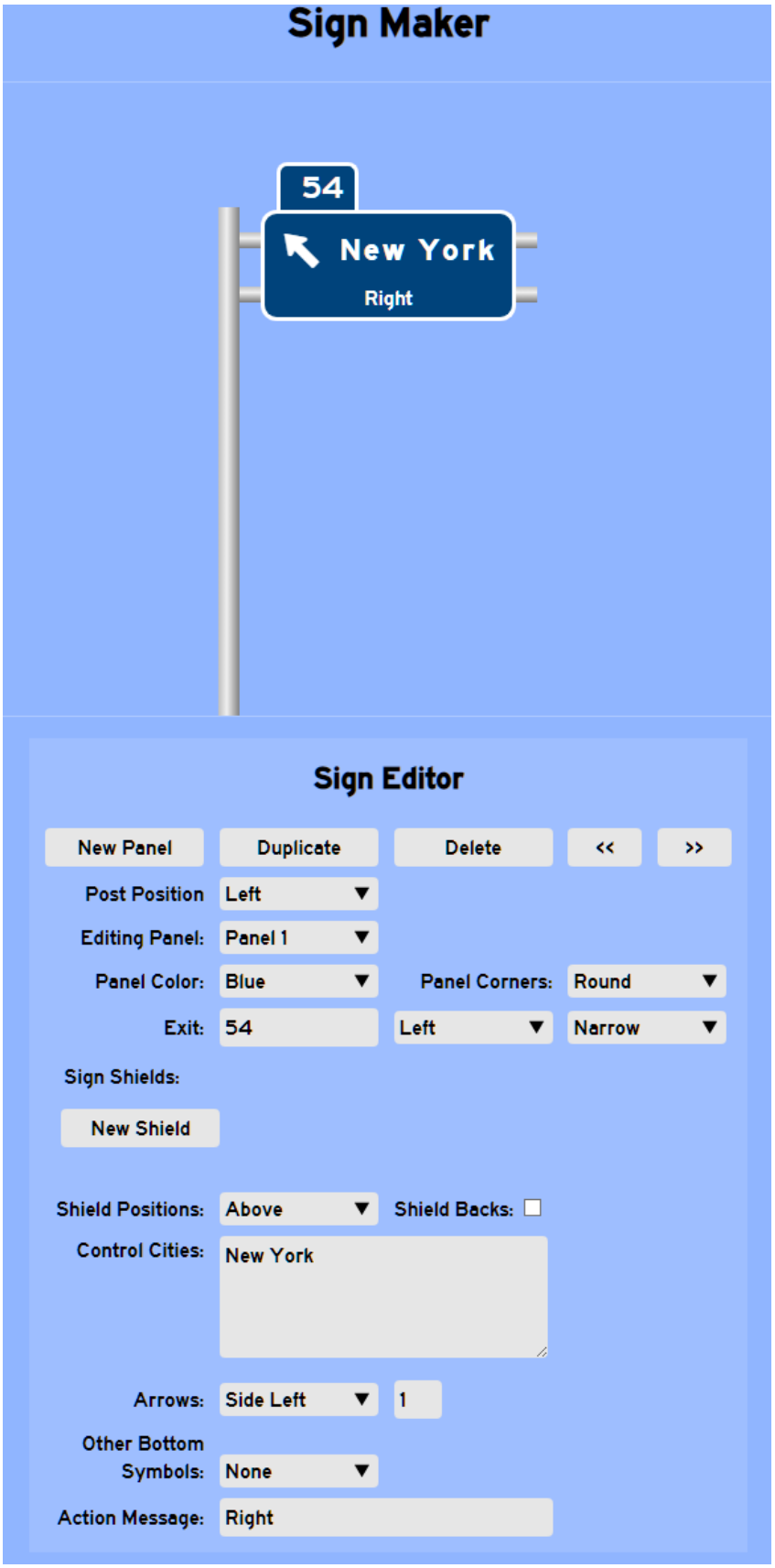


Рисунок 1.2 - Кінцевий знак

## 1.2 Огляд літератури

Далі розглянемо які бувають дорожні знаки та як їх проектують.

### 1.2.1 Схема (знак 5.51)

Схема — це схематичне зображення конфігурації перетину доріг у вигляді тих маневрів, які має виконати водій.

На схемі відображають не більше ніж сім дестинацій і, зазвичай, не більше, ніж дві для кожного з напрямів. Населені пункти для кожного напрямку обирають за наступним правилом:

1. населений пункт із найвищим статусом далі цією дорогою, окрім кінцевого чи найвіддаленішого;
2. кінцевий (найвіддаленіший).

Якщо напрямків більше ніж три, для деяких з них доводиться вказувати лише один населений пункт (починаючи з найменш пріоритетних напрямків), тоді вказують населений пункт із найвищим статусом або найближчий з тих, що мають найвищі статуси. [2]


Рисунок 1.3 - Схема перехрещення двох доріг


Рисунок 1.4 - Схема перехрещення двох доріг з відокремленим з'їздом праворуч

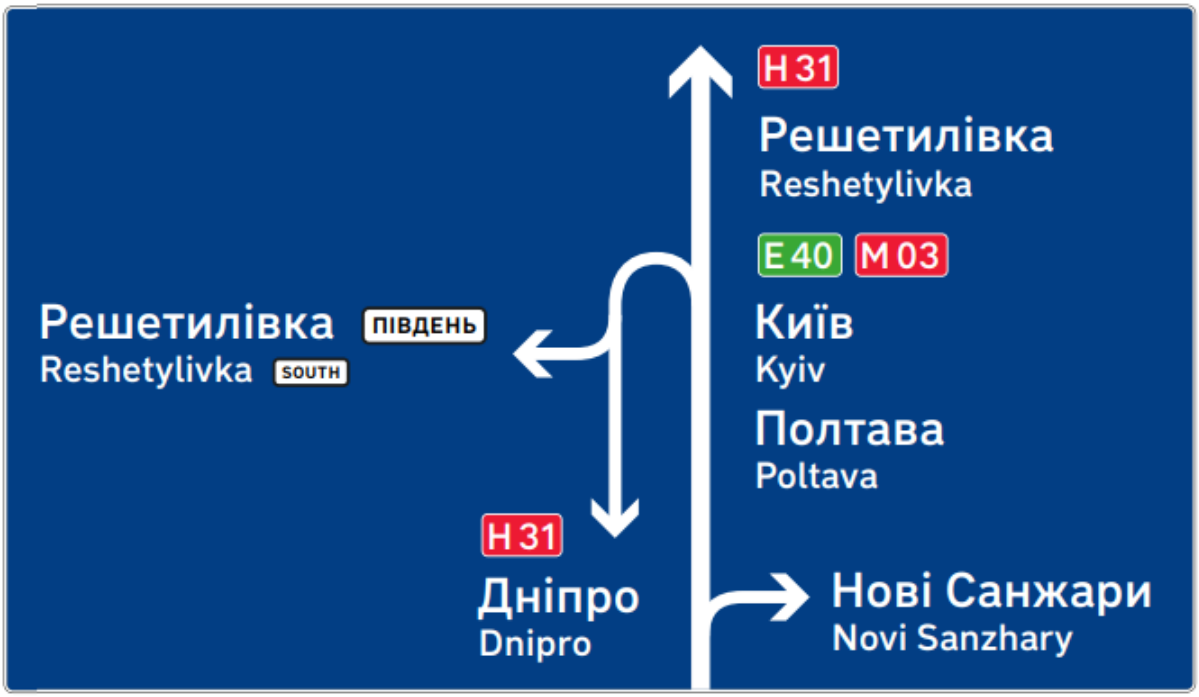


Рисунок 1.6 - Схема з трьома дестинаціями: Решетилівка, Київ та Полтава

### 1.2.2 Текст на схемі знаку

Текст розміщується біля стрілок на відстанях 5х для великої стрілки та 4х для малої стрілки. Для напрямків праворуч і ліворуч текст потрібно розміщувати в напрямку руху (навпроти стрілки), верхній край стрілки вирівнюється за краєм великої літери. У разі розташування написів ліворуч від стрілки відстань вимірюється до границі блоку тексту (рис 1.7). [2]

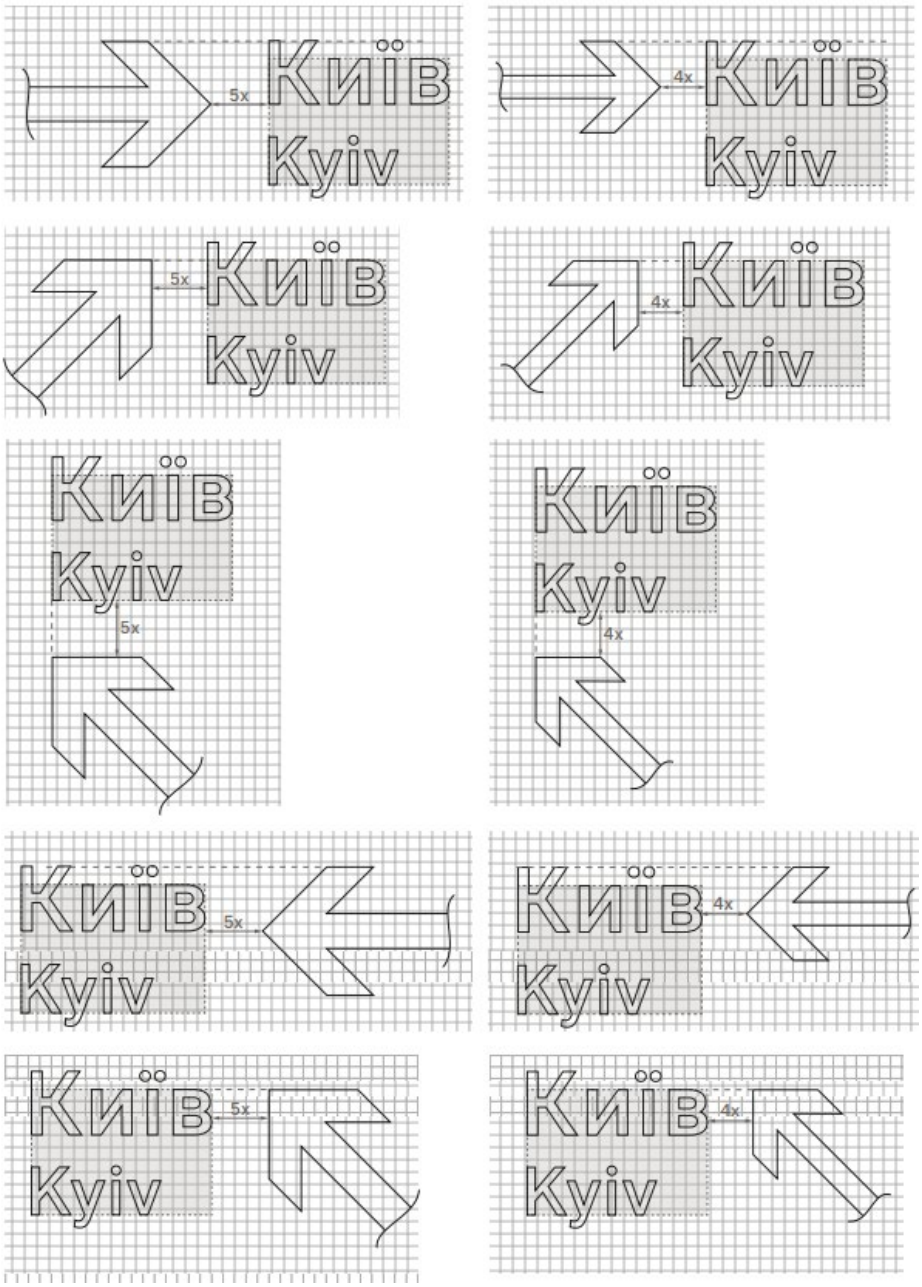


Рисунок 1.7 - Відстань розміщення тексту з різних боків від стрілки

### 1.2.3 Рекомендації при розробці елементів схеми

Рекомендовано елементи схем (повороти, перетини, згини та інші маневри) робити загальними розмірами кратними 4х, а відстань від основної лінії, не меншою ніж 8х (приклади будови деяких маневрів на рисунках 1.8 та 1.9). [2]

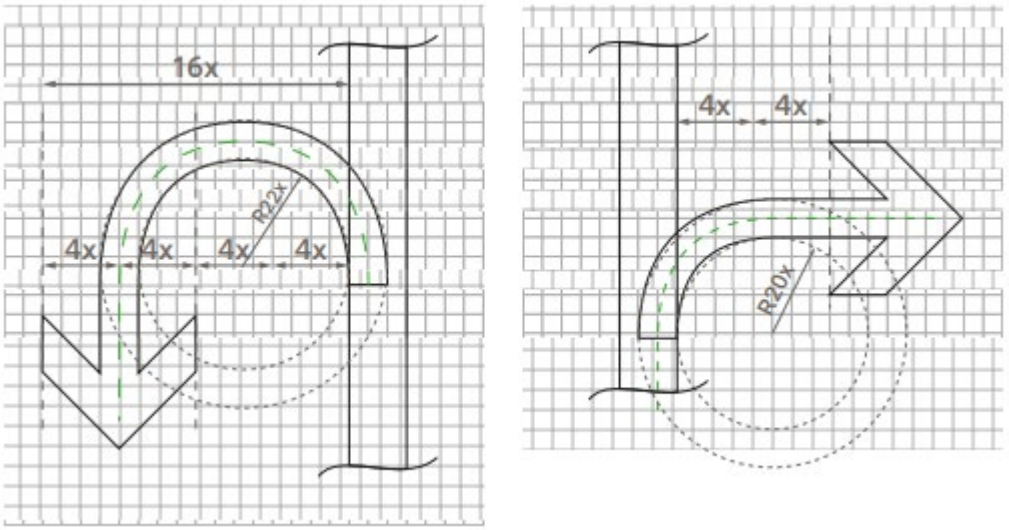


Рисунок 1.8 - Поворот праворуч та розворот

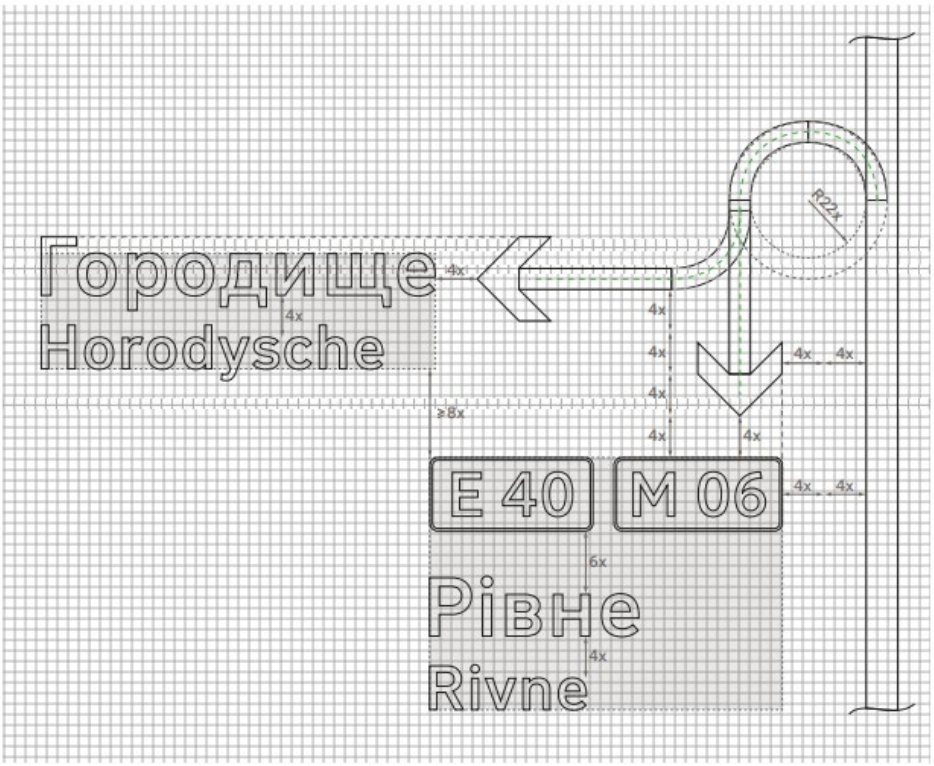


Рисунок 1.9 - Розворот та віднесений поворот ліворуч

### 1.2.4 Рекомендації при визначенні відступів

Відстань від облямівки до елементів знаку — 6x. Вона більша ніж на знаках напрямків, оскільки схеми мають велику площу, і, для кращої читабельності написів, потрібна більша відстань від країв.

Відстань між блоками різних напрямків має бути більшою ніж 8x (більшою, ніж між елементами всередині блоку). Для кращої естетики відстані потрібно робити кратними 4x. [2]

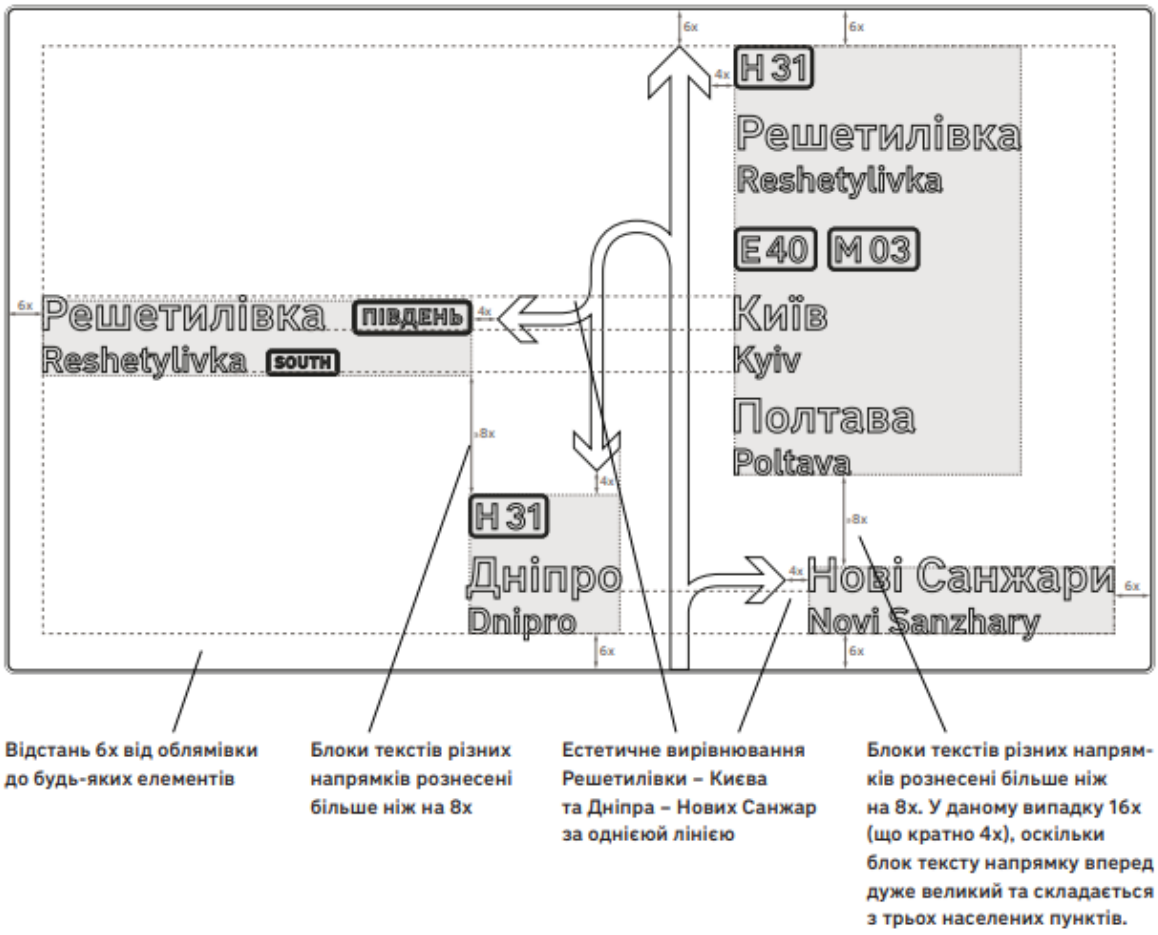


Рисунок 1.10 - Відстані на схемах

## 1.3 Опитування зацікавлених сторін

Метод опитування – психологічний вербально-комунікативний метод, що полягає в здійсненні взаємодії між інтерв'юером і опитуваними (респондентами) з метою одержання від суб'єкта відповідей на заздалегідь сформульовані запитання. Іншими словами, опитування являє собою спілкування інтерв’юера і респондента, у якому головним інструментом виступає заздалегідь сформульоване питання.

Респондент – замовник.

### 1.3.1 Перелік питань

Далі йде перелік питань, що необхідні для конкретизації задачі:

1. Який повинен бути тип застосунку веб чи десктоп?
2. Яким документом користуватись при розробці (Якщо такий є)?
3. У яких форматах повинен зберігатись кінцевий знак?
4. Чи повинна програма підтримувати декілька кольорових моделей (RGB, CMYK та ін.)?
5. Знак має статичний розмір чи він повинен змінюватись?
6. Використовувати растрову чи векторну графіку?

### 1.3.2 Відповіді респондента

Далі йде перелік відповідей респондента на поставлені питання:

1. Веб-застосунок без серверної частини.
2. Дотримуватися документа “Маршрутне орієнтування на дорогах України: системний путівник”.
3. Потрібно збереження кінцевих знаків у форматах: PDF та PNG.
4. Тільки RGB.
5. Знак повинен змінювати розміри залежно від наповнення.
6. Бажано використовувати векторну графіку для збереження якості зображення у будь-якому розмірі.

## 1.4 Постановка задачі

Потрібно розробити веб-застосунок з графічним інтерфейсом, що буде конструювати дорожній знак покладаючись на вхідні дані користувача. Програма повинна підлаштовувати розмір знаку дивлячись на його вміст та розміщення, змінюватись динамічно після будь-якої зміни у вхідних даних. Також, програма повинна мати збереження прогресу для продовження конструювання іншим разом, а по завершенні конструювання знака - зберігати готову модель у зазначених форматах.

**Висновки до розділу 1**

Під час збору вимог було оглянуто та обрано план для розробки генератора дорожніх знаків, визначено тип застосунка та затверджено усі методи, які необхідні для розробки елементів знаку. Також, були зібрані всі дані, що знадобляться користувачам для входу.

Найбільш корисними джерелами інформації були огляд літератури та опитування зацікавлених сторін. Саме користуючись цими методами приймалося рішення, щодо того, який застосунок повинен бути та які задачі він має виконувати.

# 2 ЗОВНІШНЄ І ВНУТРІШНЄ ПРОЕКТУВАННЯ

## 2.1 Зовнішнє проектування

### 2.1.1 Функціональне призначення

Програмний продукт, що розробляється, призначений для конструювання схем дорожніх знаків на основі введених даних та нового документа ДСТУ. Застосунок дозволить конструювати знак з обраних елементів для різних напрямків, а також, змінювати розмір знаку стосовно його вмісту, мати декілька кольорових схем та зберігати готовий проект у форматі PDF [3] та PNG [4]. Також, надасть можливість зберегти зроблену схему для продовження роботи іншим разом.

### 2.1.2 Експлуатаційне призначення

За допомогою даного програмного продукту у працівників з’явиться можливість швидко та коректно розробляти схеми дорожніх знаків, що пришвидшить проектування, зменшить витрати коштів на ручну розробку та гарантує дотримання правил при розробці схем. Працівники можуть користуватись цією програмою на будь-якому пристрої, що надає доступ до мережі інтернет та має інтернет-браузер з можливістю підтримки HTML [5], CSS [6], JavaScript [7].

### 2.1.3 Вимоги до функціональних характеристик

Повинна бути можливість задавати:

- Головний напрям;
- Побічні напрями;
- Назви напрямів та різні дорожні знаки до них;
- Змінювати кольорові схеми;
- Масштаб відображення, якщо знак виходить за межі вебсторінки.

Також необхідна можливість додавати декілька побічних напрямів та видаляти зайві елементи на схемі, якщо виникне потреба.

Програмний продукт повинен зображати на вебсторінці схему знаку, яка змінюється у реальному часі від додавання чи змінення даних користувачем. У процесі конструювання схеми користувач повинен мати можливість переміщати побічні напрямки угору чи низ та дорожні знаки у будь-якому напрямку. Також, схема

повинна автоматично підлаштовуватися до її вмісту та збільшуватись чи зменшуватись залежно від потреби.

Процес конструювання схеми знаку:

1. Вибір головного напряму;
2. Вибір побічних напрямків;
3. Зсув побічних напрямків за потреби;
4. Додавання тексту та знаків до напрямків;
5. Вибір кольорової схеми;
6. Вибір формату експорту при завершенні конструювання.

### 2.1.4 Вхідні дані

Вхідними даними є:

1) sign.json - файл, що зберігає всю інформацію про схему знаку:
    a) createdSign - масив об’єктів, що зберігає у собі всі елементи схеми, їх ідентифікаційний номер, номер елементу з яким він пов'язаний, назву, позицію та розміри.
    b) noShiftSign - копія createdSign, яка виконує роль контролю за зсувом позицій елементів схеми.
    c) colorTheme - змінна, яка зберігає номер обраної кольорової схеми.
    d) viewBoxScale - змінна, яка зберігає масштаб відображення схеми знаку.
2) colorTheme - випадаюче меню, у якому користувач обирає одну з запропонованих кольорових схем. Кольорова схема змінюється одразу після обрання.
3) shapeMainDropVal - випадаюче меню, у якому користувач обирає один з запропонованих головних напрямів руху. Відображення головного напрямку відбувається одразу після обрання з допомогою функцій shapeMainSelect.
4) shapeSubDropVal - випадаюче меню, у якому користувач обирає один з запропонованих побічних напрямів руху.

a) addSubShape - кнопка, яка додає обраний побічний напрям до createdSing у sign.json.

5) viewBoxScale - випадаюче меню, у якому користувач обирає один з запропонованих масштабів відображення схеми знаку. Зміна масштабу відбувається одразу після обрання.
6) signPresets - випадаюче меню, у якому користувач обирає один з запропонованих шаблонів схеми знаку.
    a) importFormPreset - кнопка, яка застосовує обраний шаблон та змінює схему знаку.
7) fileUpload - файловий ввід, який приймає файл користувача та зберігає у собі для подальшого використання.
    a) importFromJsonFile - кнопка, функція імпортує інформацію надану у файлі формату JSON [8] та змінює схему знаку відповідно даним у файлі.
8) deleteAll - кнопка, що повертає усе у початковий стан.
9) shapeAddDropVal - випадаюче меню яке з’являється при натисканні на будь-який елемент схеми, у якому користувач обирає один з запропонованих додаткових елементів до обраного напрямку, як текст або знаки дорожнього руху.
    a) textPosPick - випадаюче меню яке з’являється при виборі опцій “текст” у випадаючому меню “shapeAddDropVal”, це меню пропонує декілька варіантів розміщення для елемента “Text”.
    b) addAdditionShape - кнопка, яка додає обраний додатковий елемент до createdSing у sign.json та пов’язує його з обраним напрямом.
10) deleteShape - кнопка яка з’являється при натисканні на будь-який елемент схеми. При натисканні видаляє обраний елемент схеми.

### 2.1.5 Вихідні дані

Вихідними даними є:

1) JSON файл зі збереженням роботи - який отримується при натисканні на кнопку, функція “exportToJsonFile” зчитує інформіцію з sign.json та

зберігає інформацію у новому файлі формату JSON, після чого, зберігає цей файл на комп'ютері користувача.

2) PNG зображення схеми знаку - яке отримується при натисканні на кнопку, функція “pngConvert” копіює готову схему знаку у іншій контейнер SVG де конвертує увесь текст у криві, після чого робить знімок та завантажує файл формату PNG на комп’ютер користувача.
3) PDF схеми знаку - який отримується при натисканні на кнопку, функція “pathConvert” копіює готову схему знаку у іншій контейнер SVG, де конвертує увесь текст у криві, після чого, копіює усю векторну графіку, записує її у PDF файл та завантажує цей файл на комп’ютер користувача.

Виконаємо специфікацію функціональних вимог у вигляді прецедентів (рис. 2.1).


Рисунок 2.1 – Діаграма прецедентів

### 2.1.6 Опис зовнішнього інформаційного середовища

Схема знаку автоматично оновлюється коли користувач щось змінює або додає.

Вибір елемента схеми виконується натисканням на миші (ЛКМ) після чого, для зорового сприйняття, елемент змінює стиль для відображення обраного стану.

Коли побічний елемент обрано у користувача є можливість змінити його положення затиснувши кнопку (ЛКМ) на обраному елементі та тягнучи його угору чи униз.

Коли обрано знак, користувач може тягнути його у будь-яку сторону.

Пов’язані елементи, як текст та знаки, будуть автоматично слідувати за побічною стрілкою, якщо її буде переміщено.

Залежно від наповнення схеми програма буде автоматично підлаштовувати розмір схеми знаку під вміст.

Для функціонування ПЗ необхідно мати: підключення до мережі інтернет та інтернет-браузер з підтримкою JavaScript.

## 2.2 Внутрішнє проектування

### 2.2.1 Проектування архітектури системи

2.2.1.1 Моделювання словника системи

Ідентифіковані сутності: Головний знак, допоміжний знак, кольорова схема, масштабування, шаблони, збереження/завантаження схеми, збереження знаку.

Ідентифіковані обов’язки:

1) Схема знаку - містить у собі усі елементи для конструювання схеми та відображає їх на вебсторінці користувача.
2) Кольорова схема - елемент, який відповідає за вибір кольору для схеми знака.
3) Масштабування - елемент, який відповідає за масштаб відображення схеми знака.
4) Збереження/Завантаження схеми - модуль, який зберігає стан зробленої схеми у файл, має можливість зчитати файл збереження та продовжити працювати зі збереженою схемою.
5) Збереження знаку - модуль, який зберігає готову схему знаку у PNG чи PDF.

Атрибути та операції, необхідні для виконання обов'язків кожної сутності-класу наведено у табл. 2.1.

Таблиця 2.1 – Сутності, атрибути та методи

| Сутність | Атрибути | Методи |
|---|---|---|
| Схема знаку | **Елементи конструювання знаку:**<br>**Головні напрями:**<br>Straight0d_main,<br>Left90d_Right90d_main,<br>Left90d_Straight0d_Right90d_main.<br>**Допоміжні напрями:**<br>**Праворуч:**<br>Right90d,<br>Right90d_Left180d,<br>Right270d,<br>Right45d,<br>Right180d.<br>**Ліворуч:**<br>Left90d,<br>Left45d,<br>Left180d.<br>**Додаткові елементи:**<br>Text,<br>TruckSign.<br>ColorThemes - обрана кольорова схема.<br>textPositions - дозволені позицій тексту.<br>shapeFocusId - ідентифікаційний номер обраного елементу.<br>scaleVal - обраний масштаб. | cursor - метод, який змінює відображення курсору коли затискають (ЛКМ) на елементі<br>info - повертає ID елементу після кліку на нього та додає стилі на цей елемент для відображення елементу у фокусі.<br>drag - розпочинає процедуру переміщення елемента.<br>move - переміщує елемент.<br>drop - зупиняє переміщення. |

| | | |
|---|---|---|
| Кольорова схема | backgroundColor - колір заднього фону.<br>elementsColor - колір елементів.<br>colorThemeVal - обрана кольорова схема. | |
| Масштабування | scaleVal - обранний масштаб. | |
| Збереження/ Завантаження схеми | files - файл завантажений користувачем.<br>importFileName - назва отриманого файлу. | exportToJsonFile - зберігає стан схеми у JSON файли.<br>importFromJsonFile - зчитує файл користувача та оновлює схему знаку стосовно цього файлу.<br>updateImportFileName - відображає назву завантаженого файлу на вебсторінці |
| Збереження знаку | **Конвертація у криві:**<br>session - змінна з копією SVG знак.<br>**Конвертація у PNG:**<br>fileName - ім’я нового файлу.<br>**Конвертація у PDF:**<br>x - відступ з лівого краю документа.<br>y - відступ з правого краю документа.<br>width - ширина знаку.<br>height - висота знаку.<br>canvasWidth - ширина PDF документу.<br>canvasHeight - висота PDF документу.<br>orientation - орієнтація документу | pathConvert - конвертація шрифта у криві.<br>pngConvert - конвертація SVG у PNG та завантаження отриманого результату на комп’ютер користувача.<br>pdfConvert - конвертація SVG у PDF та завантаження отриманого результату на комп’ютер користувача. |

| | | |
|---|---|---|
| | альбомна чи книжкова.<br>fileName - ім’я нового документу. | |

#### 2.2.1.2 Моделювання розподілу обов’язків

Модулі, які працюють разом задля досягнення бажаного результату:

- Вебсторінка, головний знак - відображення процесу конструювання знаку.
- Вебсторінка, елементи інтерфейсу користувача - отримання значень обраних користувачем для конструювання знаку та команд для виконання іншого функціоналу.
- Головний знак, елементи знаку - конструювання схеми знаку.
- Конвертація у PDF та PNG, конвертація шрифта у криві - конвертація шрифта у криві для правильного відображення тексту у файлах PDF та PNG.
- Головний знак, допоміжний знак - допоміжний знак виконує обчислення для збільшення розмірів знаку та відображає зміну у головному знаку.
- Головний знак, кольорова схема - зміна кольора схеми знаку.
- Головний знак, збереження схеми - зберігає конфігурацію головного знаку.
- Головний знак, завантаження схеми - зчитує та відображає отриману конфігурацію схеми знаку.
- Головний знак, конвертація у PDF та PNG - перетворює головний знак у файл обраного формату та завантажує на комп’ютер користувача.
- Головний знак, масштаб - змінює масштаб відображення знаку на вебсторінці.

#### 2.2.1.3 Моделювання непрограмних сутностей

Ідентифіковані непрограмні сутності:

1) SvgCode - абстрактний модуль, що призначений для конструювання та зображення елементів знаку на вебсторінці;

#### 2.2.1.4 Моделювання примітивних типів, простих залежностей, наслідування та структурних зв’язків

Результат моделювання різних видів зв’язків подано у табл. 2.2.

Таблиця 2.2 – Моделювання залежностей

| Модуль, який зв’язується | Модуль, з яким зв’язуються | Тип зв’язку |
|---|---|---|
| SvgCode | Straight0d_main | Композиція |
| | Left90d_Right90d_main | Композиція |
| | Left90d_Straight0d_Right90d_main | Композиція |
| | Right90d | Композиція |
| | Right90d_Left180d | Композиція |
| | Right270d | Композиція |
| | Right45d | Композиція |
| | Right180d | Композиція |
| | Left90d | Композиція |
| | Left45d | Композиція |
| | Left180d | Композиція |
| | Text | Композиція |
| | TruckSign | Композиція |
| | sign.json | Залежність |
| | shapesTemplates.json | Залежність |
| 5.51-sign | SvgCode | Агрегація |

| | | |
|---|---|---|
| | jsPDF | Реалізація |
| | saveSvgAsPng | Реалізація |
| | sign.json | Залежність |
| | shapesTemplates.json | Залежність |

У таблицях 2.3-2.18 зображені CRC-картки для усіх модулів.

Таблиця 2.3 – CRC-картка для модуля “5.51-sign”

| Базовий модуль | Похідні модулі |
|---|---|
| 5.51-sign | SvgCode |
| Обов’язки | Зв’язки |
| Відображення конструювання знаку та інтерфейсу користувача | SvgCode, sign.json, shapesTemplates.json, jsPDF, saveSvgAsPng. |

Таблиця 2.4 – CRC-картка для модуля “SvgCode”

| Базовий модуль | Похідні модулі |
|---|---|
| SvgCode | Straight0d_main, Left90d_Right90d_main, Left90d_Straight0d_Right90d_main, Right90d, Right90d_Left180d, Right270d, Right45d, Right180d, Left90d, Left45d, Left180d, Text, TruckSign. |
| Обов’язки | Зв’язки |
| Відображення конструювання знаку та інтерфейсу користувача | Straight0d_main, Left90d_Right90d_main, Left90d_Straight0d_Right90d_main, Right90d, Right90d_Left180d, Right270d, Right45d, Right180d, Left90d, Left45d, Left180d, Text, TruckSign, colorThemes, sign.json, shapesTemplates.json. |

Таблиця 2.5 – CRC-картка для модуля “sign.json”

| Базовий модуль | Похідні модулі |
|---|---|
| sign.json | Відсутні |
| Обов’язки | Зв’язки |
| Виступає інформаційним полем знаку, зберігає такі дані як: позиції елементів, ім’я елементів, розмір елементів. | 5.51-sign, SvgCode. |

Таблиця 2.6 – CRC-картка для модуля “shapesTemplates.json”

| Базовий модуль | Похідні модулі |
|---|---|
| shapesTemplates.json | Відсутні |
| Обов’язки | Зв’язки |
| Зберігає базову конфігурацію для кожного елементу знаку. | 5.51-sign, SvgCode. |

Таблиця 2.7 – CRC-картка для модуля “Straight0d_main”

| Базовий модуль | Похідні модулі |
|---|---|
| Straight0d_main | Відсутні |
| Обов’язки | Зв’язки |
| Виступає головною прямою стрілкою у схемі знаку. | SvgCode. |

Таблиця 2.8 – CRC-картка для модуля “Left90d_Right90d_main”

| Базовий модуль | Похідні модулі |
|---|---|
| Left90d_Right90d_main | Відсутні |
| Обов’язки | Зв’язки |
| Виступає головною стрілкою з напрямками ліворуч та праворуч у схемі знаку. | SvgCode. |

Таблиця 2.9 – CRC-картка для модуля “Left90d_Straight0d_Right90d_main”

| Базовий модуль | Похідні модулі |
|---|---|
| Left90d_Straight0d_Right90d_main | Відсутні |
| Обов’язки | Зв’язки |
| Виступає головною стрілкою з напрямками ліворуч, уперед та праворуч у схемі знаку. | SvgCode. |

Таблиця 2.10 – CRC-картка для модуля “Right90d”

| Базовий модуль | Похідні модулі |
|---|---|
| Right90d | Відсутні |
| Обов’язки | Зв’язки |
| Виступає побічною стрілкою з напрямом праворуч у схемі знаку. | SvgCode. |

Таблиця 2.11 – CRC-картка для модуля “Right90d_Left180d”

| Базовий модуль | Похідні модулі |
|---|---|
| Right90d_Left180d | Відсутні |
| Обов’язки | Зв’язки |
| Виступає побічною стрілкою з напрямом праворуч та розворот у схемі знаку. | SvgCode. |

Таблиця 2.11 – CRC-картка для модуля “Right270d”

| Базовий модуль | Похідні модулі |
|---|---|
| Right270d | Відсутні |
| Обов’язки | Зв’язки |
| Виступає побічною стрілкою з розворотом ліворуч у схемі знаку. | SvgCode. |

Таблиця 2.12 – CRC-картка для модуля “Right45d”

| Базовий модуль | Похідні модулі |
|---|---|
| Right45d | Відсутні |
| Обов’язки | Зв’язки |
| Виступає побічною стрілкою з легким поворотом праворуч у схемі знаку. | SvgCode. |

Таблиця 2.13 – CRC-картка для модуля “Right180d”

| Базовий модуль | Похідні модулі |
|---|---|
| Right180d | Відсутні |
| Обов’язки | Зв’язки |
| Виступає побічною стрілкою з розворотом назад з правої сторони у схемі знаку. | SvgCode. |

