

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
(МОСКОВСКИЙ ПОЛИТЕХ)

Факультет информационных технологий
Кафедра «Инфокогнитивные технологии»

КУРСОВОЙ ПРОЕКТ

на тему: *«Разработка веб-приложения для повышения информированности о
состоянии преступности в регионах России»*

Направление подготовки 09.03.03 «Прикладная информатика»

Профиль «Корпоративные информационные системы»

Выполнил:

студент группы 211-362

Шарков Иван Александрович

19.01.2023

(подпись)

Москва 2023

Введение

Преступность зародилась вместе с обществом и существует по сегодняшний день. Эволюция преступности — это одна из составляющих истории развития общества, охватывающая все существующие сферы жизни людей. Преступность существовала во все времена, изменяясь не только от эпохи к эпохе, от страны к стране, но и, пусть и редко, от региона к региону.

Самая главная проблема преступности — угроза жизни и здоровью граждан Российской Федерации.

В наше время существует немало путей решения проблемы с преступностью, например:

- неотвратимость и суровость наказания;
- достижение невыгодности совершения преступления;
- техническая оснащенность противостоящих органов;
- привлечение международного сообщества;

Но всех этих путей решения недостаточно, ведь преступления совершаются ежедневно, в разных точках мира и в немалых количествах, что подталкивает общество искать новые пути борьбы с данной проблемой.

Веб-приложение «Наша безопасность» — это один из новых видов путей решения проблем с преступностью. Приложение помогает пользователю анализировать открытые данные МВД РФ по совершенным, зарегистрированным и раскрытым преступлениям, отображает статистические данные на графиках для интересующего региона в целях повышения информативности о состоянии преступлений в регионах России, а также предоставляет рекомендации гражданам РФ о том, как избежать и снизить риск столкновения с нарушителями закона.

1 Цель и задачи работы

Целью работы является создание адаптивного, динамического веб-приложения, которое поможет пользователю повысить информационность по состоянию преступности в регионах России. Приложение будет отображать обработанные данные на графиках и предоставлять полезные советы и рекомендации по борьбе с преступностью гражданам Российской Федерации.

Основные этапы разработки приложения:

1. сверстать HTML-структуру основных страниц, описать стили, подключить сторонние вспомогательные библиотеки;
2. создать локальную базу данных, заполнить ее датасетами, настроить подключение с базой данных, протестировать взаимодействие с приложением;
3. выгрузить данные из локальной базы данных в приложение, создать на их основе динамические кнопки для выбора региона и опций графика;
4. написать функции-обработчики, которые обработают открытые данные, преобразуют их в нужный формат и перенесут результат на графики;
5. определить рейтинг самых частых нарушений УК РФ и на его основе предоставить советы и рекомендации для каждого пункта;
6. Разместить приложение на хостинге Московского Политеха.

Исходные открытые данные содержат информацию о зарегистрированных, раскрытых и нераскрытых преступлениях, информацию о преступлениях, по которым имеются потерпевшие, информацию о преступности [1][2][3]. В качестве средств разработки используются языки программирования PHP и JavaScript для динамических вычислений и функционала приложения, язык гипертекстовой разметки HTML и препроцессор SCSS для языка стилей CSS для интерфейса приложения [4].

2 Проектирование приложения

Структура разработанного проекта состоит из компонентов, написанных на языке программирования PHP. Компоненты поделены на смысловые блоки. Блок компонентов-страниц, который отвечает за компоненты, реализующие отдельные страницы веб-интерфейса и блок компонентов-утилит, который отвечает за компоненты, реализующие вспомогательные и вычислительные функции и переменные.

Компоненты веб-приложения:

1. `index.php` – компонент-страница, являющийся титульной страницей приложения, которая содержит информацию о проекте, ссылки на использованные открытые данные, а также кнопку для перехода к работе на главной странице приложения;
2. `main.php` – компонент-страница, являющийся главной страницей приложения, которая содержит графики, динамические кнопки для выбора региона и соответствующей для данного региона опции. Содержит JavaScript-код для динамического создания и настройки графиков на странице. Для реализации графиков используется библиотека `chart.js`. Данные на графики динамически подставляются при помощи PHP-кода;
3. `recommends.php` – компонент-страница, являющийся страницей с пятью наиболее часто нарушаемыми статьями УК РФ, содержащими ссылки на рекомендации по каждому пункту, а также ссылки на использованную литературу. Список статей формируется динамически в зависимости от ранее выбранного региона на главной странице. Каждый элемент списка является ссылкой на страницу с рекомендациями по данной статье;
4. `article_recommends.php` – компонент-страница, являющийся страницей с рекомендацией по конкретной статье уголовного кодекса с ссылками

на использованную литературу. Данные динамически подставляются в зависимости от выбранной ранее статьи.

5. stats.php – компонент-утилита, содержащий основные вычислительные функции для обработки открытых данных. Содержит функции для подсчета количества данных и их процентного отношения для общей статистики, функции для подсчет количества данных и их процентного отношения для конкретного датасета, функцию для вычисления стандартного отклонения для каждого из показателей конкретного датасета;
6. config.php – компонент-утилита, служащий для подключения приложения к базе данных;
7. querys.php – компонент-утилита, содержащий основные запросы к базе данных;
8. regions.php – компонент-утилита, обрабатывающий открытые данные для создания массива-списка регионов Российской Федерации;
9. articles.php – компонент-утилита, содержащий статические переменные с рекомендациями для каждой статьи уголовного кодекса.

Диаграмма компонентов, отображающая специфику зависимостей между компонентами веб-приложения, написана на языке диаграмм UML. Пунктирные стрелки обозначают зависимость от компонента. Компонент index ссылается на main с помощью кнопки на главном экране. Компонент главной страницы main использует компонент querys для доступа ко всем видам запросов, компонент config для настройки подключения к базе данных, компонент stats, который содержит основные вычислительные функции и компонент regions для доступа к массиву всех регионов. Компонент с вычислительными функциями stats использует config и regions. Компонент страницы с рекомендациями использует компонент со статическими советами article_recommends, а также config, literature и regions. Сам компонент рекомендаций также использует компонент с литературой. Диаграмма представлена на рисунке 1.

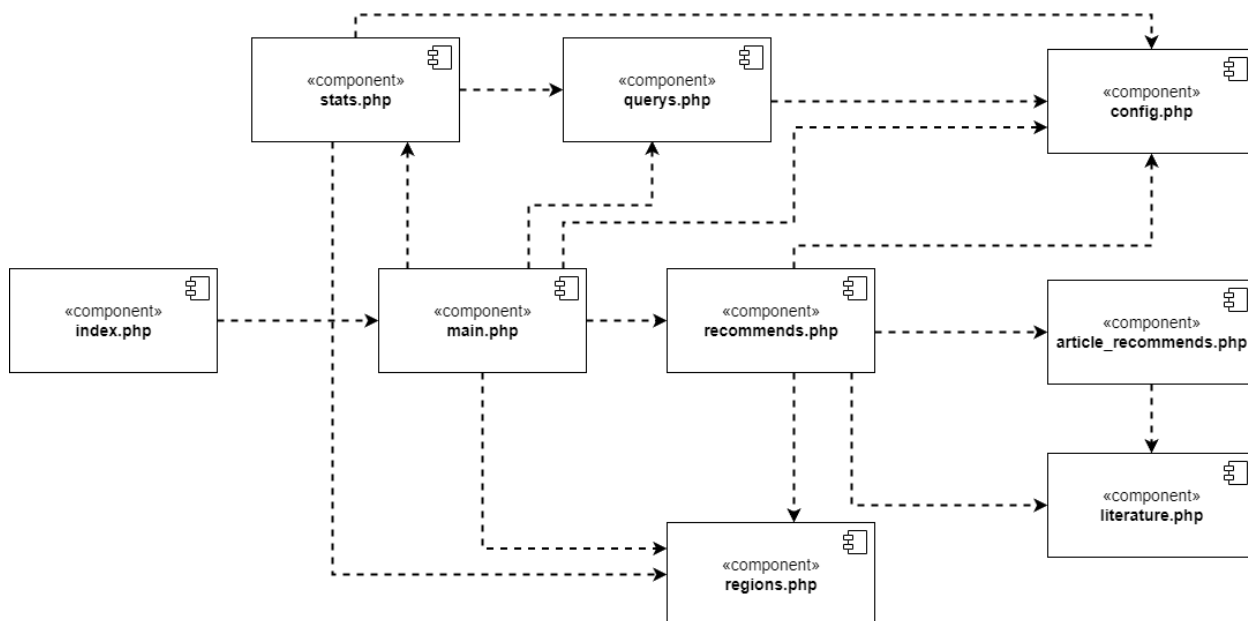


Рисунок 1 — Диаграмма компонентов веб-приложения

Для хранения открытых данных была выбрана база данных MySQL. Перед импортом данных были исправлены опечатки с лишними пробелами в csv файлах, а также первая строка данных была сразу подкорректирована под желаемые названия атрибутов. В результате получились три независимые отношения. Реляционная модель базы данных, содержащая три независимых отношения, представлена на рисунке 2.

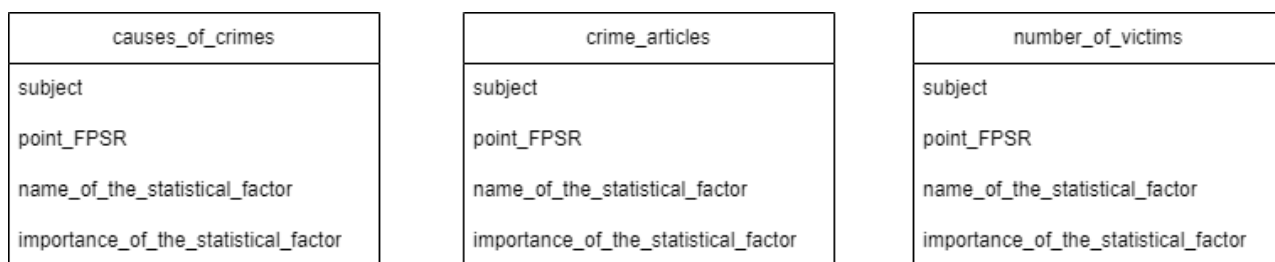


Рисунок 2 – Реляционная модель базы данных

Для разработки интерфейса использовался препроцессор SCSS для более быстрого и удобного написания стилей, а также построения понятной и красивой архитектуры. Также при разработке использовалась популярная

библиотека готовых стилей bootstrap5 [5]. Основное предназначение библиотеки пришлось на использование адаптивной сетки для приложения. Приложение имеет динамический задний фон, написанный на языке JavaScript с помощью встроенного класса Canvas [6]. Веб-приложение полностью адаптировано под мобильные устройства с помощью медиа запросов и сетки bootstrap5.

3 Реализация приложения

Первым этапом разработки приложения стала необходимость создания удаленного репозитория на платформе GitHub, подключения созданного проекта к базе данных MySQL, настройки стартового шаблона для начала проектирования. Верстка выполнена при помощи технологии «flex» с использованием библиотеки bootstrap5, которая предоставила адаптивную сетку, также использовались процентные размеры для блоков, что сделало верстку «резиновой», а препроцессор SCSS помог выстроить качественную и понятную для других разработчиков архитектуру стилей всего веб-приложения благодаря поддержке модульного подключения. Архитектура стилей построена на разбиении файлов на смысловые блоки и вложенности.

После простых обработок открытых данных, таких как удаление лишних пробелов, датасеты были выгружены в базу данных в независимые отношения (см. рисунок 2).

Далее были созданы все необходимые страницы в формате .php из сверстаных html-файлов с подключенными библиотеками bootstrap5 и chart.js. Прописаны дополнительные медиа запросы для адаптации верстки под мобильные устройства. Код для настройки гистограмм представлен в тегах «script» в компоненте главной страницы main.php. Базовый шаблон с начальными настройками был использован из документации библиотеки chart.js. В данном шаблоне, из косметических настроек, были изменены

только цвета столбцов диаграмм и убраны подписи на оси OX, поскольку они ломали верстку приложения так, что графики сужались до нечитаемых масштабов.

После успешной настройки подключения приложения к базе данных в компоненте `config.php` с помощью функций, предоставленных драйвером `mysqli`, с помощью `SELECT`-запроса к одному из датасетов был сформирован массив со всеми регионами России. Массив регионов был выгружен циклом перебора на страницу в качестве кнопок для выбора, отсортированных в алфавитном порядке с помощью функции `array_sort`, в окне с фиксированной высотой.

Следующий этап был написанием основных вычислительных функций для обработки открытых данных. В результате было написано 5 функций, которые высчитывали количественные значения атрибутов региона, их процентное соотношение и стандартное отклонение по каждому из них. Для статистической обработки открытых данных используются функции, реализованные в компоненте `stats.php`.

Функция `count_general_statistics` принимает название региона параметром, высчитывает общее количество атрибутов со всех датасетов, содержимое которых было выгружено при помощи `SELECT`-запросов в локальные переменные. Далее каждый из датасетов итерируется циклом `while`, формируя массив данных формата, согласно формуле (1), где `key` – название статистического фактора, `value` – значение статистического фактора (см. приложение А листинг – 1).

$$\text{Array (key => value)} \quad (1)$$

Функция `count_general_statistics_percent` принимает название региона параметром, высчитывает процентное соотношение атрибутов со всех датасетов, содержимое которых было выгружено при помощи `SELECT`-

запросов в локальные переменные. С помощью цикла `while` высчитывается общая сумма всех показателей со всех открытых данных, после чего, вторым циклом `while` подсчитывается сумма для каждого показателя конкретного датасета, а последний цикл `foreach` помогает высчитать процентное соотношение сумм, согласно формуле (2), где x – процент, y – сумма, n – количество и возвращает массив данных формата, согласно формуле (1) (см. приложение А листинг – 2).

$$x = \frac{y}{n} * 100 \quad (2)$$

Функция `count_quantitative_values` принимает название региона и датасета параметром, выгружает исходные открытые данные конкретного региона из конкретного датасета для выбранной опции с помощью `SELECT`-запроса к выбранному датасету. Результат запроса итерируется циклом `while`, приводя данные к виду, согласно формуле (1) (см. приложение А листинг – 3).

Функция `count_percent_values` принимает название региона и датасета параметром, высчитывает процентное соотношение показателей для конкретного региона из конкретного датасета для выбранной опции путем выгрузки результата `SELECT`-запроса. Первый цикл `while` помогает подсчитать сумму по каждому показателю, а второй высчитать процентное соотношение, согласно формуле (2), где s – сумма, n – количество и возвращает массив данных формата, согласно формуле (1) (см. приложение А листинг – 4).

Функция `count_dispersion` принимает название датасета параметром, высчитывает стандартное отклонение по каждому атрибуту для конкретных открытых путем выгрузки результата `SELECT`-запроса. С помощью первой итерации циклом `while` подсчитывается сумма по каждому статистическому

фактору со всех регионов, кроме общих данных по России. Второй цикл помогает высчитать средние значения по каждому показателю. Финальный цикл итерирует результат запроса с целью подсчитать стандартное отклонение по каждому показателю, согласно формуле (3), где D – дисперсия, x – значение показателя, μ – среднее значение всех данных, n – количество данных и возвращает массив формата, согласно формуле (1) (см. приложение А листинг – 5).

$$\sqrt{D} = \frac{\sum(x-\mu)^2}{n} \quad (3)$$

Заключительный этап стал созданием рейтинга наиболее часто нарушаемых уголовных кодексов в конкретном регионе. Результат SELECT-запроса для отношения с датасетом, содержащий статистику нарушений уголовных кодексов, был отсортирован по количеству значений каждого атрибута с помощью функции `array_sort` и обрезан до 5-го элемента включительно с помощью функции `array_slice`. Элементы рейтинга являются ссылками на страницу `article_recommends.php` с советами и рекомендациями по выбранному уголовному кодексу.

Разработанный и протестированный проект был выложен на хостинг Московского Политеха для общего доступа.

4 Основные сценарии использования приложения

Интерфейс титульной страницы содержит краткое описание веб-приложения, ссылки на использованные открытые данные, а также кнопку «Выбрать регион», которая перемещает пользователя на главную страницу приложения. Интерфейс титульной страницы представлен на рисунке 3.

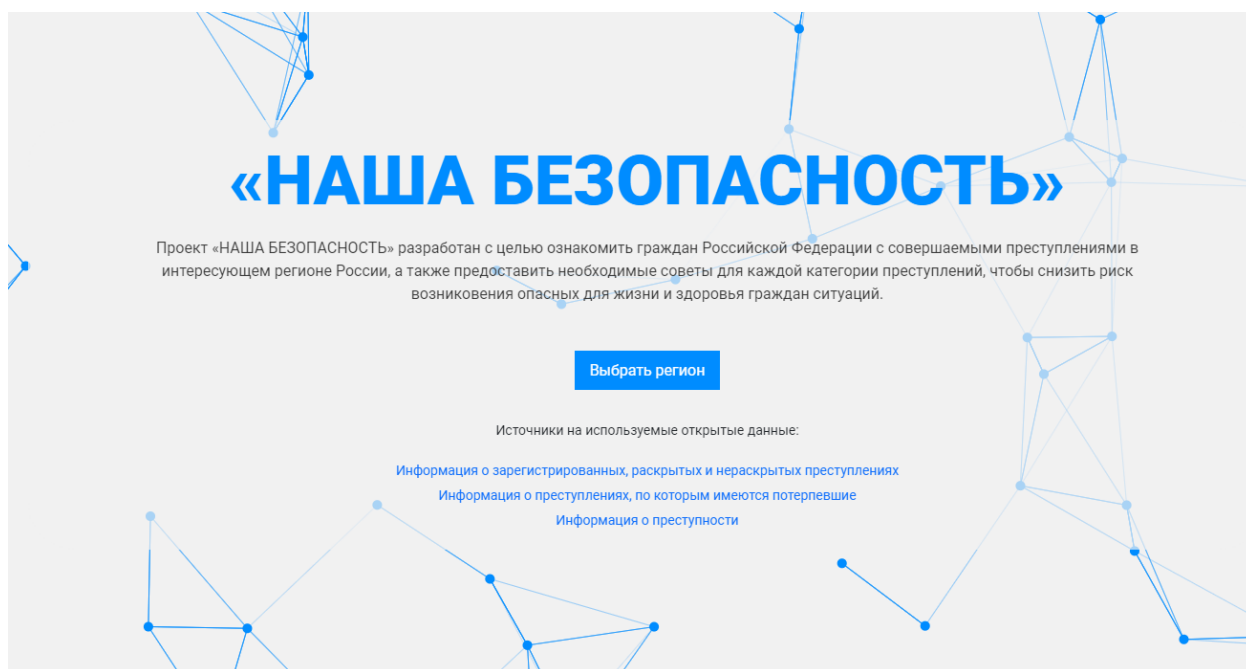


Рисунок 3 – Интерфейс титульной станицы веб-приложения

После нажатия на кнопку «Выбрать регион», пользователь перемещается на главную страницу. Главная страница содержит три основных блока: блок с кнопками для выбора региона, блок с динамическими графиками, блок с доступными для выбора опциями. При отсутствии выбранных параметров, графики не отображаются.

Для общей статистики представлен график общей статистики на основании всех датасетов, а также их процентное соотношение. Для каждого конкретного датасета реализуется три графика: количественные показатели, процентное соотношение данных показателей и стандартное отклонение по каждому из атрибутов. При наведении на элемент графика отображается текст с названием атрибута и его количества. Интерфейс главной страницы с невыбранными параметрами представлен на рисунке 4.

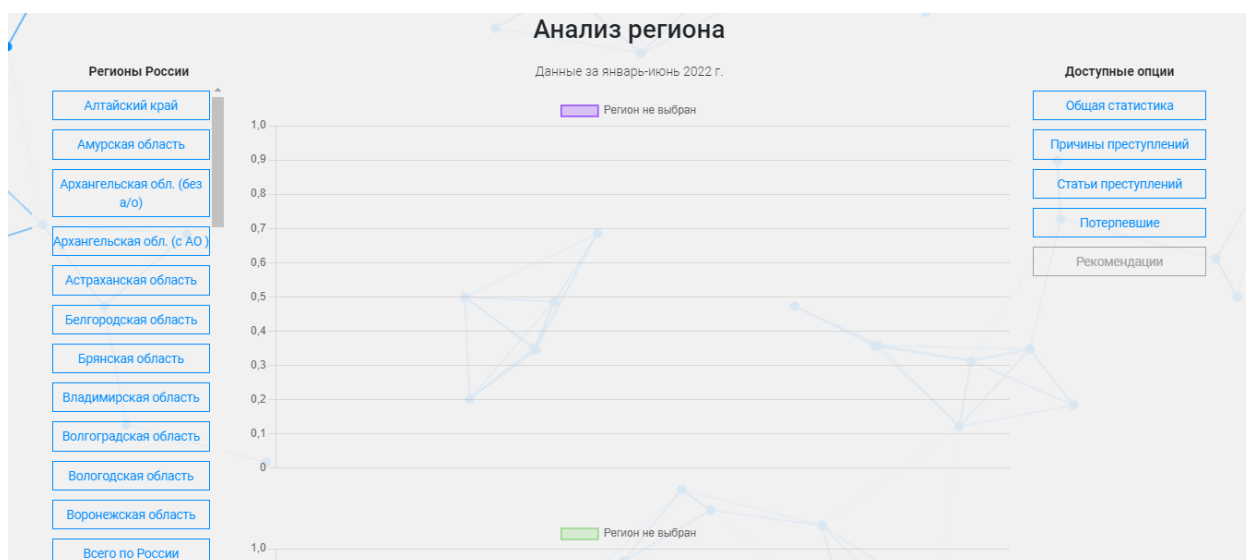


Рисунок 4 – Интерфейс главной страницы веб-приложения с невыбранными параметрами

При выборе необходимых параметров пользователем, динамически строятся статистические графики, а также становится доступным для нажатия кнопка «Рекомендации» для выбранного региона. Интерфейс с выбранным регионом и соответствующей для него опции представлен на рисунке 5 и рисунке 6.

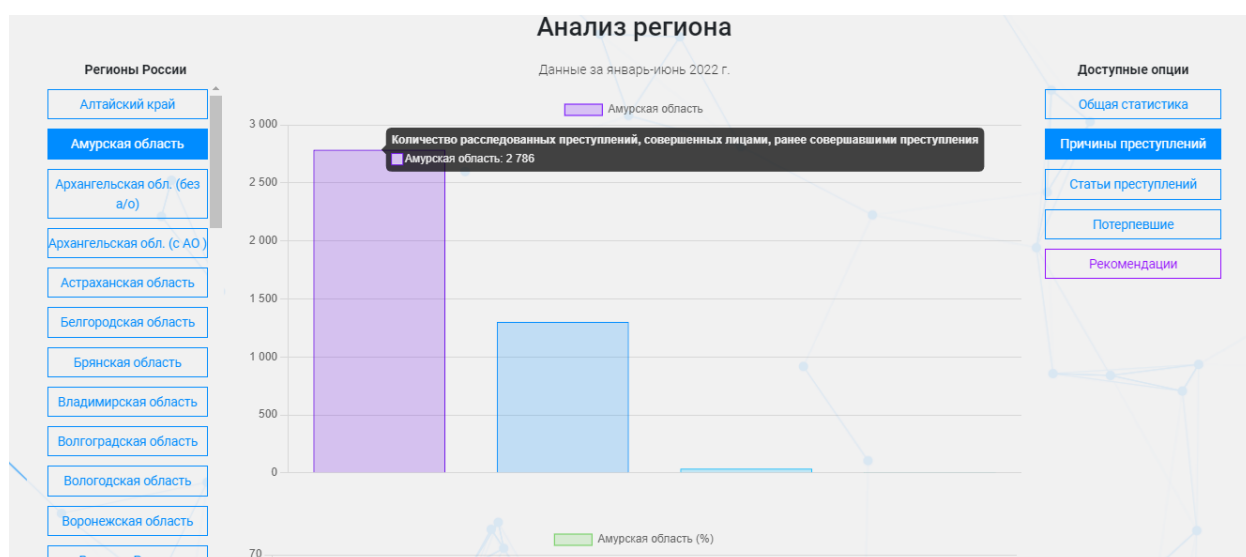


Рисунок 5 – Интерфейс главной страницы веб-приложения с выбранными параметрами

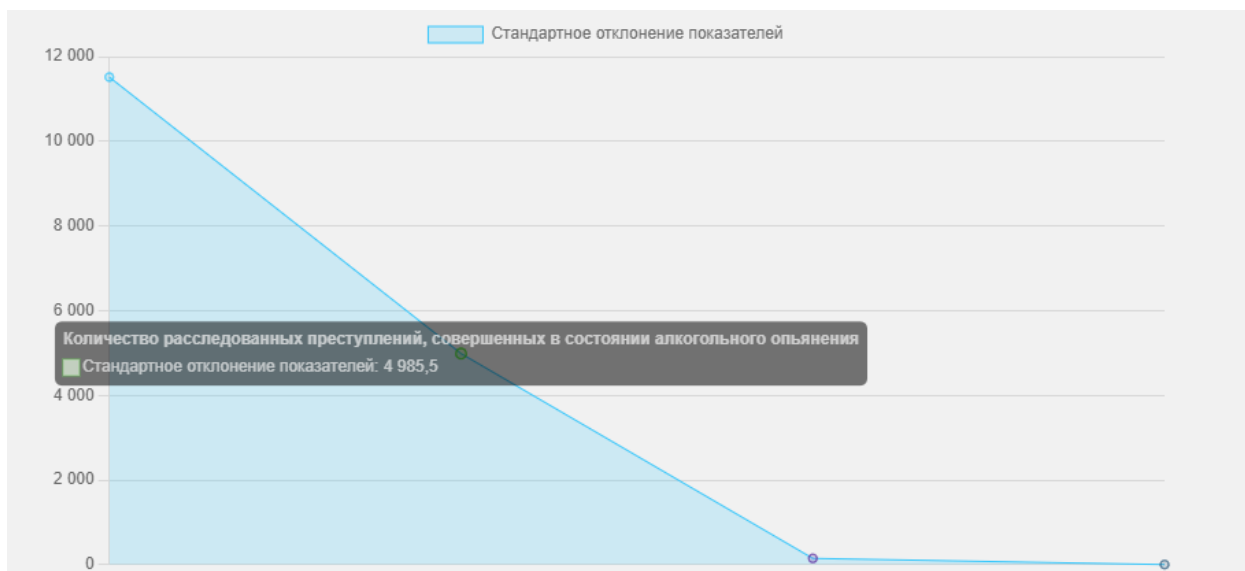


Рисунок 6 – График стандартного отклонения показателей

При нажатии на кнопку «Рекомендации» пользователь перейдет на страницу со списком наиболее часто нарушаемых статей по ранее выбранному региону. Каждый элемент списка является ссылкой на страницу с рекомендациями по данному элементу. Также снизу приведены источники на литературу, используемую для написания советов, и рекомендации. Интерфейс страницы с наиболее часто нарушаемыми статьями представлен на рисунке 7.

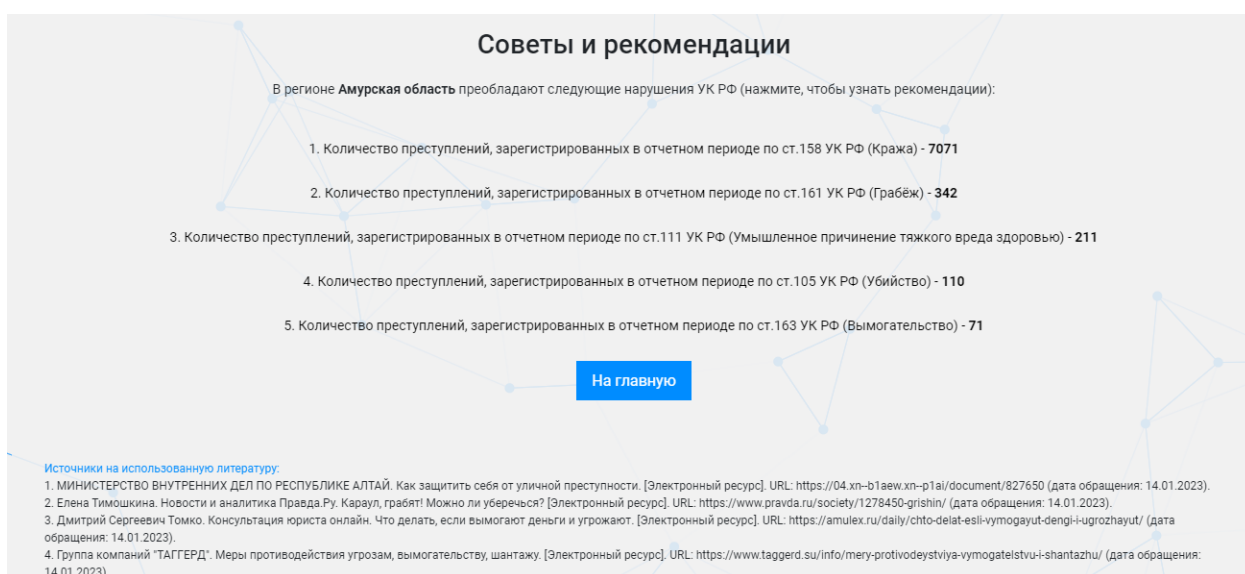


Рисунок 7 – Интерфейс страницы с наиболее часто нарушаемыми статьями

После выбора интересующей статьи, пользователь попадает на страницу с рекомендациями и советами по данному уголовному кодексу. Интерфейс страницы с рекомендациями представлен на рисунке 8.

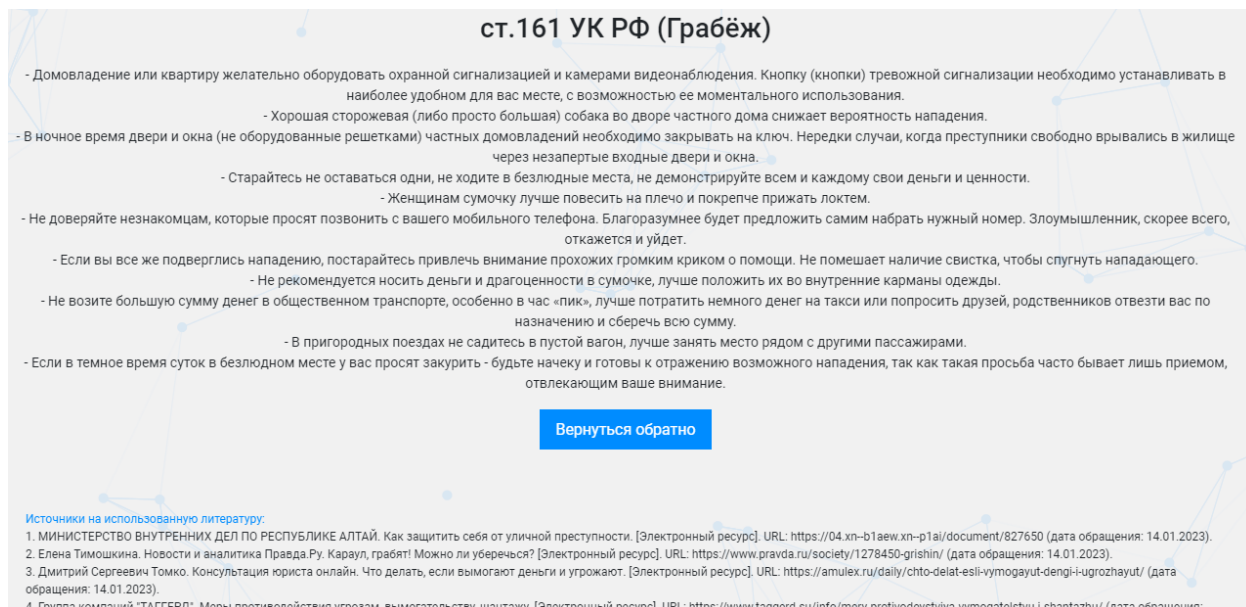


Рисунок 8 – Интерфейс страницы с рекомендациями по выбранной статье

Заключение

Результатом работы является динамическое и адаптивное веб-приложение, способное обрабатывать исходные открытые данные по статистике преступлений в регионах России, а также демонстрирующее результаты обработки на динамических графиках и рекомендации для снижения риска столкновения с противоправными ситуациями. Проект внес свой вклад в борьбу с преступностью в стране. В будущем приложение будет дорабатываться, расширяться и улучшаться. Планируется внедрить еще больше графиков со статистической обработкой открытых данных в целях повышения информационной ценности проекта.

Ссылка на веб-приложение: <http://our-safety.std-2021.ist.mospolytech.ru/>.

Ссылка на удаленный репозиторий проекта: <https://github.com/scharkoff/our-safety>.

Список литературы и интернет-ресурсов

1. Информация о зарегистрированных, раскрытых и нераскрытых преступлениях. [Электронный ресурс]. URL: <https://xn--b1aew.xn--p1ai/opendata/7727739372-MVDGIAC38> (дата обращения 10.01.2023).
2. Информация о преступлениях, по которым имеются потерпевшие. [Электронный ресурс]. URL: <https://xn--b1aew.xn--p1ai/opendata/7727739372-MVDGIAC310> (дата обращения 10.01.2023).
3. Информация о преступности. [Электронный ресурс]. URL: <https://xn--b1aew.xn--p1ai/opendata/7727739372-MVDGIAC33> (дата обращения 10.01.2023).
4. Информация о зарегистрированных, раскрытых и нераскрытых преступлениях. [Электронный ресурс]. URL: <https://xn--b1aew.xn--p1ai/opendata/7727739372-MVDGIAC38> (дата обращения 10.01.2023).
5. Введение. [Электронный ресурс]. URL: <https://bootstrap5.ru/docs/getting-started/introduction> (дата обращения 10.01.2023).
6. Beautiful HTML5 Charts & Graphs. [Электронный ресурс]. URL: <https://canvasjs.com/> (дата обращения 10.01.2023).

Приложение А

(справочное)

Программный код основных вычислительных функций

1. Листинг 1 – Функция вычисления общей статистики.

```
function count_general_statistics($region) {
    $result = array();
    global $connect;
    global $crime_articles;
    global $causes_of_crimes;
    global $number_of_victims;
    // -- Return arrow to start of query string result
    mysqli_data_seek($crime_articles, 0);
    mysqli_data_seek($causes_of_crimes, 0);
    mysqli_data_seek($number_of_victims, 0);
    // -- Sum total amount for every name of the statistical
    factor of the current region:
    // -- Add to result array new values [$key => name of the
    statistical factor, $value => sum of this statistical factor]
    while ($row = mysqli_fetch_assoc($number_of_victims)) {
        if (preg_replace('/\s+/', '', $row["subject"]) ==
        $region) {
            if (isset($result["Общее количество потерпевших"]))
            {
                $result["Общее количество потерпевших"] +=
                $row["importance_of_the_statistical_factor"];
            } else {
                $result["Общее количество потерпевших"] =
                $row["importance_of_the_statistical_factor"];
            }
        }
    }
    // -- Add to result array new values [$key => name of the
    statistical factor, $value => sum of this statistical factor]
    while ($row = mysqli_fetch_assoc($causes_of_crimes)) {
        if (preg_replace('/\s+/', '', $row["subject"]) ==
        $region) {
            if (isset($result["Алкогольные, токсические,
            наркотические опьянения или лица, ранее совершавшие
            преступления"])) {
                $result["Алкогольные, токсические, наркотические
                опьянения или лица, ранее совершавшие преступления"] +=
                $row["importance_of_the_statistical_factor"];
            } else {
                $result["Алкогольные, токсические, наркотические
                опьянения или лица, ранее совершавшие преступления"] =
                $row["importance_of_the_statistical_factor"];
            }
        }
    }
}
```

```

    }
}
}
// -- Add to result array new values [$key => name of the
statistical factor, $value => sum of this statistical factor]
while ($row = mysqli_fetch_assoc($crime_articles)) {
    if (preg_replace('/\s+/', '', $row["subject"]) ==
$region) {
        if (isset($result["Общее число нарушений УК РФ"])) {
            $result["Общее число нарушений УК РФ"] +=
$row["importance_of_the_statistical_factor"];
        } else {
            $result["Общее число нарушений УК РФ"] =
$row["importance_of_the_statistical_factor"];
        }
    }
}
return $result;
}

```

2. Листинг 2 – Функция вычисления общей статистики в процентном соотношении.

```

function count_general_statistics_percent($region) {
    $result = array();
    global $connect;
    global $crime_articles;
    global $causes_of_crimes;
    global $number_of_victims;
    // -- Return arrow to start of query string result
    mysqli_data_seek($crime_articles, 0);
    mysqli_data_seek($causes_of_crimes, 0);
    mysqli_data_seek($number_of_victims, 0);
    $total_count = 0;
    // -- Total sums:
    while ($row = mysqli_fetch_assoc($crime_articles)) {
        if (preg_replace('/\s+/', '', $row["subject"]) ==
$region) {
            $total_count +=
$row["importance_of_the_statistical_factor"];
        }
    }
    while ($row = mysqli_fetch_assoc($causes_of_crimes)) {
        if (preg_replace('/\s+/', '', $row["subject"]) ==
$region) {
            $total_count +=
$row["importance_of_the_statistical_factor"];
        }
    }
}

```

```

    }
    while ($row = mysqli_fetch_assoc($number_of_victims)) {
        if (preg_replace('/\s+/', '', $row["subject"]) ==
$region) {
            $total_count +=
$row["importance_of_the_statistical_factor"];
        }
    }
    // -- Return arrow to start of query string result
    mysqli_data_seek($crime_articles, 0);
    mysqli_data_seek($causes_of_crimes, 0);
    mysqli_data_seek($number_of_victims, 0);
    // -- Count total values for every name of statistical
factor of current region:
    while ($row = mysqli_fetch_assoc($number_of_victims)) {
        if (preg_replace('/\s+/', '', $row["subject"]) ==
$region) {
            if (isset($result["Общее количество потерпевших"]))
{
                $result["Общее количество потерпевших"] +=
$row["importance_of_the_statistical_factor"];
            } else {
                $result["Общее количество потерпевших"] =
$row["importance_of_the_statistical_factor"];
            }
        }
    }
    while ($row = mysqli_fetch_assoc($causes_of_crimes)) {
        if (preg_replace('/\s+/', '', $row["subject"]) ==
$region) {
            if (isset($result["Алкогольные, токсические,
наркотические опьянения или лица, ранее совершавшие
преступления"])) {
                $result["Алкогольные, токсические, наркотические
опьянения или лица, ранее совершавшие преступления"] +=
$row["importance_of_the_statistical_factor"];
            } else {
                $result["Алкогольные, токсические, наркотические
опьянения или лица, ранее совершавшие преступления"] =
$row["importance_of_the_statistical_factor"];
            }
        }
    }
    while ($row = mysqli_fetch_assoc($crime_articles)) {
        if (preg_replace('/\s+/', '', $row["subject"]) ==
$region) {
            if (isset($result["Общее число нарушений УК РФ"])) {
                $result["Общее число нарушений УК РФ"] +=
$row["importance_of_the_statistical_factor"];
            } else {
                $result["Общее число нарушений УК РФ"] =
$row["importance_of_the_statistical_factor"];
            }
        }
    }

```

```

    }
}
// -- Count percent values for every region:
foreach ($result as $key => $value) {
    $result[$key] = round($value / $total_count, 3) * 100;
}
return $result;
}

```

3. Листинг 3 – Функция преобразования исходных данных конкретного датасета в формат для подстановки на графики.

```

function count_quantitative_values($region, $query) {
    $result = array();
    global $connect;
    mysqli_data_seek($query, 0);
    // -- Create array of data [$key => region, $value => amount
of the statistical factor]
    while ($row = mysqli_fetch_assoc($query)) {
        if (preg_replace('/\s+/', '', $row["subject"]) ==
$region) {
            $result[$row["name_of_the_statistical_factor"]] =
$row["importance_of_the_statistical_factor"];
        }
    }
    return $result;
}

```

4. Листинг 4 – Функция для вычисления процентного соотношения конкретного датасета.

```

function count_percent_values($region, $query) {
    $result = array();
    global $connect;
    // -- Return arrow to start of query string result
    mysqli_data_seek($query, 0);
    $total_sum = 0;
    // -- Count sum of all statistical factors for region
    while ($row = mysqli_fetch_assoc($query)) {
        if (preg_replace('/\s+/', '', $row["subject"]) ==
$region) {
            $total_sum +=

```

```

$row["importance_of_the_statistical_factor"];
    }
}
// -- Return arrow to start of query string result
mysqli_data_seek($query, 0);

// -- Create array of data [$key => region, $value =>
percent of the statistical factor]
while ($row = mysqli_fetch_assoc($query)) {
    if (preg_replace('/\s+/', '', $row["subject"]) ==
$region) {
        $result[$row["name_of_the_statistical_factor"]] =
round($row["importance_of_the_statistical_factor"] / $total_sum,
3) * 100;
    }
}
return $result;
}

```

5. Листинг 5 – