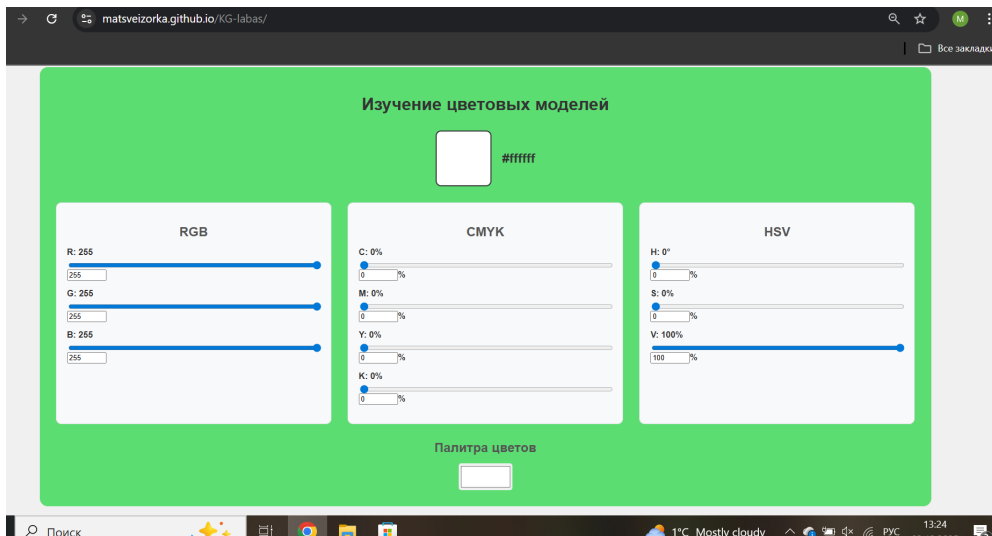


**Вариант 2.** В данной работе проводится изучение цветовых моделей RGB, CMYK, HSV и реализация веб-сайта, который позволяет продемонстрировать работу приложения.

Код реализован на html, css и javascript.

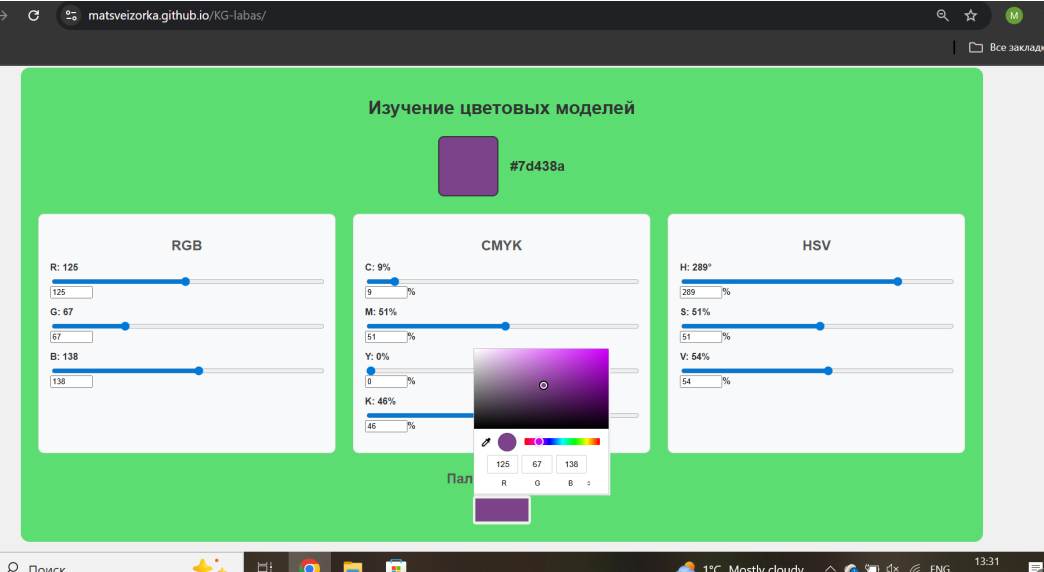
Главная страница приложения:



Возможности приложения:

- крутить ползунки-колёсики и изменять значения цвета во всех требуемых моделях;
- тыкать на кнопки увеличить/уменьшить рядом с числами и изменять цвет;
- вводить значения числовых параметров вручную;
- выбрать цвет из палитры цветов.

Иллюстрация:



Для интерфейса решил сделать блоки div внутри большого зелёного div-а. Числовые значения поместил под ползунки.

*Особенности реализации кода*

1) Для того, чтобы все цвета менялись одновременно, чтобы не прописывать преобразования между всеми возможными парами, сделал преобразование всех моделей в RGB и наоборот. Для палитры преобразовывал в hex. Всё это реализовал как статические функции класса ColorConverter.

При открытии приложения создаётся экземпляр класса ColorApp, у которого вызывается конструктор.

2) Изменение реализовал при помощи многочисленных функций обновления, не забывая про целость параметров и что значения не могут становиться меньше 0, как например тут

```
    this.updateFromCmyk(c, m, y, k);
}

onCmykInputChange() {
    const c = Math.min(100, Math.max(0, parseInt(this.cmykCInput.value) || 0));
    const m = Math.min(100, Math.max(0, parseInt(this.cmykMInput.value) || 0));
    const y = Math.min(100, Math.max(0, parseInt(this.cmykYInput.value) || 0));
    const k = Math.min(100, Math.max(0, parseInt(this.cmykKInput.value) || 0));

    this.updateFromCmyk(c, m, y, k);
}
```

3) Преобразование координат осуществляется по формулам из презентаций. Их все можно посмотреть в js файле. Писать их все сюда не вижу смысла, вот пример одного:

```
static rgbToCmyk(r, g, b) {
    r = r / 255;
    g = g / 255;
    b = b / 255;

    let k = 1 - Math.max(r, g, b);
    if (k === 1) return [0, 0, 0, 100];

    let c = (1 - r - k) / (1 - k);
    let m = (1 - g - k) / (1 - k);
    let y = (1 - b - k) / (1 - k);

    return [
        Math.round(c * 100),
        Math.round(m * 100),
        Math.round(y * 100),
        Math.round(k * 100)
    ];
}
```

*С чем возникли проблемы при написании кода?*

При первой реализации я долго не мог понять, почему при движении некоторых ползунков другие в той же модели двигаются странно. Это возникало из-за того, что я приводил всё к rgb. Условно я изменяю СМΥК, меняется RGB, которое провоцирует изменения СМΥК и так далее. Возникали циклические зависимости. Поборотся с ними оказалось не очень сложно, но вот найти их заняло время.

**Выводы.** Таким образом, я разобрался, как делаются преобразования между различными цветовыми моделями (RGB, СМΥК, HSV в моем случае), а также вспомнил, как можно осуществлять взаимодействие между пользователями при помощи javascript и слушателей и избавляться от циклических зависимостей.