Таблиця 2.14 – CRC-картка для модуля “Left90d”

| Базовий модуль | Похідні модулі |
|---|---|
| Left90d | Відсутні |
| Обов’язки | Зв’язки |
| Виступає побічною стрілкою з напрямом ліворуч у схемі знаку. | SvgCode. |

Таблиця 2.15 – CRC-картка для модуля “Left45d”

| Базовий модуль | Похідні модулі |
|---|---|
| Left45d | Відсутні |
| Обов’язки | Зв’язки |
| Виступає побічною стрілкою з легким поворотом ліворуч у схемі знаку. | SvgCode. |

Таблиця 2.16 – CRC-картка для модуля “Left180d”

| Базовий модуль | Похідні модулі |
|---|---|
| Left180d | Відсутні |
| Обов’язки | Зв’язки |
| Виступає побічною стрілкою з розворотом назад з лівої сторони у схемі знаку. | SvgCode. |

Таблиця 2.17 – CRC-картка для модуля “Text”

| Базовий модуль | Похідні модулі |
|---|---|
| Text | Відсутні |
| Обов’язки | Зв’язки |
| Виступає додатковим елементом схеми знаку у ролі назви напряму(Текст). | SvgCode. |

Таблиця 2.18 – CRC-картка для модуля “TruckSign”

| Базовий модуль | Похідні модулі |
|---|---|
| TruckSign | Відсутні |
| Обов’язки | Зв’язки |
| Виступає додатковим елементом схеми знаку у ролі дорожнього знаку. | SvgCode. |

Кінцевий результат моделювання у виді діаграми модулів (рис. 2.2). [9]

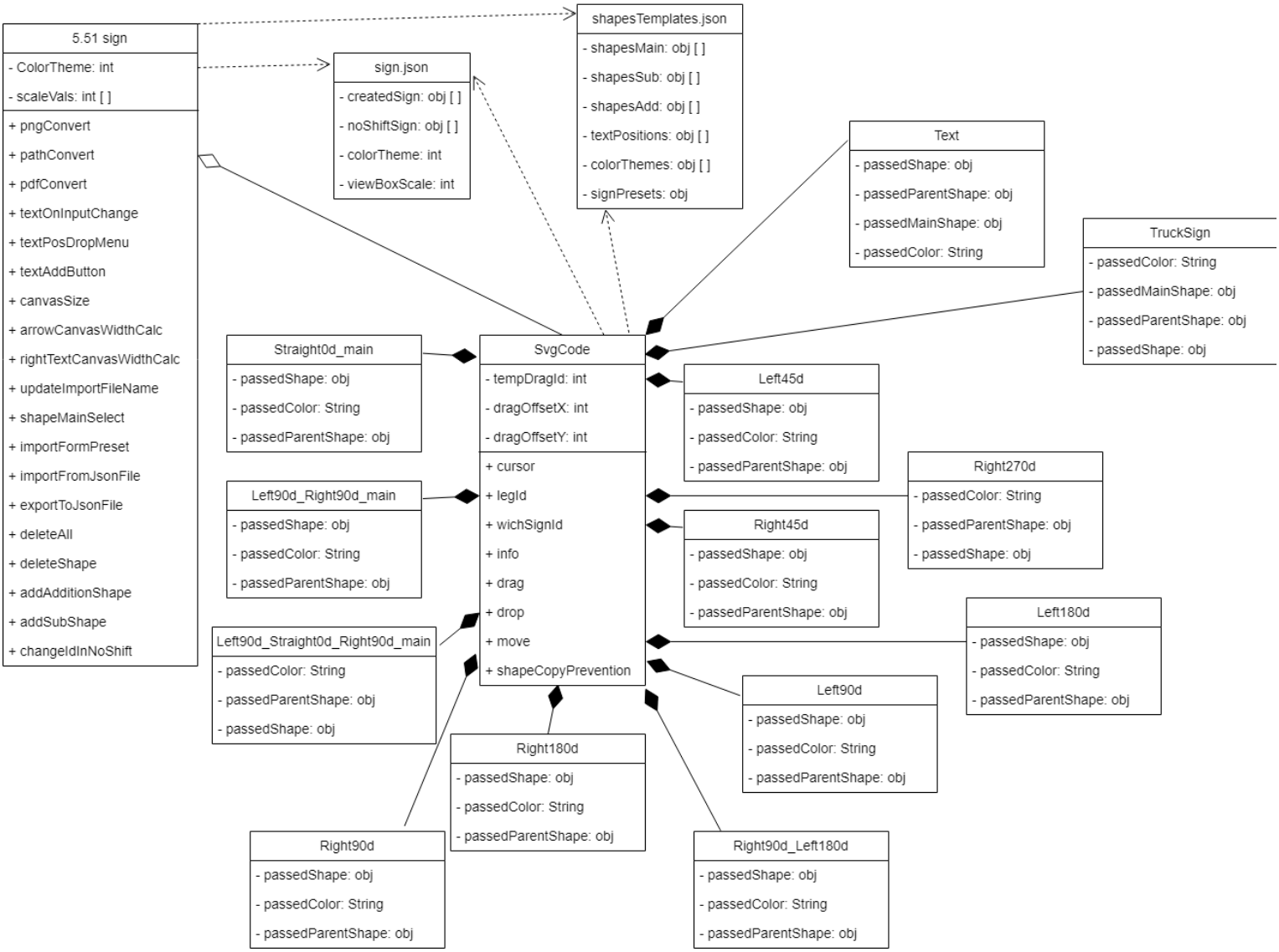


Рисунок 2.2 – Діаграма модулів

### 2.2.2 Проектування інтерфейсу користувача

На вебсторінці потрібно відобразити:

- Конструювання схеми знаку;
- Вибір кольорової схеми;

- Вибір головного напряму;
- Вибір побічного напряму та кнопку для його додавання;
- Вибір масштабу;
- Вибір заздалегідь зробленного шаблону;
- Поле для завантаження файла збереження та дві кнопки, першу для застосування схеми з файлу збереження та другу для отримання нового файлу збереження;
- Дві кнопки для збереження готової схеми у форматах PDF та PNG;
- Кнопка для повернення усього до початкового стану;
- Окреме меню для взаємодії з елементом у фокусі;
    - Вибір додаткового елементу;
    - Якщо текст, то вибір позицій тексту з запропонованих;
    - Кнопка додати обраний елемент;
    - Кнопка видалення обраного елементу.

Визначимо розташування даних та елементів інтерфейсу на вебсторінці (рис. 2.3). На вебсторінці увага приділена зображенню знаку, де буде відбуватися процес

конструювання схеми. Після проектування ескізу був створений макет (рис. 2.4), який відображає місця елементів інтерфейсу на вебсторінці.

Рисунок 2.3 - Ескіз вебсторінки

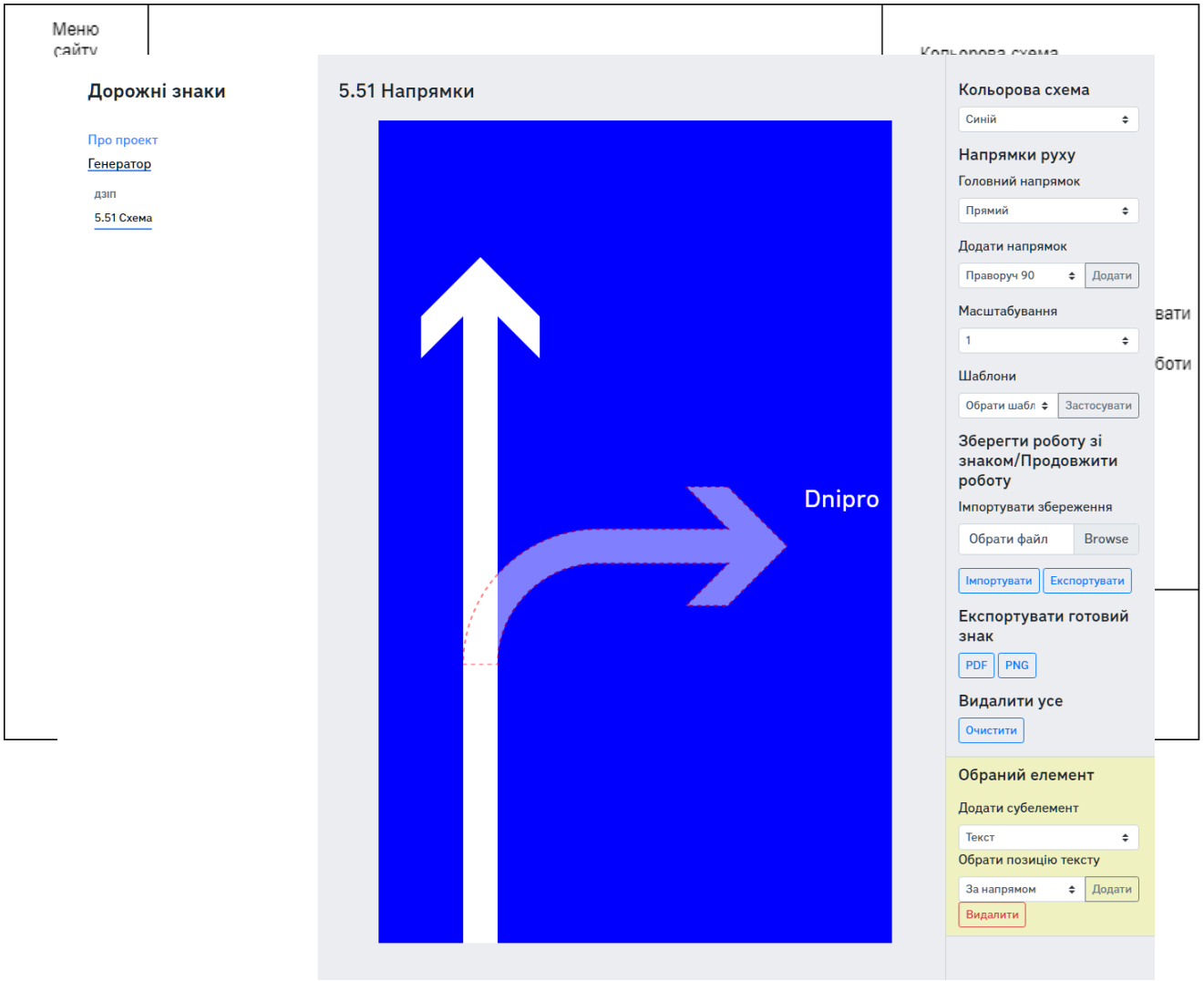


Рисунок 2.4 - Макет розміщення елементів інтерфейсу

### 2.2.3 Проектування динаміки системи

На рис. 2.5 зображена діаграма послідовностей, вона відображає головні можливості ПЗ. Програма є реактивною [10], тобто вона реагує на будь-які зміни у даних та змінюється відповідно одразу.

**Висновки до розділу 2**

Під час зовнішнього проектування було визначено поведінку роботи вебзастосунку, те, як він повинен виконувати поставлену задачу, а саме, отримативати конфігураційні значення від користувача та змінюватися стосовно цих значень якомога швидше для комфортной роботи та швидкого відгуку на зміни.

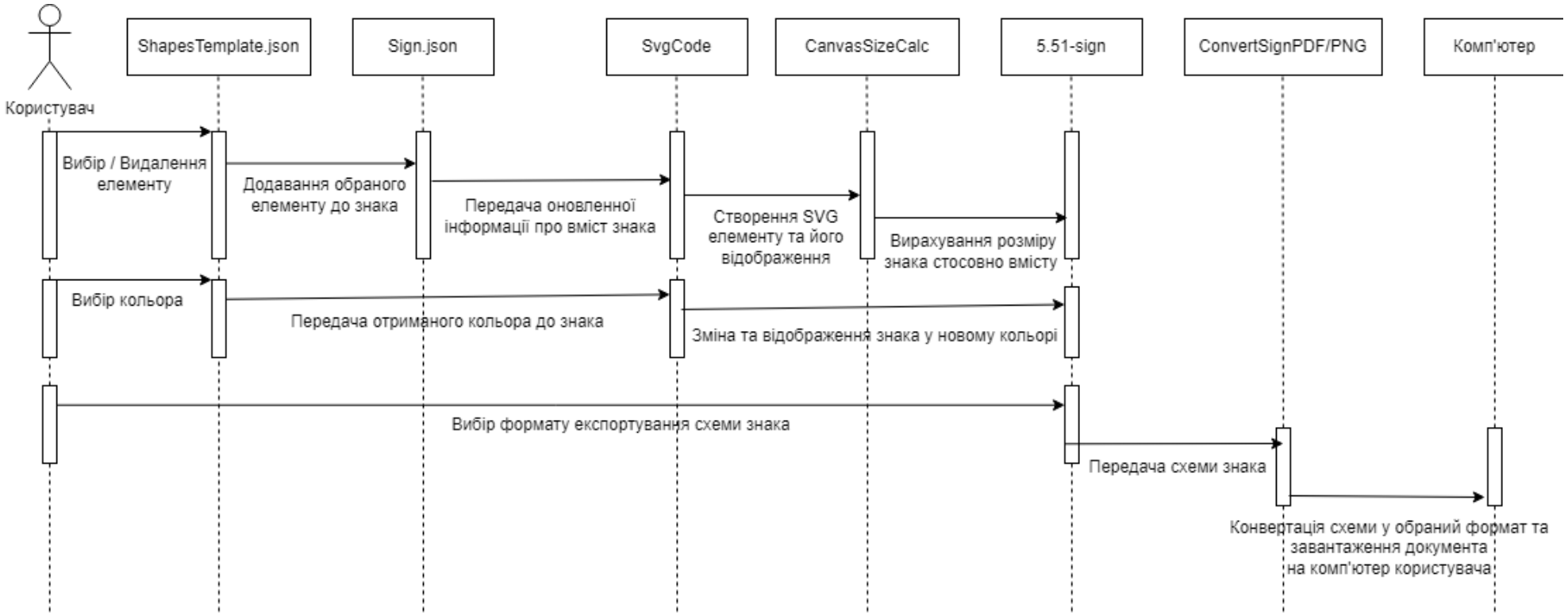


Рисунок 2.5 – Діаграма послідовностей для додавання та видалення елементів, зміни кольору та завантаження готової схеми знаку

# 3 РОЗРОБКА ПРОГРАМИ

## 3.1 Вибір мови програмування

Для реалізації програмного продукту було прийняте рішення обрати мову програмування JavaScript та використовувати фреймворк Vue.js [11]. Так як поставлена задача розробити реактивний вебзастосунок, який будет швидко підлаштовуватися під нові дані та мати змогу працювати на будь-якому пристрої, що підтримує підключення до мережі інтернет та має інтернет браузер, тож, під час вибору мови та фреймворку досить важливим фактором був досвід використання цієї мови та фреймворку на попередньому проекті, який я розробляв продовж виробничої практики.

Для створення схеми знаку використовується SVG - Scalable Vector Graphic [12] це бібліотека векторной графіки, що дозволяє створювати зображення з геометричних фігур, які можна легко масштабувати без втрати якості зображення.

Фреймворк Vue.js був обраний, бо має вбудовану систему реактивності завдяки чому, робити реактивні вебзастосунки легше та займає менше часу. Також, фреймворк дозволяє розбивати код на окремі модулі, які виконують окрему частину роботи, після чого ці модулі можна об’єднувати для роботи разом для вирішення великої задачі.

## 3.2 Розробка алгоритмів

Головним алгоритмом є той, що конструює або змінює знак залежно від дій користувача. Також, є ще декілька допоміжних алгоритмів, які виконують малі, але не менш важливі задачі при користуванні програмою.

Далі зазначені потрібні алгоритми:

1) Додавання елементів у схему знака рис. 3.1. Коли користувач бажає додати новий елемент до схеми, алгоритм повинен зрозуміти який це елемент, знайти базовий опис цього елементу у файлі shapesTemplates.json та додати опис до файлу інформації про схему знака sign.json. Після чого, модуль SvgCode побачить зміни у файлі sign.json та перебудує знак наново з урахуванням нових елементів.

2) Зміна кольорів у схемі знака рис. 3.2. Коли користувач обирає іншу кольорову тему алгоритм шукає цю тему у файлі shapesTemplates.json та записує обраний варіант у файл знаку sign.json. Після чого, SvgCode побачивши зміни у файлі, замінить кольори елементів на нові.

3) Експортування схеми знака у зазначеному форматі рис. 3.3. Коли користувач обирає експортування схеми знаку у PDF чи PNG, алгоритм копіює знак у окремий контейнер, де перекладає увесь текст у криві використовуючи бібліотеку svg-text-to-path [13] для того, щоб зберегти вид шрифта. Після чого, алгоритм використовує сторонні бібліотеки, такі як jsPDF [14] для конвертацій SVG у PDF чи PNG.


Рисунок 3.1 - Додавання елементів у схему знака


Рисунок 3.2 - Зміна кольорів у схемі знака


Рисунок 3.3 - Експортування схеми знака у зазначеному форматі

**Висновки до розділу 3**

Мовою для розробки було обрано JavaScript та фреймворком виступає Vue.js для простої побудови реактивного вебзастосунку. Також, було розроблено декілька алгоритмів, які виконують поставлені задачі.

Було використано декілька сторонніх бібліотек, які спрощують розробку взаємодії з PDF та PNG файлами. Через це можна було приділити більше часу розробці головних модулей та оформленню сайта.

# 4 ТЕСТУВАННЯ ТА НАЛАГОДЖЕННЯ

Для забезпечення коректного виконання програми треба перевірити методи та їх специфікації. Для тестування було взято два методи, які мають непросту логіку та відповідають за важливу частину програми. Ці методи змінюють розмір схеми знаку залежно від вмісту схеми, тим самим обираючи оптимальний розмір схеми. Спираючись на те, що методи не будуть використовуватись у інших проектах чи бібліотеках - використовувати спосіб тестування «чорною» скринькою не є необхідним, через те, що інтерфейс на вебсторінці перевіряє вхідні дані на правильність. [15]

## 4.1 Специфікація методів

Нижче наведено список методів, які будуть проходити тестування:

1) arrowCanvasWidthCalc(rightW, leftW, canvasWidth, sign)

Змінює розмір схеми залежно від кількості та розмірів напрямків у схемі знаку.

Приймає на вхід:

у явному вигляді:

- Розмір найширшого напрямку з правої сторони, тип даних - int
- Розмір найширшого напрямку з лівої сторони, тип даних - int
- Ширина схеми знаку, тип даних - float
- Схема знаку без зсуву, тип даних - obj

у неявному вигляді нічого не приймає;

Повертає на вихід:

у явному вигляді:

- Повертає об’єкт, типу float; float

у неявному вигляді нічого не повертає;

Текст функції:

```
arrowCanvasWidthCalc(rightW, leftW, canvasWidth, sign)
    {
      let arrowPos = 0;
      if(sign.length > 1)
      {
      //No side arrows
        if(rightW == 0 && leftW == 0)
```

```
{
  canvasWidth = sign[1].width + sign[0].width;
  arrowPos = (canvasWidth - sign[0].width) / 2;
}
//Only right arrows
else if(rightW > 0 && leftW == 0)
{
  if(sign[1].multiWay)
  {
    canvasWidth = sign[0].width + (sign[1].width / 2) + rightW;
    arrowPos = (sign[1].width / 2);
  }
  else
  {
    canvasWidth = sign[1].width + rightW;
    arrowPos = sign[1].width - sign[0].width;
  }
}
//Only left arrows
else if(rightW == 0 && leftW > 0)
{
  if(sign[1].multiWay)
  {
    canvasWidth = (sign[1].width / 2) + leftW + sign[0].width;
    arrowPos = canvasWidth - (sign[1].width / 2) - sign[0].width;
  }
  else
  {
    canvasWidth = sign[1].width + leftW;
    arrowPos = canvasWidth - sign[1].width;
  }
}
//Left and Right arrows
else
{
  if(sign[1].multiWay)
  {
    if(rightW > leftW)
    {
      canvasWidth = leftW + rightW + sign[0].width * 2;
      arrowPos = (canvasWidth - sign[0].width - (rightW - leftW)) / 2;
```

```
        }
        else if(rightW < leftW)
        {
          canvasWidth = leftW + rightW + sign[0].width * 2;
          arrowPos = (canvasWidth - sign[0].width + (leftW - rightW)) / 2;
        }
        else
        {
          canvasWidth = leftW + rightW + sign[0].width * 2;
          arrowPos = (canvasWidth - sign[0].width) / 2;
        }
      }
      else
      {
        if(rightW > leftW)
        {
          canvasWidth = sign[1].width + leftW + rightW - sign[0].width;
          arrowPos = (canvasWidth -sign[0].width - (rightW - leftW)) / 2;
        }
        else if(rightW < leftW)
        {
          canvasWidth = sign[1].width + leftW + rightW - sign[0].width;
          arrowPos = (canvasWidth - sign[0].width + (leftW - rightW)) / 2;
        }
        else
        {
          canvasWidth = sign[1].width + leftW + rightW - sign[0].width;
          arrowPos = (canvasWidth - sign[0].width) / 2;
        }
      }
    }
  }

  return {canvasWidth, arrowPos};
}
```

2) rightTextCanvasWidthCalc(textElem, textElemArr, canvasWidth)

Вираховує додатковий розмір для схеми знаку залежний від розміра найдальшого правого тексту.

Приймає на вхід:

у явному вигляді:

- Усі текстові елементи з правої сторони, тип даних масив елементів тексту.
- Усі текстові елементи з правої сторони, тип даних масив інформації про текстові елементи.
- Ширина схеми знаку, тип даних - float

у неявному вигляді нічого не приймає;

Повертає на вихід:

у явному вигляді:

- Оновлену ширину схеми знаку, тип даних - int

у неявному вигляді нічого не повертає;

Текст функції:

```
rightTextCanvasWidthCalc(textElem, textElemArr, canvasWidth)
    {
      for(let i = 0; i < textElemArr.length; i++)
      {
        textElem.push(document.getElementById(textElemArr[i].id));
      }

      let mostRightText = 0;
      for(let i = 0; i < textElemArr.length; i++)
      {
        if(textElemArr[mostRightText].x + textElem[mostRightText].getComputedTextLength() < textElemArr[i].x + textElem[i].getComputedTextLength())
        {
          mostRightText = i;
        }
      }

      if(textElemArr.length > 0)
      {
        if(textElemArr[mostRightText].x + createdSign[0].x + textElem[mostRightText].getComputedTextLength() > canvasWidth)
        {
          let widthDiff = textElemArr[mostRightText].x + createdSign[0].x + textElem[mostRightText].getComputedTextLength() - canvasWidth;
          canvasWidth = canvasWidth + widthDiff + 15;
        }
```

```
    }

    return canvasWidth;
}
```

## 4.2 Тестування методом «білої» скриньки

### 4.2.1 Розроблення тестів

Метод 1(arrowCanvasWidthCalc):

1) Додавання головного напряму

**Вхідні дані:**

rightW - 0, leftW - 0, canvasWidth - 0, sign - noShiftSign.

**Вихідні дані:**

canvasWidth - 187, arrowPos - 72.5.

Збільшення схеми знаку та зсув головного напряму у центр, приклад до виконання наведено у рисунку 4.1 та приклад після виконання у рисунку 4.2.

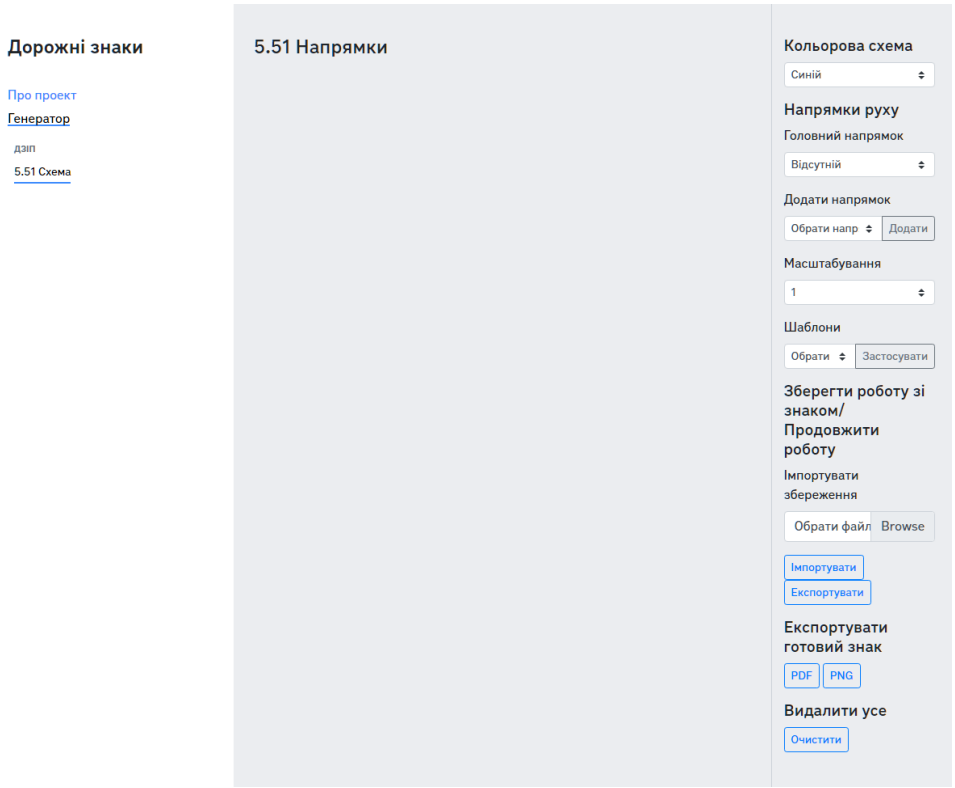


Рисунок 4.1 - Стан схеми до виконання метода

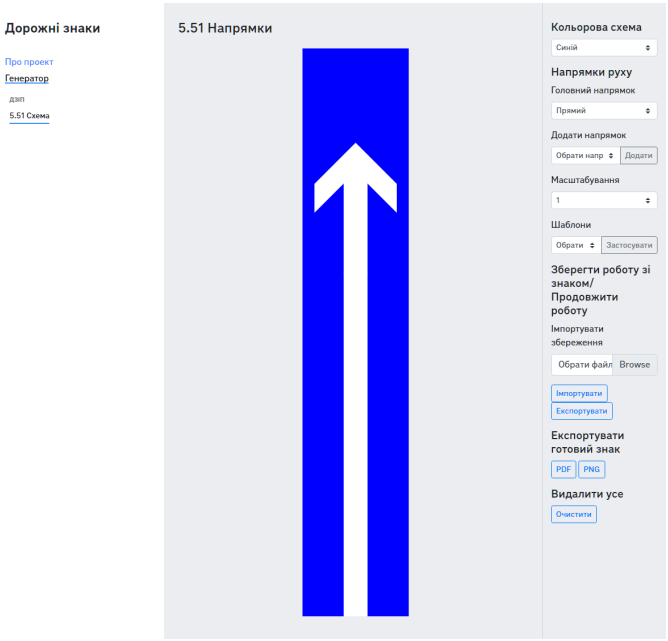


Рисунок 4.2 - Стан після виконання метода

2) Додавання побічного напряму з правої сторони до головного напряму
**Вхідні дані:**
rightW - 395, leftW - 0, canvasWidth - 0, sign - noShiftSign.
**Вихідні дані:**
canvasWidth - 540, arrowPos - 103.
Збільшення схеми знаку та зсув головного напряму у ліву сторону, приклад до виконання наведено у рисунку 4.3 та приклад після виконання у рисунку 4.4.

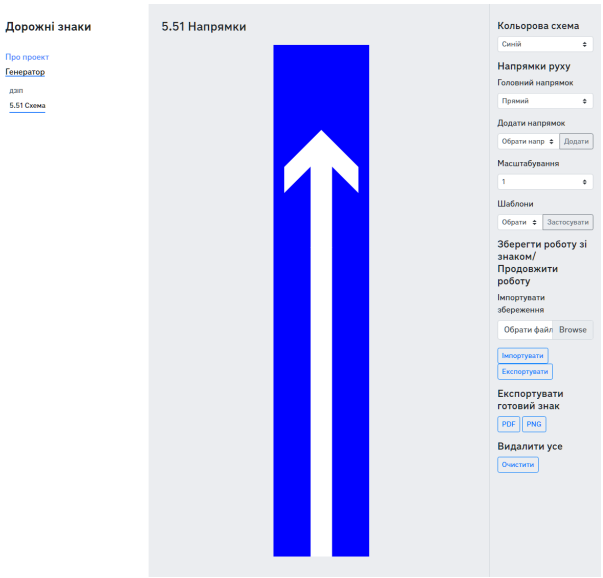


Рисунок 4.3 - Стан схеми до виконання метода

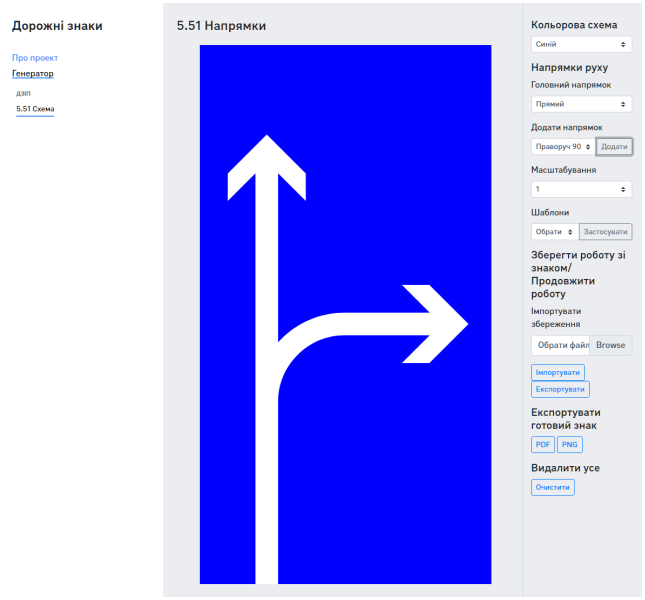


Рисунок 4.4 - Стан після виконання метода

3) Додавання побічного напряму з лівої сторони до схеми з головним напрямом та напрямом з правої сторони

**Вхідні дані:**

rightW - 395, leftW - 375, canvasWidth - 0, sign - noShiftSign.

**Вихідні дані:**

canvasWidth - 873, arrowPos - 405.5.

Збільшення схеми знаку та зсув головного напряму до центру, приклад до виконання наведено у рисунку 4.6 та приклад після виконання у рисунку 4.5.

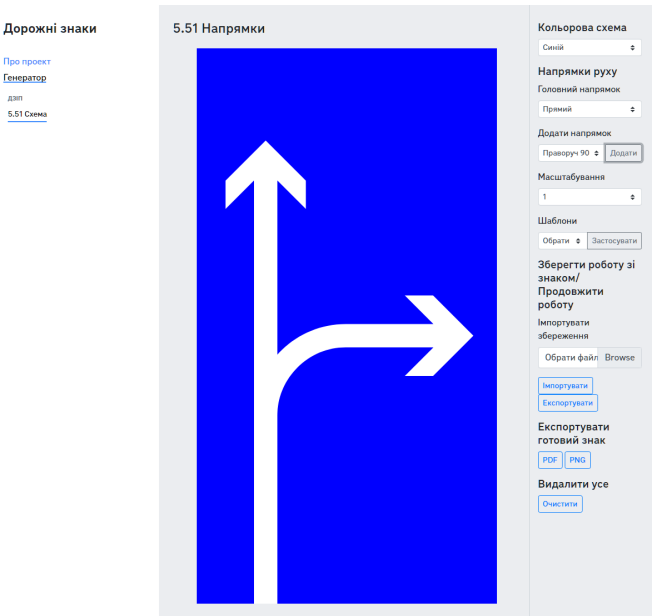


Рисунок 4.5 - Стан схеми до виконання метода

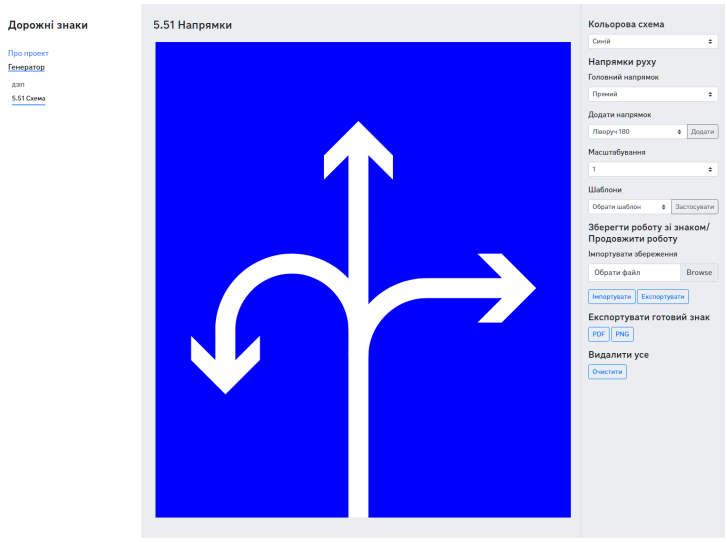


Рисунок 4.6 - Стан після виконання метода

Метод 2(rightTextCanvasWidthCalc):

1) Додавання текстового елементу до напряму

   **Вхідні дані:**

   textElem - один елемент тексту з DOM, textElemArr - опис текстового елементу з sign.json, canvasWidth - 540.

   **Вихідні дані:**

   canvasWidth - 574.984375.

   Збільшення схеми знаку у праву сторону, приклад до виконання наведено у рисунку 4.7 та приклад після виконання у рисунку 4.8.

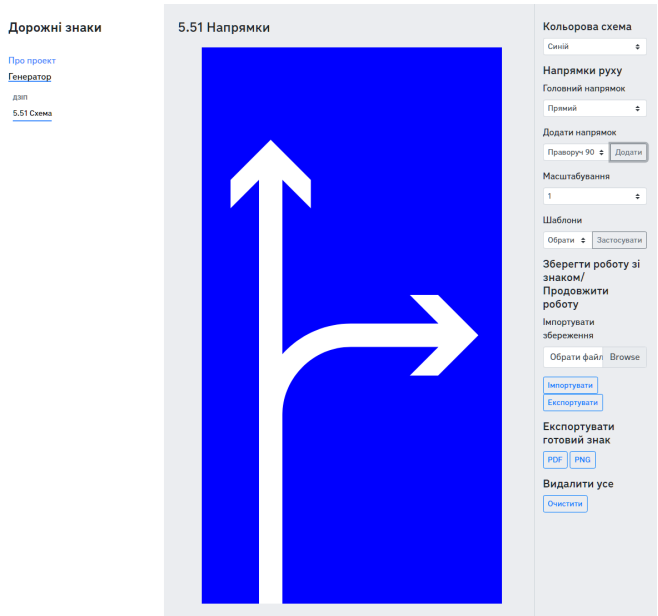


Рисунок 4.7 - Стан схеми до виконання метода

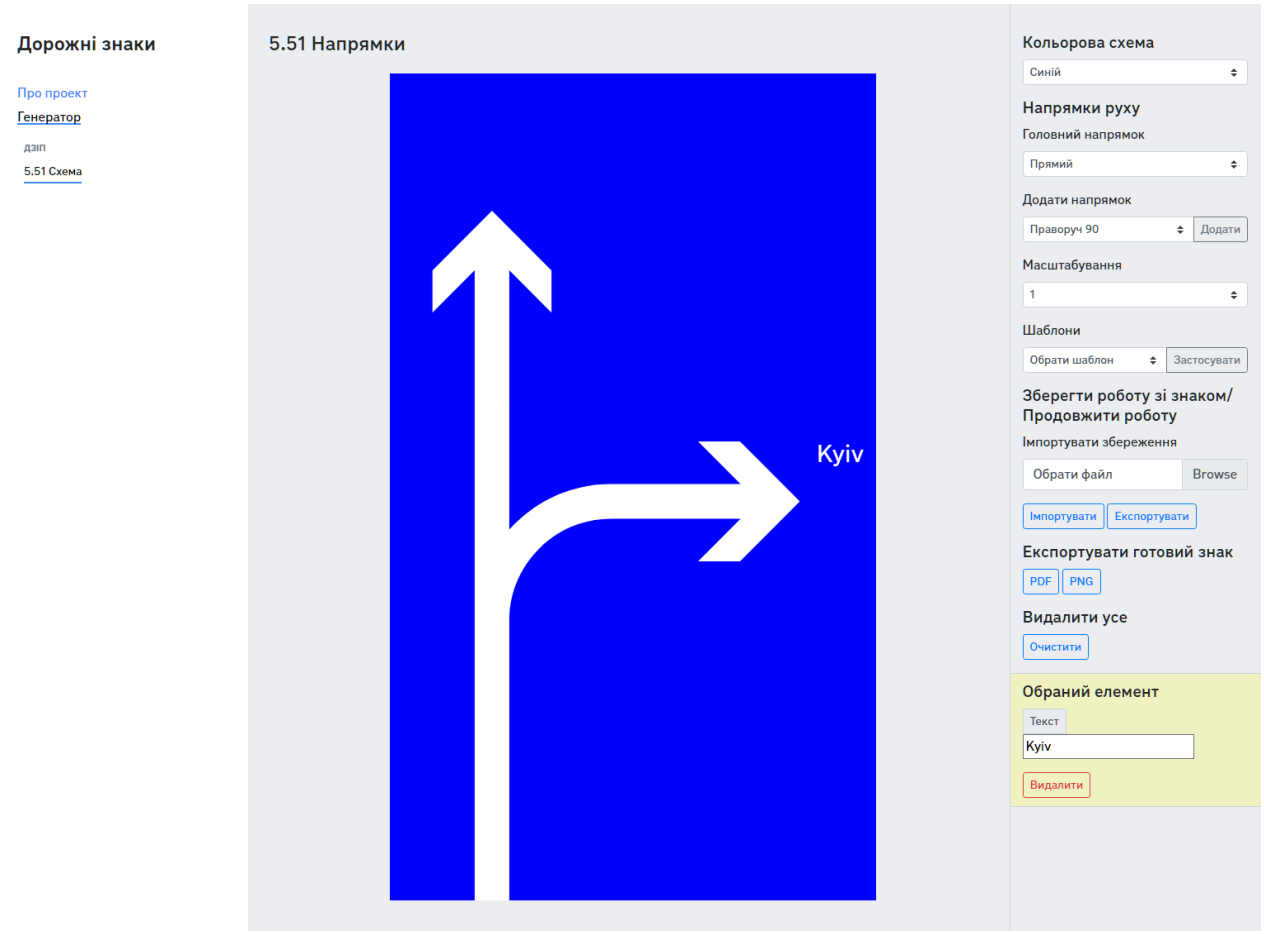


Рисунок 4.8 - Стан після виконання метода

2) Зміна розміру тексту

**Вхідні дані:**

textElem - один елемент тексту з DOM, textElemArr - опис текстового елементу з sign.json, canvasWidth - 540.

**Вихідні дані:**

canvasWidth - 680.375.

Збільшення схеми знаку у праву сторону, приклад до виконання наведено у рисунку 4.9 та приклад після виконання у рисунку 4.10.

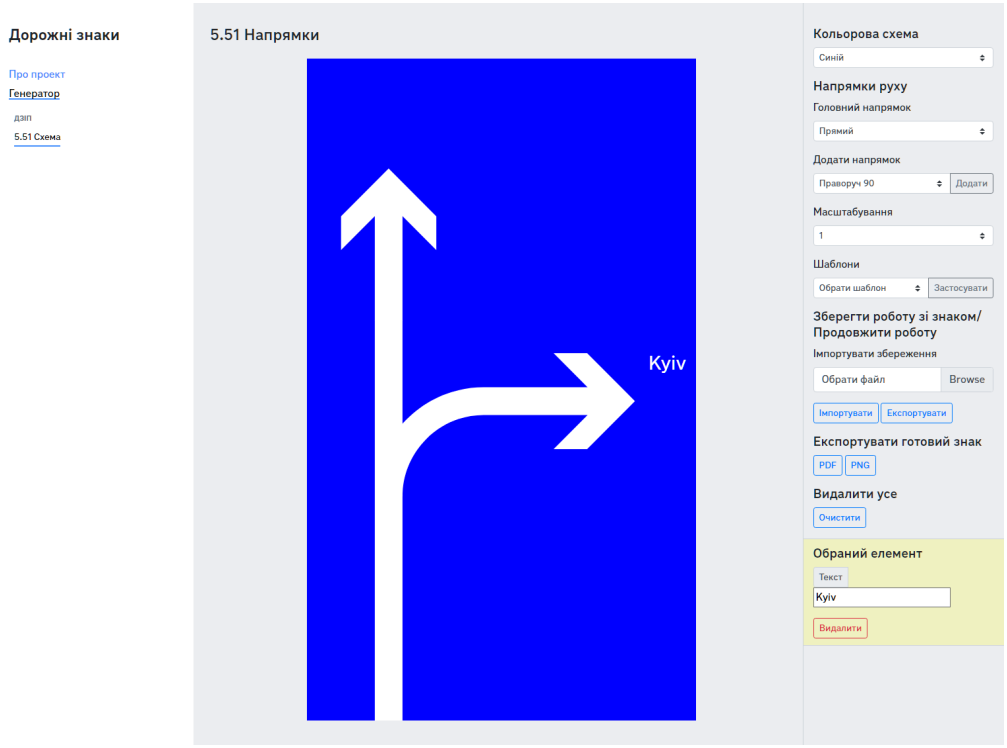


Рисунок 4.9 - Стан схеми до виконання метода

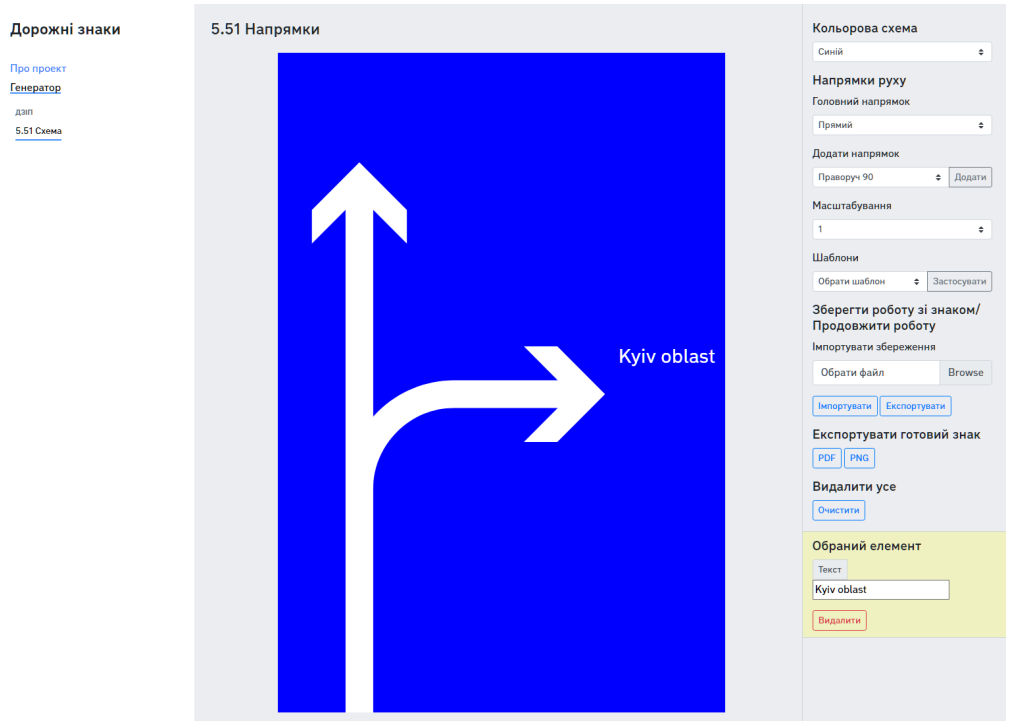


Рисунок 4.10 - Стан після виконання метода

### 4.2.2 Покриття рішень

Тестування використовуючи метод покриття рішень для методу 1 (arrowCanvasWidthCalc) надано у таблиці 4.1.

Тестування використовуючи метод покриття рішень для методу 2 (rightTextCanvasWidthCalc) надано у таблиці 4.2.

Таблиця 4.1 – Покриття рішень для методу 1

| Номер тесту | Номер рішення | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | | 2 | | 3 | | 4 | | 5 | |
| | + | - | + | - | + | - | + | - | + | - |
| 1 | * | | * | | | * | | * | | * |
| 2 | * | | | * | * | | | * | | * |
| 3 | * | | | * | | * | | * | * | |

Таблиця 4.2 – Покриття рішень для методу 2

| Номер тесту | Номер рішення | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | | 2 | | 3 | | 4 | | 5 | |
| | + | - | + | - | + | - | + | - | + | - |
| 1 | * | | * | | * | | * | | | * |
| 2 | * | | * | | * | | * | | * | |

## 4.3 Налагодження ПЗ

При тестуванні можна відокремити такі види помилок:

- Логічні (семантичні) - вони призводять до помилок під час виконання програми через неправильні обчислення. Такі помилки можна знайти виконуючи ретельне тестування зі спеціально взятими даними, до яких заздалегідь відомий правильний результат.

- Синтаксичні - це помилки, які виникають через недотримання правил мови програмування. Виявляються, зазвичай, під час компіляції. Усуваються доволі легко. Як правило, компілятор під час трансляції виявить такі помилки та попередить розробника про них.

У процесі розробки було виправлено кілька помилок таких типів.

Помилки синтаксичного типу були усунуті за допомогою компілятора, він підкреслює такі помилки ще у процесі написання коду.

Для виявлення семантичних помилок доводиться докладати більше зусиль, ніж для синтаксичних. Для цього необхідно перевіряти логіку та правильність її виконання після кожної зміни, писати для цього тести та дотримуватися норм програмування, щоб запобігти їх виникненню. [16]

**Висновки до розділу 4**

У процесі тестування, користуючись методом «білої» скриньки було знайдено та усунуто кілька зайвих умов, що виконувались завжди та деякі умови, які зовсім не виконувались. Виходячи з того, що методи не будуть використовуватися у інших програмах чи бібліотеках та дані, які задає користувач обмежені графічним інтерфейсом, тестування методом «чорної» скрині можна не проводити.

# ВИСНОВКИ

У результаті виконання цієї роботи було розроблено вебзастосунок, що дозволяє користувачу робити схеми дорожніх знаків індивідуального проектування.

У процесі розробки цього проекта був використаний принцип “реактивності” програми. Така програма дуже швидко реагує на будь-які зміни які вносить користувач та змінює вміст вебсайту чи програми стосовно цих змін. Через використання цього принципу користувач отримує дуже швидкий відгук на його дії,а також зменшує кількість зайвих елементів інтерфейсу (кнопки та ін.)

Програма була розбита на окремі модулі, це у майбутньому допоможе з підтримкою програми та пришвидшить доповнення програми новим функціоналом чи новими елементами для конструювання схем дорожніх знаків.

Сховище та конфігурація елементів схеми виконано у окремих файлах, що спрощує додавання нових елементів схеми та пришвидчує внесення змін до конфігурації елементів.

Були використані навички здобуті при проходженні курсу комп’ютерної графіки, завдяки ним розробка проекта була легшою, були розглянуті схожі проекти та принципи їх розбоки. Курс інтернет-програмування [17] допоміг з розробкою цього проекта у вебпросторі, а принцип реактивності вебзастосунку був вивчений впродовж виконання практики на виробництві.

# БІБЛІОГРАФІЧНИЙ СПИСОК


1. Oglesby S. SignMaker [Електронний ресурс] / Scott Oglesby. – Режим доступу: http://www.kurumi.com/roads/signmaker/index.html.
2. Колодько О. Маршрутне орієнтування на дорогах України: системний путівник [Електронний ресурс] / Олександр Колодько. – Режим доступу: http://roadguide.a3.kyiv.ua/#/.
3. ISO 32000-2. Portable Document Format [Електронний ресурс]. – На заміну ISO 32000 ; чинний від 1993-06-15. – Вид. офіц. – San Jose, California : ISO, 2020. – 986 с. – Режим доступу: https://www.iso.org/standard/75839.html.
4. ISO/IEC 15948. Portable Network Graphics [Електронний ресурс]. – На заміну RFC 2083 ; чинний від 2004-03-01. – Вид. офіц. – [Б. м.] : ISO/IEC, 2004. – 80 с. – Режим доступу: https://www.iso.org/standard/29581.html.
5. Duckett J. HTML and CSS: Design and Build Websites / Jon Duckett. – [Б. м] : John Wiley & Sons, 2011. – 490 с.
6. Meyer A. E. CSS: The Definitive Guide: The Definitive Guide / Eric A. Meyer. – [Б. м] : O'Reilly Media, 2006. – 538 с.
7. Haverbeke M. Eloquent JavaScript / Marijn Haverbeke. – [Б. м] : No Starch Press, 2011. – 224 с.
8. Crockford D. JavaScript Object Notation [Електронний ресурс] / Douglas Crockford // ECMA-404 The JSON Data Interchange Standard. – 2017. – Режим доступу:https://www.ecma-international.org/publications-and-standards/standards/ecma-404.
9. UML 2. (. D. G. UML 2002-- the Unified Modeling Language: Model engineering, concepts, and tools : 5th International Conference, Dresden, Germany, September 30-October 4, 2002 : proceedings / 2002 (2002 Dresden Germany) UML. – Berlin : Springer, 2002. – 447 с.
10. You E. Reactivity Fundamentals [Електронний ресурс] / Evan You. – Режим доступу: https://vuejs.org/guide/essentials/reactivity-fundamentals.html.
11. You E. Vue.js documentation [Електронний ресурс] / Evan You. – Режим доступу: https://vuejs.org.

12. Berners-Lee T. J. Scalable Vector Graphics (SVG) 2 [Електронний ресурс] / Timothy John Berners-Lee. – Режим доступу: https://www.w3.org/TR/2018/CR-SVG2-20181004/.
13. Arnold D. svg-text-to-path [Електронний ресурс] / David Arnold. – Режим доступу: https://github.com/paulzi/svg-text-to-path/blob/master/documentation.md.
14. Hall J. jsPDF [Електронний ресурс] / James Hall. – Режим доступу: https://raw.githack.com/MrRio/jsPDF/master/docs/index.html.
15. Стратегії тестування [Електронний ресурс] – Режим доступу: www.4stud.info/software-construction-and-testing/lecture10.html.
16. British Computer Society Specialist Interest Group in Software Testing, Standard for Software Component Testing [Електронний ресурс] – Режим доступу: http://www.testingstandards.co.uk/Component%20Testing.pdf.
17. Андрющенко, В. О. ІНТЕРНЕТ (програмування) дистанційний курс у системі лідер [Електронний ресурс] – Режим доступу: https://lider.ust.edu.ua/course/view.php.
18. Sinha A. Agile Methodology Vs. Traditional Waterfall SDLC: A case study on Quality Assurance process in Software Industry [Електронний ресурс] / Abhiup Sinha, Pallabi Das // 2021 5th International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech), Kolkata, India, 24–26 верес. 2021 р. – [Б. м.], 2021. – Режим доступу: https://doi.org/10.1109/iementech53263.2021.9614779.
19. Основні принципи ООП: інкапсуляція, успадкування, поліморфізм [Електронний ресурс] – Режим доступу: www.gosit.wikia.com/wiki/osnovi_opp/.
20. Zanon D. Building Serverless Web Applications: Develop scalable web apps using the Serverless Framework on AWS / Diego Zanon. – [Б. м] : Packt Publishing, 2017. – 354 с.

# ДОДАТКИ

Додаток А

# ТЕКСТ ПРОГРАМИ

**551-sign.vue**

```
<template lang="pug">
div
  .container-fluid
    .row
      .col-9.templatePreview
        h2 5.51 Напрямки

        div
          <svg id="mainSvg" :viewBox="`0 0 ${store.canvasSizeWidth} ${store.canvasSizeHeight}`" :width="store.canvasSizeWidth * store.scaleVal" :height="store.canvasSizeHeight * store.scaleVal" style="display:block; margin:auto" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink">
            <svgCode v-if="renderComponent" :chosenSign="1" v-show="false"></svgCode>
            <svgCode id="templateSvg" v-if="renderComponent" :chosenSign="0"></svgCode>
          </svg>

          <svg hidden id="instanceSVG" :viewBox="`0 0 ${store.canvasSizeWidth} 1000`" :width="store.canvasSizeWidth * store.scaleVal" :height="1000 * store.scaleVal" style="display:block; margin:auto" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink">
          </svg>

      .col-3.detailsSidebar
        h5 Кольорова схема
        .form-group
          select.custom-select.custom-select-sm(
            v-model="store.colorThemeVal"
            )
            option(value="1" name="colorTheme") Синій
            option(value="2" name="colorTheme") Зелений
            option(value="3" name="colorTheme") Білий

        h5 Напрямки руху
        .form-group
          label Головний напрямок
          select.custom-select.custom-select-sm(id="shapeMainDropVal" @change="shapeMainSelect()")
            option(value="none") Відсутній
            option(v-for="shape in shapesMain" :value="shape.name") {{shape.nameUI}}

        .form-group
          label Додати напрямок
          .input-group
            select.custom-select.custom-select-sm(id="shapeSubDropVal")
              option(value="none") Обрати напрямок
```

```
            option(v-for="shape in shapesSub"
:value="shape.name") {{shape.nameUI}}
          .input-group-append
            button.btn.btn-sm.btn-outline-
secondary(@click="addSubShape()") Додати

      .form-group
        label Масштабування
        .input-group
          select.custom-select.custom-select-
sm(id="viewBoxScale" @change="event =>
store.scaleVal = event.target.value")
            option(v-for="scale in scaleVals")
{{scale}}

      .form-group
        label Шаблони
        .input-group
          select.custom-select.custom-select-
sm(id="signPresets")
            option(value="none") Обрати шаблон
            option(v-for="preset in signPresets"
:value="preset.presetName")
{{preset.presetNameUI}}
          .input-group-append
            button.btn.btn-sm.btn-outline-
secondary(@click="importFormPreset()")
Застосувати

      h5 Зберегти роботу зі
знаком/Продовжити роботу
      .form-group
        label(for="fileUpload") Імпортувати
збереження
        .input-group.input-group-sm.mb-3
          .custom-file
            input.custom-file-input(id="fileUpload"
type="file"
@change="updateImportFileName()")
            label.custom-file-label
{{importFileName}}

      .form-group
        .form-inline
          button.btn.btn-sm.btn-outline-
primary(@click="importFromJsonFile()")
Імпортувати
          |  
          button.btn.btn-sm.btn-outline-
primary(@click="exportToJsonFile()")
Експортувати

      h5 Експортувати готовий знак
      .form-group
        .form-inline
          button.btn.btn-sm.btn-outline-
primary(@click="pathConvert()") PDF
          |  
          button.btn.btn-sm.btn-outline-
primary(@click="pngConvert()") PNG

      h5 Видалити усе
      .form-group
        .form-inline
          button.btn.btn-sm.btn-outline-
primary(@click="deleteAll()") Очистити

      .selectedObject(v-if="store.shapeFocusId !=
null")
        h5 Обраний елемент
```

```
//- Text content
.form-group.form-row(v-if="createdSign[store.shapeFocusId].textContent != null")
  .col-5
    .input-group.input-group-sm
      .input-group-prepend
       .input-group-text Текст
      input(id="textInput" :value="createdSign[store.shapeFocusId].textContent" @input="textOnInputChange()")

div(v-if="createdSign[store.shapeFocusId].parentId < 1 || createdSign[store.shapeFocusId].parentId == undefined")
  .form-group
  label Додати субелемент
  .input-group
    select.custom-select.custom-select-sm(id="shapeAddDropVal" @change="textPosDropMenu()")
      option(value="none") Обрати елемент
      option(v-for="shape in shapesAdd" :value="shape.name") {{shape.nameUI}}
    .input-group-append(v-if="!textPosVisible")
      button.btn.btn-sm.btn-outline-secondary(v-if="createdSign[store.shapeFocusId].parentId != undefined", @click="addAdditionShape(store.shapeFocusId)") Додати
      button.btn.btn-sm.btn-outline-secondary(v-if="createdSign[store.shapeFocusId].parentId == undefined", @click="addAdditionShape(1)") Додати
  label(v-if="textPosVisible") Обрати позицію тексту
  .input-group
    select.custom-select.custom-select-sm(id="textPosPick" v-if="textPosVisible" @change="textAddButton()")
      option(value="none") Обрати позицію
      option(v-for="pos in textPositions[store.textPositionsIndex].positions" :value="[pos.x, pos.y]") {{pos.posName}}
    .input-group-append(v-if="textAddButtonVisible")
      button.btn.btn-sm.btn-outline-secondary(v-if="createdSign[store.shapeFocusId].parentId != undefined", @click="addAdditionShape(store.shapeFocusId)") Додати
      button.btn.btn-sm.btn-outline-secondary(v-if="createdSign[store.shapeFocusId].parentId == undefined", @click="addAdditionShape(1)") Додати

.input-group
  button.btn.btn-sm.btn-outline-danger(v-if="store.shapeFocusId > 1", @click="deleteShape()") Видалити
  button.btn.btn-sm.btn-outline-danger(v-else, @click="deleteAll()") Видалити
```

```
</template>

<script>
import { store } from '@/store.js';

//import { svgConvert } from '@/svgConvert.js';
import { saveSvgAsPng } from 'save-svg-as-png/lib/saveSvgAsPng.js'

// SVG text → path converter V2.0
import Session from 'svg-text-to-path/entries/browser.js';
import FontkitFont from 'svg-text-to-path/renderer/FontkitFont.js';
import ConfigProvider from 'svg-text-to-path/providers/config/ConfigProvider.js';

Session.defaultRenderer = FontkitFont;
Session.defaultProviders = [ConfigProvider];

// SVG 2 PDF converter
import { jsPDF } from 'jspdf'
import 'svg2pdf.js'

import svgCode from '/src/components/svgCode.vue'
import {createdSign, noShiftSign, colorTheme, viewBoxScale} from "/src/sign.json"
import * as signJson from "/src/sign.json"

import { shapesMain, shapesSub, shapesAdd, textPositions, signPresets } from "/src/shapesTemplates.json"

export default {
name: 'test',
components:
{
 svgCode
},

data(){
 return{
   store,
   saveSvgAsPng,
   createdSign,
   noShiftSign,
   colorTheme,
   viewBoxScale,
   signJson, //used in JSON export only
   shapesMain,
   shapesSub,
   shapesAdd,
   textPositions,
   signPresets,
   renderComponent: true,
   importFileName: "Обрати файл",

   scaleVals: [1, 0.75, 0.5, 0.25, 0.1],
   textPosVisible: false,
   textAddButtonVisible: false,
 }
},

watch:
{
 "store.colorThemeVal"()
 {
   this.colorTheme = this.store.colorThemeVal;
```

```
    this.signJson.default.colorTheme = this.store.colorThemeVal;
   },

   "store.scaleVal"()
   {
    this.viewBoxScale = this.store.scaleVal;
    this.signJson.default.viewBoxScale = this.store.scaleVal;
   },

   "store.shapeFocusId"()
   {
    if(this.store.shapeFocusId == null || this.createdSign[this.store.shapeFocusId].name == "Text")
    {
     this.textPosVisible = false;
     this.textAddButtonVisible = false;
    }
   },

   "createdSign.length"()
   {
    this.canvasSize();
   },
  },

  computed:
  {

  },

  methods:
  {
   pngConvert()
   {
    let fileName = 'Created-sign.png';

saveSvgAsPng(document.querySelector("#templateSvg"), fileName, {fonts:[{
      url: "http://" + window.location.host + "/fonts/RoadUA-Regular.woff2",
      text:
       `@font-face {
       font-family: 'RoadUA';
       src: url(http://`+ window.location.host +`/fonts/RoadUA-Regular.woff2) format('woff2');
       }`
      }]
     });
   },

   pathConvert()
   {

document.querySelector('#instanceSVG').innerHTML = document.querySelector('#templateSvg').innerHTML;

    let session = new Session(document.querySelector('#instanceSVG'), {
      fonts: {
       "RoadUA": [
        {
         "wght": 400,
```

```
          "ital": 0,
          "source":       "/fonts/RoadUA/RoadUA-
Regular.otf"
         }
        ],
         "RoadUA-Medium": [
         {
          "wght": 400,
          "ital": 0,
          "source":       "/fonts/RoadUA/RoadUA-
Medium.otf"
         }
        ],
         "RoadUA-Bold": [
         {
          "wght": 400,
          "ital": 0,
          "source":       "/fonts/RoadUA/RoadUA-
Bold.otf"
         }
        ],
       },
    });

    session.replaceAll().then(() => {
     this.pdfConvert();
    })
   },

   pdfConvert()
   {
    // PDF document setup
    let x = 10;
    let y = 10;
    let width = this.store.canvasSizeWidth *
this.store.scaleVal;
    let height = this.store.canvasSizeHeight *
this.store.scaleVal;
    let canvasWidth = width + 20;
    let canvasHeight = height + 20;
    let orientation = canvasWidth > canvasHeight
? 'l' : 'p';
    let fileName = 'Created-sign.pdf';

    const doc = new jsPDF({
     orientation: orientation,
     unit: 'mm',
     format: [canvasWidth, canvasHeight],
     putOnlyUsedFonts:true,
     compress: true})

    const                element                =
document.getElementById('instanceSVG')
    doc.svg(element, {
        x,
        y,
        width,
        height
      })
      .then(() => {
       // Save pdf
       doc.save(fileName);
      })
   },

   textOnInputChange()
   {
```

```
    let textSizeBeforeChange = document.getElementById(this.store.shapeFocusId).getComputedTextLength();


this.createdSign[this.store.shapeFocusId].textContent = event.target.value;

document.getElementById(this.store.shapeFocusId).textContent = event.target.value;

this.noShiftSign[this.store.shapeFocusId].textContent = event.target.value;

document.getElementById(this.store.shapeFocusId + "c").textContent = event.target.value;

    let textSizeAfterChange = document.getElementById(this.store.shapeFocusId).getComputedTextLength();


if(this.noShiftSign[this.store.shapeFocusId].side == "left")
    {
      this.noShiftSign[this.store.shapeFocusId].x += textSizeBeforeChange - textSizeAfterChange;
    }

    this.canvasSize();
  },

  textPosDropMenu()
  {
if(document.getElementById("shapeAddDropVal").value == "Text")
    {
     this.textPosVisible = true;
    }
    else
    {
     this.textPosVisible = false;
     this.textAddButtonVisible = false;
    }
  },

  textAddButton()
  {

if(document.getElementById("textPosPick").value == "none")
    {
     this.textAddButtonVisible = false;
    }
    else
    {
     this.textAddButtonVisible = true;
    }
  },

  async canvasSize()
  {
    let widthClac = 0;
    let arrowPosition = 0;

    let rightWidest = 0;
    let leftWidest = 0;
```

```
    let textElementArrayRight = [];
    let textElementArrayLeft = [];

    for(let i = 0; i < this.noShiftSign.length; i++)
    {
     if(this.noShiftSign[i].facing != undefined)
     {
      if(this.noShiftSign[i].facing == "right" &&
rightWidest < this.noShiftSign[i].width)
      {
       rightWidest = this.noShiftSign[i].width;
      }
      else if(this.noShiftSign[i].facing == "left"
&& leftWidest < this.noShiftSign[i].width)
      {
       leftWidest = this.noShiftSign[i].width;
      }
     }

     if(this.noShiftSign[i].name == "Text")
     {
      if(this.noShiftSign[i].side == "right")
      {

textElementArrayRight.push(noShiftSign[i]);
      }
      else
      {

textElementArrayLeft.push(noShiftSign[i]);
      }
     }
    }
```

```
    let buffObj =
this.arrowCanvasWidthCalc(rightWidest,
leftWidest, widthClac, this.noShiftSign);

    widthClac = buffObj.canvasWidth;
    arrowPosition = buffObj.arrowPos;

    let renderPromise = new
Promise(function(resolve)
    {
     setTimeout(function() {resolve();}, 50);
    });

    let textElementsRight = await renderPromise;
    let textElementsLeft = await renderPromise;
    textElementsRight = [];
    textElementsLeft = [];


    let widthChangeCheck = widthClac;
    widthClac =
this.rightTextCanvasWidthCalc(textElementsRig
ht, textElementArrayRight, widthClac);

    for(let i = 0; i < textElementArrayLeft.length;
i++)
    {

textElementsLeft.push(document.getElementByI
d(textElementArrayLeft[i].id));
    }


    let mostLeftText = 0;
    for(let i = 0; i < textElementsLeft.length; i++)
```

```
    {

if(textElementsLeft[mostLeftText].x.baseVal[0].
value >= textElementsLeft[i].x.baseVal[0].value)
      {
       mostLeftText = i;
      }
    }

    let elemShift = 0;
    if(textElementsLeft.length > 0)
    {

if(textElementsLeft[mostLeftText].x.baseVal[0].
value < 0)
      {
       if(widthChangeCheck - widthClac == 0)
       {
         widthClac                              +=
(textElementsLeft[mostLeftText].x.baseVal[0].v
alue * -1);
       }
       elemShift                                =
(textElementsLeft[mostLeftText].x.baseVal[0].v
alue * -1) + 15;
      }
    }

    if(this.store.shapeFocusId != null)
    {

if(this.createdSign[this.store.shapeFocusId].nam
e == "Text")
      {
        this.createdSign[this.store.shapeFocusId].x
= this.noShiftSign[this.store.shapeFocusId].x;
       }
      }

     this.createdSign[0].x = arrowPosition +
elemShift;
     this.noShiftSign[0].x = arrowPosition;

     this.store.canvasSizeWidth = widthClac;
    },

   arrowCanvasWidthCalc(rightW,          leftW,
canvasWidth, sign)
    {
     let arrowPos = 0;
     if(sign.length > 1)
     {
     //No side arrows
      if(rightW == 0 && leftW == 0)
      {
       canvasWidth      =      sign[1].width      +
sign[0].width;
       arrowPos = (canvasWidth - sign[0].width) /
2;
      }
      //Only right arrows
      else if(rightW > 0 && leftW == 0)
      {
       if(sign[1].multiWay)
       {
        canvasWidth      =      sign[0].width      +
(sign[1].width / 2) + rightW;
        arrowPos = (sign[1].width / 2);
       }
```

```
        else
        {
          canvasWidth = sign[1].width + rightW;
          arrowPos = sign[1].width - sign[0].width;
        }
      }
      //Only left arrows
      else if(rightW == 0 && leftW > 0)
      {
        if(sign[1].multiWay)
        {
          canvasWidth = (sign[1].width / 2) + leftW
+ sign[0].width;
          arrowPos = canvasWidth - (sign[1].width
/ 2) - sign[0].width;
        }
        else
        {
          canvasWidth = sign[1].width + leftW;
          arrowPos = canvasWidth - sign[1].width;
        }
      }
      //Left and Right arrows
      else
      {
        if(sign[1].multiWay)
        {
          if(rightW > leftW)
          {
            canvasWidth = leftW + rightW +
sign[0].width * 2;
            arrowPos = (canvasWidth -
sign[0].width - (rightW - leftW)) / 2;
          }
          else if(rightW < leftW)
          {
            canvasWidth = leftW + rightW +
sign[0].width * 2;
            arrowPos = (canvasWidth -
sign[0].width + (leftW - rightW)) / 2;
          }
          else
          {
            canvasWidth = leftW + rightW +
sign[0].width * 2;
            arrowPos = (canvasWidth -
sign[0].width) / 2;
          }
        }
        else
        {
          if(rightW > leftW)
          {
            canvasWidth = sign[1].width + leftW +
rightW - sign[0].width;
            arrowPos = (canvasWidth -sign[0].width
- (rightW - leftW)) / 2;
          }
          else if(rightW < leftW)
          {
            canvasWidth = sign[1].width + leftW +
rightW - sign[0].width;
            arrowPos = (canvasWidth -
sign[0].width + (leftW - rightW)) / 2;
          }
          else
          {
            canvasWidth = sign[1].width + leftW +
rightW - sign[0].width;
```

```
            arrowPos = (canvasWidth - sign[0].width) / 2;
          }
        }
      }
    }

    return {canvasWidth, arrowPos};
  },

  rightTextCanvasWidthCalc(textElem, textElemArr, canvasWidth)
  {
    for(let i = 0; i < textElemArr.length; i++)
    {
      textElem.push(document.getElementById(textElemArr[i].id));
    }

    let mostRightText = 0;
    for(let i = 0; i < textElemArr.length; i++)
    {
      if(textElemArr[mostRightText].x + textElem[mostRightText].getComputedTextLength() < textElemArr[i].x + textElem[i].getComputedTextLength())
      {
        mostRightText = i;
      }
    }

    if(textElemArr.length > 0)
    {
      if(textElemArr[mostRightText].x + createdSign[0].x + textElem[mostRightText].getComputedTextLength() > canvasWidth)
      {
        let widthDiff = textElemArr[mostRightText].x + createdSign[0].x + textElem[mostRightText].getComputedTextLength() - canvasWidth;
        canvasWidth = canvasWidth + widthDiff + 15;
      }
    }

    return canvasWidth;
  },

  updateImportFileName()
  {
    let fullPath = document.getElementById("fileUpload").value;
    if (fullPath)
    {
      let startIndex = (fullPath.indexOf('\\') >= 0 ? fullPath.lastIndexOf('\\') : fullPath.lastIndexOf('/'));
      var filename = fullPath.substring(startIndex);
      if (filename.indexOf('\\') === 0 || filename.indexOf('/') === 0)
      {
        filename = filename.substring(1);
      }
    }
```

```
    this.importFileName = filename;
   },

   shapeMainSelect()
   {
    let dropVal = document.getElementById("shapeMainDropVal").value;

    if(dropVal == "none")
    {
     this.deleteAll();
    }
    else
    {
     for(let i = 0; i < this.shapesMain.length; i++)
     {
      if(this.shapesMain[i].name == dropVal)
      {
       let delIndex = null;
       for(let i = 0; i < this.createdSign.length; i++)
       {
        if(this.createdSign[i].id == 1)
        {
         delIndex = i;
        }
       }

       this.store.shapeFocusId = null;

       if(delIndex != null)
       {
        for(let i = 0; i < this.createdSign.length; i++)
        {
         if(this.createdSign[i].parentId == 1)
         {
          this.createdSign.splice(i, 1);
          this.noShiftSign.splice(i, 1);
          i--;
         }
        }

        this.createdSign.splice(delIndex, 1, JSON.parse(JSON.stringify(this.shapesMain[i])));
        this.noShiftSign.splice(delIndex, 1, JSON.parse(JSON.stringify(this.shapesMain[i])));
        this.noShiftSign[createdSign.length - 1].id = "1c";
       }
       else
       {

this.createdSign.push(JSON.parse(JSON.stringify(this.shapesMain[i])));

this.noShiftSign.push(JSON.parse(JSON.stringify(this.shapesMain[i])));
        this.noShiftSign[createdSign.length - 1].id = "1c";
       }
      }
     }
    }
```

```
    if(dropVal != "none" && this.createdSign.length > 1)
    {
     this.canvasSize();
    }
    else
    {
     this.store.canvasSizeWidth = 0;
    }

    this.forceRerender();
   },

   importFormPreset()
   {
    let dropVal = document.getElementById("signPresets").value;
    this.store.shapeFocusId = null;

    for(let i = this.createdSign.length; i > 0; i--)
    {
     this.createdSign.pop();
     this.noShiftSign.pop();
    }

    for(let i = 0; i < this.signPresets[dropVal].sign.length; i++)
    {
     let buffSign = JSON.parse(JSON.stringify(this.signPresets[dropVal].sign[i]));
     let buffNoShiftSign = JSON.parse(JSON.stringify(this.signPresets[dropVal].noShiftSign[i]));
     this.noShiftSign.push(buffNoShiftSign);
     this.createdSign.push(buffSign);
    }

    this.store.colorThemeVal = this.signPresets[dropVal].colorTheme;
    this.store.scaleVal = this.signPresets[dropVal].viewBoxScale;

    document.getElementById("viewBoxScale").value = this.signPresets[dropVal].viewBoxScale;

    document.getElementById("shapeMainDropVal").value = this.signPresets[dropVal].sign[1].name;
    this.forceRerender();
    this.canvasSize();
   },

   importFromJsonFile()
   {
    const files = document.getElementById('fileUpload').files;
    if (files.length <= 0)
    {
     return false;
    }

    const fr = new FileReader();
    fr.onload = e =>
    {
     const result = JSON.parse(e.target.result);

     this.store.shapeFocusId = null;

     for(let i = this.createdSign.length; i > 0; i--)
```

```
      {
       this.createdSign.pop();
       this.noShiftSign.pop();
      }

      for(let i = 0; i < result.default.createdSign.length; i++)
      {

this.createdSign.push(result.default.createdSign[i]);

this.noShiftSign.push(result.default.noShiftSign[i]);
      }

      this.store.colorThemeVal = result.default.colorTheme;
      this.store.scaleVal = result.default.viewBoxScale;

document.getElementById("viewBoxScale").value = result.default.viewBoxScale;

document.getElementById("shapeMainDropVal").value = result.default.createdSign[1].name;
      this.forceRerender();
     }

     fr.readAsText(files.item(0));
   },

   exportToJsonFile()
   {
    let dataStr = JSON.stringify(this.signJson);

    let dataUri = 'data:application/json;charset=utf-8,'+ encodeURIComponent(dataStr);

    let exportFileDefaultName = 'data.json';

    let linkElement = document.createElement('a');
    linkElement.setAttribute('href', dataUri);
    linkElement.setAttribute('download', exportFileDefaultName);
    linkElement.click();
   },

   printJsonData()
   {
    console.log(this.signJson);
   },

   deleteAll()
   {
    if(this.createdSign.length > 0)
    {
     this.createdSign.splice(1, this.createdSign.length - 1);
     this.noShiftSign.splice(1, this.noShiftSign.length - 1);
    }

    this.store.canvasSizeWidth = 0;
    this.store.shapeFocusId = null;
    this.canvasSize();
    this.forceRerender();
```

```
document.getElementById("shapeMainDropVal
").selectedIndex = 0;
  },

  deleteShape()
  {
   for(let i = this.createdSign.length - 1; i >= 0;
i--)
   {
    let boolFocusShapeCheck =
this.createdSign[i].id == this.store.shapeFocusId;
    if(this.createdSign[i].parentId ==
this.createdSign[this.store.shapeFocusId].id ||
boolFocusShapeCheck)
    {
     this.createdSign.splice(i, 1);
     this.noShiftSign.splice(i, 1);
     for(let j = i; j < this.createdSign.length; j++)
     {
      this.createdSign[j].id--;
      this.noShiftSign[j].id =
this.changeIdInNoShift(this.noShiftSign[j].id, -
1);

      if(this.createdSign[j].parentId > i)
      {
       this.createdSign[j].parentId--;
       this.noShiftSign[j].parentId =
this.changeIdInNoShift(this.noShiftSign[j].paren
tId, -1);
      }
     }

     if(boolFocusShapeCheck)
     {
      i = 0;
     }
    }
   }

   this.store.shapeFocusId = null;
   this.canvasSize();
   this.forceRerender();
  },

  changeIdInNoShift(noShiftElem, idChange)
  {
   noShiftElem = noShiftElem.slice(0, -1);
   noShiftElem = parseInt(noShiftElem);
   noShiftElem += idChange;
   noShiftElem = noShiftElem.toString();
   noShiftElem = noShiftElem + "c";
   return noShiftElem;
  },

  addAdditionShape(parId)
  {
   let dropVal =
document.getElementById("shapeAddDropVal")
.value;

   for(let i = 0; i < this.shapesAdd.length; i++)
   {
    if(this.shapesAdd[i].name == dropVal)
    {

this.createdSign.push(JSON.parse(JSON.stringif
y(this.shapesAdd[i])));
```

```
this.noShiftSign.push(JSON.parse(JSON.stringify(this.shapesAdd[i])));

        this.createdSign[this.createdSign.length - 1].id = this.createdSign.length - 1;
        this.noShiftSign[this.createdSign.length - 1].id = (this.createdSign.length - 1).toString() + "c";

        this.createdSign[this.createdSign.length - 1].parentId = parId;
        this.noShiftSign[this.createdSign.length - 1].parentId = parId.toString() + "c";
      }
    }

    if(dropVal == "Text")
    {
     let textPosDropVal = document.getElementById("textPosPick").value.split(",");
      this.createdSign[this.createdSign.length - 1].x = parseInt(textPosDropVal[0]);
      this.createdSign[this.createdSign.length - 1].y = parseInt(textPosDropVal[1]);
      this.createdSign[this.createdSign.length - 1].side = "right";

      this.noShiftSign[this.noShiftSign.length - 1].x = parseInt(textPosDropVal[0]);
      this.noShiftSign[this.noShiftSign.length - 1].y = parseInt(textPosDropVal[1]);
      this.noShiftSign[this.noShiftSign.length - 1].side = "right";

      if(this.createdSign[parId].facing == "left")
      {
       this.createdSign[this.createdSign.length - 1].x = (this.createdSign[this.createdSign.length - 1].x + 100) * -1;
       this.createdSign[this.createdSign.length - 1].side = "left";

       this.noShiftSign[this.noShiftSign.length - 1].x = (this.noShiftSign[this.noShiftSign.length - 1].x + 100) * -1;
       this.noShiftSign[this.noShiftSign.length - 1].side = "left";
      }

      if(this.createdSign[parId].facing == undefined)
      {
       if(this.createdSign[this.createdSign.length - 1].x < 0)
       {
        this.createdSign[this.createdSign.length - 1].side = "left";
        this.noShiftSign[this.noShiftSign.length - 1].side = "left";
       }
      }
     }

     this.canvasSize();
     this.forceRerender();
    },

    addSubShape()
```

```
    {
     let dropVal =
document.getElementById("shapeSubDropVal")
.value;

     for(let i = 0; i < this.shapesSub.length; i++)
     {
      if(this.shapesSub[i].name == dropVal)
      {

this.createdSign.push(JSON.parse(JSON.stringif
y(this.shapesSub[i])));

this.noShiftSign.push(JSON.parse(JSON.stringif
y(this.shapesSub[i])));

        this.createdSign[this.createdSign.length -
1].id = this.createdSign.length - 1;
        this.noShiftSign[this.createdSign.length -
1].id = (this.createdSign.length - 1).toString() +
"c";
       }
     }

     this.canvasSize();
     this.forceRerender();
    },

   forceRerender()
   {
    this.renderComponent = false;
    this.$nextTick(() => {
      this.renderComponent = true;
    });
   },
  },
 }

</script>

<style lang="scss" scoped>
@import '@/assets/styles/_svg-templates.scss';
.templatePreview {
 padding-top: 60px;
}

.focusStyle {
 border: 1px solid red;
}

.preview {
 min-height: 320px;
 padding-top: 20px;
 padding-bottom: 0;
 @include respond-below(md) {
  min-height: 240px;
 }
}

.sign-size {
 div {
  opacity: .5;
 }
}

.detailsSidebar {
 padding-top: 60px;
 background-color: transparent;
 min-height: 100vh;
 border-left: 1px solid $LightGrey;
```

```
}

.selectedObject {
  margin: 0 -25px 0 -15px;
  padding: 10px 25px 10px 15px;
  background-color: rgba(255,255,0,.2);
  border-width: 1px 0 1px 0;
  border-style: solid;
  border-color: $LightGrey;
}


</style>
```

**svgCode.vue**

```
<template lang="pug">
<g ref="box" class="box">
  <rect :width="store.canvasSizeWidth" :height="store.canvasSizeHeight" :fill="colorThemes[store.colorThemeVal - 1].backgroundColor" @mousedown="info(null)" @mouseup="drop()"/>
  <rect :id="legId()" v-if="signs[chosenSign].length > 1" :width="signs[chosenSign][0].width" :height="1000 - signs[chosenSign][0].y" :x="signs[chosenSign][0].x" :y="signs[chosenSign][0].y" :style="cursor" :fill="colorThemes[store.colorThemeVal - 1].elementsColor" @mousedown="info(0)"/>
  <component :id="shape.id" v-for="shape in signs[chosenSign]" :passedShape="shape" :passedParentShape="signs[chosenSign][wichSignId(shape)]" :passedMainShape="shapeCopyPrevention(signs[chosenSign][wichSignId(shape)])" :passedColor="colorThemes[store.colorThemeVal - 1].elementsColor" :style="cursor" @mousedown="drag($event, shape.id); info(shape.id)" @mouseup="drop()" :is="shape.name"/>
</g>
</template>

<script>
import {createdSign, noShiftSign} from "/src/sign.json"

import { colorThemes, textPositions } from "/src/shapesTemplates.json"
import { store } from '@/store.js';

//main arrows
import Straight0d_main from "./Straight0d_main.vue";
import Left90d_Right90d_main from "./Left90d_Right90d_main.vue";
import Left90d_Straight0d_Right90d_main from "./Left90d_Straight0d_Right90d_main";

//sub shapes
import Right90d from "./Right90d.vue";
import Right90d_Left180d from "./Right90d_Left180d.vue";
import Right270d from "./Right270d.vue";
import Right45d from "./Right45d.vue";
import Right180d from "./Right180d.vue";

import Left90d from "./Left90d.vue";
import Left45d from "./Left45d.vue";
import Left180d from "./Left180d.vue";
```

```
//add shapes
import Text from "./Text.vue";
import TruckSign from "./TruckSign.vue";

export default {
 name: 'svgCode',
 components: {
  Straight0d_main,
  Left90d_Right90d_main,
  Left90d_Straight0d_Right90d_main,

  Right90d,
  Right90d_Left180d,
  Right270d,
  Right45d,
  Right180d,

  Left90d,
  Left45d,
  Left180d,

  Text,
  TruckSign
 },

 props: ['chosenSign'],

 data(){
  return{
   store,
   createdSign,
   noShiftSign,
   colorThemes,
   textPositions,
   tempDragId: null,
   dragOffsetX: null,
   dragOffsetY: null,

   signs: [createdSign, noShiftSign]

  }
 },

 watch:
 {
 },

 computed:
 {
  cursor()
  {
   return `cursor: ${this.dragOffsetX ? 'grabbing' : 'grab'}`
  },
 },

 methods:
 {
  legId()
  {
   let retId = null;
   if(this.chosenSign == 1)
   {
    retId = "0c";
   }
   else
   {
    retId = 0;
   }
```

```
 return retId;
},


wichSignId(forShape)
{
 let retId = null;
 if(this.chosenSign == 1)
 {
  if(forShape.parentId != undefined)
  {
   if(!forShape.parentId.isInteger)
   {
    let buffStr = forShape.parentId;
    retId = parseInt(buffStr);
   }
   else
   {
    retId = forShape.parentId;
   }
  }
 }
 else
 {
  retId = forShape.parentId;
 }

 return retId;
},


info(id)
{
 if(this.store.shapeFocusId != null)
 {
  if(this.store.shapeFocusId > 1)
  {

document.getElementById(this.store.shapeFocus
Id).classList.remove("focusStyle");
  }
  else
  {

document.getElementById(0).classList.remove("
focusStyle");

document.getElementById(1).classList.remove("
focusStyle");
  }
 }

 this.store.shapeFocusId = id;

 if(this.store.shapeFocusId != null)
 {
  if(this.store.shapeFocusId > 1)
  {

document.getElementById(this.store.shapeFocus
Id).classList.add("focusStyle");
  }
  else
  {

document.getElementById(0).classList.add("foc
usStyle");
```

```
document.getElementById(1).classList.add("focusStyle");
      }
    }

    if(this.store.shapeFocusId != null)
    {
     for(let i = 0; i < this.textPositions.length; i++)
      {

if(this.signs[this.chosenSign][this.store.shapeFocusId].name.replace(/Right|Left|Straight/g, "") == this.textPositions[i].name)
       {
        this.store.textPositionsIndex = i;
       }
      }
    }
  },

  drag({offsetX, offsetY}, shapeId)
  {
   if(shapeId != 1 && this.signs[this.chosenSign][shapeId].name != "Text")
    {
     if(this.chosenSign == 1)
     {
      this.tempDragId = parseInt(shapeId.slice(0, -1));
     }
     else
     {
      this.tempDragId = shapeId;
     }

     if(this.tempDragId == 1)
     {
      this.tempDragId--;
     }


if(this.signs[this.chosenSign][this.tempDragId].facing == "left")
     {
      this.dragOffsetX = offsetX + this.signs[this.chosenSign][this.tempDragId].x * store.scaleVal;
     }
     else
     {
      this.dragOffsetX = offsetX - this.signs[this.chosenSign][this.tempDragId].x * store.scaleVal;
     }
     this.dragOffsetY = offsetY - this.signs[this.chosenSign][this.tempDragId].y * store.scaleVal;


this.$refs.box.addEventListener('mousemove', this.move);
    }
  },

  drop()
  {
   this.dragOffsetX = this.dragOffsetY = null;
```

```
this.$refs.box.removeEventListener('mousemove
', this.move)
   },

   move(event)
   {

if(this.signs[this.chosenSign][this.tempDragId].f
acing == "left")
    {

this.signs[this.chosenSign][this.tempDragId].x =
(-event.offsetX + this.dragOffsetX) /
store.scaleVal;
    }
    else
    {

this.signs[this.chosenSign][this.tempDragId].x =
(event.offsetX - this.dragOffsetX) /
store.scaleVal;
    }



this.signs[this.chosenSign][this.tempDragId].y =
(event.offsetY - this.dragOffsetY) /
store.scaleVal;
   },

   shapeCopyPrevention(shape)
   {
    if(this.signs[this.chosenSign][0] == shape)
    {
     let returnVal = {"x": 0, "y": 0};
     return returnVal;
    }
    else
    {
     return this.signs[this.chosenSign][0];
    }
   }
  },
}

</script>

<style lang="scss" scoped>
rect.focusStyle {
  stroke: red;
  stroke-width: 2px;
  stroke-dasharray: 5 5;
  opacity: .5;
}
</style>
```

**Left45d.vue**

```
<template lang="pug">
//<rect :x="passedParentShape.x +
passedParentShape.width - 1" :y="passedShape.y
+ passedParentShape.y"
:height="passedShape.height"
:width="passedShape.width + passedShape.x"
:fill="passedColor"/>

<svg version="1.1" id="Layer_1"
xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink"
:x="passedParentShape.x - passedShape.width +
passedParentShape.width" :y="passedShape.y +
passedParentShape.y" width="304px"
```

```
height="322px" viewBox="0 0 304 322"
style="enable-background:new 0 0 304 322;"
xml:space="preserve">
<path :fill="passedColor"
d="M2.2717285,0h102.5304565l35.999939,36H
67.9703369c0,0,184.3724365,184.2884521,187.
6917725,187.5257568C282.6689453,249.86853
03,304,270.6751709,304,320.3543701h-42c0-
31.9750977-11.1767578-42.8769531-
35.6640625-
66.7626953C222.9248047,250.2640381,38.2716
064,65.6987305,38.2716064,65.698730510.0000
61,72.8317871l-35.999939-
36L2.2716064,36L2.2717285,0z"/>
</svg>

</template>

<script>
export default {
  name: 'Left45d',
  components: {
  },

  props: ['passedShape','passedParentShape',
'passedColor'],

  data(){
  return{

  }
  },
}

</script>
```

```
<style lang="scss" scoped>
.focusStyle path {
  stroke: red;
  stroke-width: 2px;
  stroke-dasharray: 5 5;
  opacity: .5;
}
</style>
```

**Left90d_Right90d_main.vue**

```
<template lang="pug">
<svg version="1.1" id="Layer_1"
xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink"
:x="passedParentShape.x - 278.5"
:y="passedParentShape.y - 140" width="600px"
height="148px" viewBox="0 0 600 148"
style="enable-background:new 0 0 600 148;"
xml:space="preserve">
<path :fill="passedColor" d="M526.7808838,0h-
50.9116211l51.5,51.5H392.572876h-
22.9324341h-140h-
27.067627H71.9116211l51.5-
51.5H72.5L0,72.5l25.4558105,25.4560547L72.5
,145h50.9116211l-51.5-
51.5h130.6611938h27.067627c27.0185547,0,49,
21.9804688,49,48.9990234l0,0c0,0.0004883,0,0.
0007324,0,0.0009766l21-
0.0004883l21,0.0004883c0-0.0002441,0-
0.0004883,0-0.0009766l0,0c0-
27.0185547,21.9814453-48.9990234,49-
48.9990234h22.9324341h134.7963867l-
51.5,51.5h50.9116211l47.0441895-
47.0439453L599.2808838,72.5L526.7808838,0z
"/>
```

```
</svg>

</template>

<script>
export default {
  name: 'Left90d_Right90d_main',
  components: {
  },

  props: ['passedShape', 'passedParentShape', 'passedColor'],

  data(){
    return{

    }
  },
}

</script>

<style lang="scss" scoped>
.focusStyle path {
  stroke: red;
  stroke-width: 2px;
  stroke-dasharray: 5 5;
  opacity: .5;
}
</style>
```

**Left90d_Straight0d_Right90d_main.vue**

```
<template lang="pug">
<svg version="1.1" id="Layer_1" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" :x="passedParentShape.x - 278.5" :y="passedParentShape.y - 340" width="600px" height="355px" viewBox="0 0 600 355" style="enable-background:new 0 0 600 355;" xml:space="preserve">
<path :fill="passedColor" d="M526.7808838,205.6806641h-50.9116211l51.5,51.5H369.6404419c-18.0273438,0-34.8469849,5.269043-49,14.3481445V71.9116211l51.5,51.5V72.5l-47.0441284-47.0441895L299.6404419,0l-72.5,72.5v50.9116211l51.5-51.5v199.6171875c-14.1530151-9.0791016-30.9726562-14.3481445-49-14.3481445H71.9116211l51.5-51.5H72.5l-72.5,72.5l25.4558105,25.4558105L72.5,350.6806641h50.9116211l-51.5-51.5h157.7288208c27.0185547,0,49,21.9804688,49,48.9990234l0,0c0,0.0002441,0,0.0004883,0,0.0009766l21-0.0004883l21,0.0004883c0-0.0004883,0-0.0007324,0-0.0009766l0,0c0-27.0185547,21.9814453-48.9990234,49-48.9990234h157.7288208l-51.5,51.5h50.9116211l47.0441895-47.0441895l25.4558105-25.4558105L526.7808838,205.6806641z"/>
</svg>
</template>

<script>
export default {
  name: 'Left90d_Straight0d_Right90d_main',
  components: {
  },
```

```
 props: ['passedShape', 'passedParentShape', 'passedColor'],

 data(){
  return{

  }
 },
}

</script>

<style lang="scss" scoped>
.focusStyle path {
 stroke: red;
 stroke-width: 2px;
 stroke-dasharray: 5 5;
 opacity: .5;
}
</style>
```

**Left90d.vue**

```
<template lang="pug">
//<rect :x="passedParentShape.x - 1" :y="passedShape.y + passedParentShape.y" :height="passedShape.height"
:width="passedShape.width + passedShape.x" transform="scale(-1, 1)" :transform-origin="passedParentShape.x"
:fill="passedColor"/>
<svg version="1.1" id="Layer_1" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink"
:x="passedParentShape.x - passedShape.width + passedParentShape.width" :y="passedShape.y + passedParentShape.y" width="395px"
height="217px" viewBox="0 0 395 217" style="enable-background:new 0 0 395 217;" xml:space="preserve">
<path :fill="passedColor" d="M72.500061,0h50.91162111l-51.5,51.5h157.7288208c90.9814453,0,165,74.0185547,165,165l-42-0.0004883c0-32.6953125-12.8378906-63.5390625-36.1494141-86.8505859C293.180542,106.3378906,262.3358154,93.5,229.6405029,93.5H71.9116821l51.5,51.5H72.500061L25.4558105,97.9559326L0,72.5L72.500061,0z"/>
</svg>

</template>

<script>
export default {
 name: 'Left90d',
 components: {
 },

 props: ['passedShape','passedParentShape', 'passedColor'],

 data(){
  return{

  }
 },
}

</script>

<style lang="scss" scoped>
```

```
.focusStyle path {
  stroke: red;
  stroke-width: 2px;
  stroke-dasharray: 5 5;
  opacity: .5;
}
</style>
```

**Left180d.vue**

```
<template lang="pug">
//<rect :x="passedParentShape.x - 1"
:y="passedShape.y + passedParentShape.y"
:height="passedShape.height"
:width="passedShape.width + passedShape.x"
transform="scale(-1, 1)" :transform-
origin="passedParentShape.x"
:fill="passedColor"/>

<svg version="1.1" id="Layer_1"
xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink"
:x="passedParentShape.x - passedShape.width +
passedParentShape.width + 2"
:y="passedShape.y + passedParentShape.y"
width="375px" height="298px" viewBox="0 0
375 298" style="enable-background:new 0 0 375
298;" xml:space="preserve">
<path :fill="passedColor"
d="M51.2908936,223.7768555V161c0-
88.7758789,72.2246094-161,161-
161s161,72.2241211,161,161h-42c0-
31.6313477-12.4208984-61.472168-
34.9746094-
84.0258789C273.7635498,54.4208984,243.9227
295,42,212.2908936,42s-
61.4726562,12.4208984-
84.0253906,34.9741211C105.711792,99.527832
,93.2908936,129.3686523,93.2908936,161v63.1
950684L145,172.4858398v50.9116211l-
47.0441284,47.0441895L72.5,295.8974609l-
72.5-72.5v-
50.9116211L51.2908936,223.7768555z"/>
</svg>
</template>

<script>
export default {
  name: 'Left180d',
  components: {
  },

  props: ['passedShape','passedParentShape',
'passedColor'],

  data(){
    return{

    }
  },
}

</script>

<style lang="scss" scoped>
.focusStyle path {
  stroke: red;
  stroke-width: 2px;
  stroke-dasharray: 5 5;
  opacity: .5;
}
</style>
```

**Right45d.vue**

```
<template lang="pug">
//<rect :x="passedParentShape.x + passedParentShape.width - 1" :y="passedShape.y + passedParentShape.y" :height="passedShape.height" :width="passedShape.width + passedShape.x" :fill="passedColor"/>

<svg version="1.1" id="Layer_1" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" :x="passedParentShape.x" :y="passedShape.y + passedParentShape.y" width="304px" height="322px" viewBox="0 0 304 322" style="enable-background:new 0 0 304 322;" xml:space="preserve">
<path :fill="passedColor" d="M301.7282715,0H199.1978149l-35.999939,36h72.8317871c0,0-184.3724365,184.2884521-187.6917725,187.5257568C21.3310547,249.8685303,0,270.6751709,0,320.3543701h42c0-31.9750977,11.1767578-42.8769531,35.6640625-66.7626953c3.4111328-3.3276367,188.0643311-187.8929443,188.0643311-187.8929443l-0.000061,72.8317871l35.999939-36L301.7283936,36L301.7282715,0z"/>
</svg>
</template>

<script>
export default {
  name: 'Right45d',
  components: {
  },

  props: ['passedShape','passedParentShape', 'passedColor'],

  data(){
  return{

  }
  },
}

</script>

<style lang="scss" scoped>
.focusStyle path {
  stroke: red;
  stroke-width: 2px;
  stroke-dasharray: 5 5;
  opacity: .5;
}
</style>
```

**Right90d_Left180d.vue**

```
<template lang="pug">
//<rect :x="passedParentShape.x + passedParentShape.width - 1" :y="passedShape.y + passedParentShape.y" :height="passedShape.height" :width="passedShape.width + passedShape.x" :fill="passedColor"/>

<svg version="1.1" id="Layer_1" xmlns="http://www.w3.org/2000/svg"
```

```
xmlns:xlink="http://www.w3.org/1999/xlink"
:x="passedParentShape.x" :y="passedShape.y +
passedParentShape.y" width="640px"
height="498px" viewBox="0 0 640 498"
style="enable-background:new 0 0 640 498;"
xml:space="preserve">
<path :fill="passedColor"
d="M565.4020386,279.7907715h-
50.9116211l51.5,51.5H387.15271c32.3410645-
29.4675293,52.6690674-
71.9038086,52.6690674-119c0-88.7753906-
72.2241211-161-161-161h-
62.7768555L267.3358154,0h-50.9116211l-
72.5,72.5l25.4558105,25.4558105L216.4241943
,145h50.9116211L215.626709,93.2907715h63.1
950684c31.6313477,0,61.472168,12.4208984,84
.0258789,34.9746094c22.5532227,22.5527344,3
4.9741211,52.3935547,34.9741211,84.0253906s
-12.4208984,61.4726562-
34.9741211,84.0253906c-
22.5537109,22.5537109-
52.3945312,34.9746094-
84.0258789,34.9746094H165c-90.9814453,0-
165,74.0185547-165,165l42-0.0004883c0-
32.6953125,12.8378906-
63.5390625,36.1494141-
86.8505859C101.4599609,386.1286621,132.304
6875,373.2907715,165,373.2907715h400.99041
75l-51.5,51.5h50.9116211l47.0441895-
47.0439453l25.4558716-
25.4560547L565.4020386,279.7907715z"/>
</svg>

</template>

<script>
export default {
  name: 'Right90d_Left180d',
  components: {
  },

  props: ['passedShape','passedParentShape',
'passedColor'],

  data(){
    return{

    }
  },
}

</script>

<style lang="scss" scoped>
.focusStyle path {
  stroke: red;
  stroke-width: 2px;
  stroke-dasharray: 5 5;
  opacity: .5;
}
</style>
```

**Right90d.vue**

```
<template lang="pug">
//<rect :x="passedParentShape.x +
passedParentShape.width - 1" :y="passedShape.y
+ passedParentShape.y"
:height="passedShape.height"
:width="passedShape.width + passedShape.x"
:fill="passedColor"/>
```

```
<svg version="1.1" id="Layer_1" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" :x="passedParentShape.x" :y="passedShape.y + passedParentShape.y" width="395px" height="217px" viewBox="0 0 395 217" style="enable-background:new 0 0 395 217;" xml:space="preserve">
<path :fill="passedColor" d="M322.1404419,0h-50.9116211l51.5,51.5H165c-90.9814453,0-165,74.0185547-165,165l42-0.0004883c0-32.6953125,12.8378906-63.5390625,36.1494141-86.8505859C101.4599609,106.3378906,132.3046875,93.5,165,93.5h157.7288208l-51.5,51.5h50.9116211l47.0442505-47.0440674L394.6405029,72.5L322.1404419,0z"/>
</svg>

</template>

<script>
export default {
  name: 'Right90d',
  components: {
  },

  props: ['passedShape','passedParentShape', 'passedColor'],

  data(){
    return{

    }
  },
}

</script>

<style lang="scss" scoped>
.focusStyle path {
  stroke: red;
  stroke-width: 2px;
  stroke-dasharray: 5 5;
  opacity: .5;
}
</style>
```

**Right180d.vue**

```
<template lang="pug">
//<rect :x="passedParentShape.x + passedParentShape.width - 1" :y="passedShape.y + passedParentShape.y" :height="passedShape.height" :width="passedShape.width + passedShape.x" :fill="passedColor"/>

<svg version="1.1" id="Layer_1" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" :x="passedParentShape.x" :y="passedShape.y + passedParentShape.y" width="375px" height="298px" viewBox="0 0 375 298" style="enable-background:new 0 0 375 298;" xml:space="preserve">
<path :fill="passedColor" d="M322,223.7768555V161C322,72.2241211,249.7753906,0,161,0S0,72.2241211,0,161h42c0-31.6313477,12.4208984-61.472168,34.9746094-84.0258789C99.5273438,54.4208984,129.36816
```

```
41,42,161,42s61.4726562,12.4208984,84.02539
06,34.9741211C267.5791016,99.527832,280,12
9.3686523,280,161v63.1950684l-51.7091064-
51.7092285v50.9116211l47.0441284,47.044189
5l25.4558716,25.4558105l72.5-72.5v-
50.9116211L322,223.7768555z"/>
</svg>
</template>

<script>
export default {
  name: 'Right180d',
  components: {
  },

  props:  ['passedShape','passedParentShape',
'passedColor'],

  data(){
    return{

    }
  },
}

</script>

<style lang="scss" scoped>
.focusStyle path {
  stroke: red;
  stroke-width: 2px;
  stroke-dasharray: 5 5;
  opacity: .5;
}
</style>
```

**Right270d.vue**

```
<template lang="pug">
//<rect  :x="passedParentShape.x  +
passedParentShape.width - 1" :y="passedShape.y
+  passedParentShape.y"
:height="passedShape.height"
:width="passedShape.width  +  passedShape.x"
:fill="passedColor"/>

<svg  version="1.1"  id="Layer_1"
xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink"
:x="passedParentShape.x"  :y="passedShape.y  +
passedParentShape.y"  width="455.0670471px"
height="374.6788635px"  viewBox="0  0
455.0670471  374.6788635"  style="enable-
background:new 0 0 455.0670471 374.6788635;"
xml:space="preserve">
<path  :fill="passedColor"
d="M293,0H165C74.0185547,0,0,74.0185547,0,
165l42-0.0004883c0-32.6953125,12.8378906-
63.5390625,36.1494141-
86.8505859C101.4599609,54.8378906,132.3046
875,42,165,42h128c31.6313477,0,61.472168,12.
4208984,84.0258789,34.9746094C399.5791016,
99.5273438,412,129.3681641,412,161s-
12.4208984,61.4726562-
34.9741211,84.0253906C354.472168,267.57910
16,324.6313477,280,293,280h-
57.8242188l51.7091064-51.7092285h-
50.9116211l-47.0441895,47.0441895l-
25.4558105,25.4558105l72.5,72.5h50.9116211L
235.5939941,322H293c88.7758789,0,161-
72.2246094,161-161S381.7758789,0,293,0z"/>
</svg>
```

```
</template>

<script>
export default {
  name: 'Right270d',
  components: {
  },

  props:  ['passedShape','passedParentShape', 'passedColor'],

  data(){
  return{

  }
  },
}

</script>

<style lang="scss" scoped>
.focusStyle path {
  stroke: red;
  stroke-width: 2px;
  stroke-dasharray: 5 5;
  opacity: .5;
}
</style>
```

**Straight0d_main.vue**

```
<template lang="pug">
<svg  :x="passedParentShape.x  -  51.5" :y="passedParentShape.y - 229" width="145px" height="230px"  viewBox="0  0  145  230" xml:space="preserve">
  <polygon  :fill="passedColor"  points="0,123.4 72.5,50.9 145,123.4 145,72.5 98,25.5 72.5,0 0,72.5 "/>
  <rect  x="51.5"  y="40"  :fill="passedColor" :width="42" :height="passedShape.height"/>
</svg>
</template>

<script>
export default {
  name: 'Straight0d_main',
  components: {
  },

  props:  ['passedShape',  'passedParentShape', 'passedColor'],

  data(){
    return{

    }
  },
}

</script>

<style lang="scss" scoped>
.focusStyle path {
  stroke: red;
  stroke-width: 2px;
  stroke-dasharray: 5 5;
  opacity: .5;
}
</style>
```

**Text.vue**

```
<template lang="pug">
<text :x="passedShape.x + passedMainShape.x"
:y="passedShape.y + passedParentShape.y +
passedMainShape.y"
:fill="passedColor">{{passedShape.textContent
}}</text>
</template>

<script>
export default {
  name: 'Text',
  components: {
  },

  props:      ['passedShape','passedParentShape',
'passedMainShape', 'passedColor'],

  data(){
    return{

    }
  },

  methods:
  {

  }

}

</script>

<style lang="scss" scoped>
@import '@/assets/styles/_svg-templates.scss';

  text {
    font-family: RoadUA;
    font-size: 30px;
  }

</style>
```

**TruckSign.vue**

```
<template lang="pug">
<svg version="1.1" id="Layer_1"
xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink"
:x="leftOrRight()" :y="passedShape.y +
passedParentShape.y + passedMainShape.y"
width="126px" height="126px" viewBox="0 0
126 126" style="enable-background:new 0 0 126
126;" xml:space="preserve">
<g class="container">
        <path style="fill:#FFFFFF;"
d="M62.9999962,122.0238647c-
32.5979538,0.0000305-59.0238342-26.425827-
59.0238533-59.0237808c-0.0000093-
15.6541176,6.2185678-30.6670876,17.2877121-
41.7362137c23.0502338-
23.0502338,60.4220505-
23.0502338,83.47229,0s23.0502243,60.4220505
,0,83.4722824C93.6912994,115.8415375,78.662
5748,122.0666504,62.9999962,122.0238647z"/>
        <path style="fill:#DA251D;"
d="M63,10.8620691c28.8041611,0.0221577,52.
1365814,23.3905067,52.1144257,52.1946678c-
0.016098,20.933033-12.5463409,39.8281746-
31.8233414,47.9884148c-
26.5434647,11.185463-57.1288071-1.2646484-
68.3142776-
```

27.8081131C3.803844,56.7232552,16.2128429,
26.1697407,42.7089157,14.9548531C49.128376
,12.2393389,56.0298309,10.8472891,63,10.8620
691M63,0C28.2060566,0,0,28.2060566,0,63s28.
2060566,63,63,63s63-28.2060547,63-
63S97.7939453,0,63,0z"/>
<g>
<path
d="M38.2344818,77.1206894c2.0396461,0,3.69
31038,1.6534576,3.6931038,3.6931c0,2.03965-
1.6534576,3.6931076-3.6931038,3.6931076c-
2.0396423,0-3.6931038-1.6534576-3.6931038-
3.6931076l0,0C34.5436821,78.7751007,36.1957
932,77.1229935,38.2344818,77.1206894
M38.2344818,74.0793076c-3.7193527,0-
6.7344818,3.0151367-
6.7344818,6.7344818c0,3.7193527,3.0151291,6.
7344894,6.7344818,6.7344894s6.7344818-
3.0151367,6.7344818-
6.7344894C44.9689636,77.0944443,41.9538345
,74.0793076,38.2344818,74.0793076z"/>
<path
d="M84.506897,77.1206894c2.03965,0,3.6931,
1.6534576,3.6931,3.6931c0,2.03965-
1.65345,3.6931076-3.6931,3.6931076s-
3.6931076-1.6534576-3.6931076-
3.6931076l0,0C80.8160934,78.7751007,82.4682
007,77.1229935,84.506897,77.1206894M84.506
897,74.0793076c-3.7193527,0-
6.7344818,3.0151367-
6.7344818,6.7344818c0,3.7193527,3.0151291,6.
7344894,6.7344818,6.7344894s6.7344818-
3.0151367,6.7344818-
6.7344894C91.2413788,77.0944443,88.2262497
,74.0793076,84.506897,74.0793076z"/>
<path
d="M48.0103455,73.8620682V55.8310356c-
0.0006485-1.7994156-1.4592018-3.2579727-
3.2586212-3.2586212H32.2168961c-0.7200985-
0.0000191-1.3935642,0.3562393-
1.7987576,0.951519l-6.6106548,9.7324104c-
0.3666134,0.5403786-0.5626183,1.1783409-
0.5626545,1.8313408v12.4674911c0.0006561,1.
799408,1.4592037,3.2579575,3.2586193,3.2586
136h3.5844822c0-4.4992142,3.647337-
8.1465454,8.1465511-
8.1465454s8.146553,3.6473312,8.146553,8.146
5454h29.492691l4.0710983-
6.9517212H48.0103455z
M44.534481,62.5655174c-
0.0004349,1.1996078-0.972805,2.171978-
2.1724129,2.1724129h-11.798378c-
1.2003269,0.0003815-2.1736908-0.9723663-
2.1740685-2.172699c-0.0001354-
0.4289589,0.1266708-0.8483734,0.3644466-
1.205402413.2998981-4.9465828c0.6042442-
0.906601,1.6216545-1.4511833,2.7111702-
1.4511757h7.5969315c1.1996117,0.0004311,2.1
719818,0.9727974,2.1724129,2.1724129V62.56
55174z"/>
<path
d="M53.006897,40.8413811h42.3620682c1.799
6902,0,3.2586212,1.458931,3.2586212,3.258617
4l0,0v27.155174310,0H49.748275810,0V44.099
9985C49.7482758,42.300312,51.2072105,40.84
13811,53.006897,40.8413811L53.006897,40.84
13811z"/>
<path
d="M99.2793121,73.8620682v6.9517212h-
6.7344818c0.0014725-2.8650894-1.5233231-

```
5.5140381-4.0015945-
6.9517212H99.2793121z"/>
          </g>
</g>
</svg>
</template>

<script>
export default {
 name: 'TruckSign',
 components: {
 },

 props:        ['passedShape','passedParentShape',
'passedMainShape', 'passedColor'],

 data(){
  return{

  }
 },

 methods:
 {
  leftOrRight()
  {
   if(this.passedParentShape.facing == "left")
   {
    return          this.passedShape.x          +
this.passedMainShape.x;
   }
   else
   {
    return          this.passedShape.x          +
this.passedMainShape.x;
   }
  }
 }

}

</script>

<style lang="scss" scoped>
.focusStyle .container {
 stroke: red;
 stroke-width: 2px;
 stroke-dasharray: 5 5;
 opacity: .5;
}
</style>
```

**shapesTemplates.json**

```
{
   "shapesMain":
   [
      {"id":      1,"name":      "Straight0d_main",
"nameUI":"Прямий", "x": 0, "y": 0, "height":
190, "width": 145, "parentId": 0, "multiWay":
false},
      {"id": 1,"name": "Left90d_Right90d_main",
"nameUI":"Ліворуч та Праворуч", "x": 0, "y": 0,
"height": 190, "width": 600, "parentId": 0,
"multiWay": true},
      {"id":                              1,"name":
"Left90d_Straight0d_Right90d_main",
"nameUI":"Ліворуч, Прямо, Праворуч", "x": 0,
"y": 0, "height": 190, "width": 600, "parentId": 0,
"multiWay": true}
   ],
```

```
  "shapesSub":
  [
    {"id": null,"name": "Right45d", "nameUI":"Праворуч 45", "x": 0, "y": 50, "height": 322, "width": 304, "parentId": 0, "facing": "right"},
    {"id": null,"name": "Right90d", "nameUI":"Праворуч 90", "x": 0, "y": 50, "height": 217, "width": 395, "parentId": 0, "facing": "right"},
    {"id": null,"name": "Right180d", "nameUI":"Праворуч 180", "x": 0, "y": 50, "height": 298, "width": 375, "parentId": 0, "facing": "right"},
    {"id": null,"name": "Right270d", "nameUI":"Праворуч 270", "x": 0, "y": 50, "height": 455, "width": 456, "parentId": 0, "facing": "right"},
    {"id": null,"name": "Right90d_Left180d", "nameUI":"Праворуч 90, Ліворуч 180", "x": 0, "y": 50, "height": 217, "width": 640, "parentId": 0, "facing": "right"},

    {"id": null,"name": "Left45d", "nameUI":"Ліворуч 45", "x": 0, "y": 50, "height": 322, "width": 304, "parentId": 0, "facing": "left"},
    {"id": null,"name": "Left90d", "nameUI":"Ліворуч 90", "x": 0, "y": 50, "height": 217, "width": 395, "parentId": 0, "facing": "left"},
    {"id": null,"name": "Left180d", "nameUI":"Ліворуч 180", "x": 0, "y": 50, "height": 298, "width": 375, "parentId": 0, "facing": "left"}
  ],

  "shapesAdd":
  [
    {"id": null,"name": "TruckSign", "nameUI":"Рух вантажних автомобілів заборонено", "x": 0, "y": 0, "parentId": null},
    {"id": null,"name": "Text", "nameUI":"Текст", "x": 0, "y": 0, "textContent": "Some text", "parentId": null, "side": null}
  ],

  "textPositions":
  [
    {"name": "0d_main", "positions":[ {"x": -220, "y": -240, "posName": "Ліворуч"}, {"x": 100, "y": -240, "posName": "Праворуч"} ] },
    {"name": "90d_90d_main", "positions":[ {"x": -430, "y": -100, "posName": "Ліворуч"}, {"x": 340, "y": -100, "posName": "Праворуч"} ] },
    {"name": "90d_0d_90d_main", "positions":[ {"x": -430, "y": -100, "posName": "Ліворуч"}, {"x": -210, "y": -335, "posName": "Ліворуч(Пряма стрілка)"}, {"x": 110, "y": -335, "posName": "Праворуч(Пряма стрілка)"}, {"x": 340, "y": -100, "posName": "Праворуч"} ] },

    {"name": "45d", "positions":[ {"x": 324, "y": 25, "posName": "За напрямом"} ] },
    {"name": "90d", "positions":[ {"x": 415, "y": 25, "posName": "За напрямом"} ] },
    {"name": "180d", "positions":[ {"x": 395, "y": 225, "posName": "За напрямом"} ] },
```

{"name": "270d", "positions":[ {"x": 125, "y": 430, "posName": "За напрямом"} ] },
{"name": "90d_180d", "positions":[ {"x": 660, "y": 300, "posName": "За напрямом"}, {"x": 100, "y": -20, "posName": "За додатковим напрямом"} ] }
],

"colorThemes":
[
{"backgroundColor":"blue", "elementsColor":"white"},
{"backgroundColor":"green", "elementsColor":"white"},
{"backgroundColor":"white", "elementsColor":"black"}
],

"signPresets":
{
"Kyiv/Kharkiv/Dnipro":
{
"presetName":"Kyiv/Kharkiv/Dnipro",
"presetNameUI":"Київ, Харків, Дніпро",

"sign":
[

{"id":0,"name":"MainArrowLeg","x":156.640625,"y":395,"height":100,"width":42},

{"id":1,"name":"Straight0d_main","x":0,"y":0,"height":190,"width":145,"parentId":0,"multiWay":false},

{"id":2,"name":"Right90d","x":8,"y":265,"height":217,"width":395,"parentId":0,"facing":"right"},
{"id":3,"name":"Right45d","x":3,"y":-93,"height":322,"width":304,"parentId":0,"facing":"right"},

{"id":4,"name":"TruckSign","x":139,"y":10,"parentId":2},

{"id":5,"name":"Text","x":415,"y":25,"textContent":"Dnipro","parentId":2,"side":"right"},

{"id":6,"name":"Text","x":324,"y":25,"textContent":"Kharkiv","parentId":3,"side":"right"},
{"id":7,"name":"Text","x":-141.640625,"y":-240,"textContent":"Kyiv","parentId":1,"side":"left"}
],

"noShiftSign":
[

{"id":"0c","name":"MainArrowLeg","x":103,"y":395,"height":100,"width":42},

{"id":"1c","name":"Straight0d_main","x":0,"y":0,"height":190,"width":145,"parentId":0,"multiWay":false},

{"id":"2c","name":"Right90d","x":0,"y":50,"height":217,"width":395,"parentId":0,"facing":"right"},

{"id":"3c","name":"Right45d","x":0,"y":50,"height":322,"width":304,"parentId":0,"facing":"right"},

{"id":"4c","name":"TruckSign","x":0,"y":0,"parentId":"2c"},

{"id":"5c","name":"Text","x":415,"y":25,"textContent":"Dnipro","parentId":"2c","side":"right"},

{"id":"6c","name":"Text","x":324,"y":25,"textContent":"Kharkiv","parentId":"3c","side":"right"},
{"id":"7c","name":"Text","x":-141.640625,"y":-240,"textContent":"Kyiv","parentId":"1c","side":"left"}
],

"colorTheme":"1",
"viewBoxScale":"1"
},

"Dnipro/Pidhorodne":
{
"presetName":"Dnipro/Pidhorodne",
"presetNameUI":"Дніпро, Підгородне",

"sign":
[

{"id":0,"name":"MainArrowLeg","x":466.59375,"y":395,"height":100,"width":42},

{"id":1,"name":"Left90d_Right90d_main","x":0,"y":0,"height":190,"width":600,"parentId":0,"multiWay":true},
{"id":2,"name":"Text","x":-451.59375,"y":-100,"textContent":"Pidhorodne","parentId":1,"side":"left"},
{"id":3,"name":"Text","x":340,"y":-100,"textContent":"Dnipro","parentId":1,"side":"right"},
{"id":4,"name":"Right180d","x":-3,"y":118,"height":298,"width":375,"parentId":0,"facing":"right"},

{"id":5,"name":"Text","x":395,"y":225,"textContent":"Zaporizhzhia","parentId":4,"side":"right"},

{"id":6,"name":"TruckSign","x":169,"y":-14,"parentId":4},
{"id":7,"name":"Left90d","x":-11,"y":302,"height":217,"width":395,"parentId":0,"facing":"left"},
{"id":8,"name":"Text","x":-433,"y":25,"textContent":"Lviv","parentId":7,"side":"left"}
],

"noShiftSign":
[

{"id":"0c","name":"MainArrowLeg","x":416,"y":395,"height":100,"width":42},

{"id":"1c","name":"Left90d_Right90d_main","x":0,"y":0,"height":190,"width":600,"parentId":0,"multiWay":true},
{"id":"2c","name":"Text","x":-451.59375,"y":-100,"textContent":"Pidhorodne","parentId":"1c","side":"left"},

{"id":"3c","name":"Text","x":340,"y":-100,"textContent":"Dnipro","parentId":"1c","side":"right"},

{"id":"4c","name":"Right180d","x":0,"y":50,"height":298,"width":375,"parentId":0,"facing":"right"},

{"id":"5c","name":"Text","x":395,"y":225,"textContent":"Zaporizhzhia","parentId":"4c","side":"right"},

{"id":"6c","name":"TruckSign","x":0,"y":0,"parentId":"4c"},

{"id":"7c","name":"Left90d","x":0,"y":50,"height":217,"width":395,"parentId":0,"facing":"left"},
{"id":"8c","name":"Text","x":-433,"y":25,"textContent":"Lviv","parentId":"7c","side":"left"}
],

"colorTheme":"3",
"viewBoxScale":"1"
}
}

**sign.json**

{
"createdSign":
[
{"id": 0,"name": "MainArrowLeg", "x": 0, "y": 395, "height": 100, "width": 42}
],

"noShiftSign":
[
{"id": "0c","name": "MainArrowLeg", "x": 0, "y": 395, "height": 100, "width": 42}
],

"colorTheme": "1",
"viewBoxScale": "1"
}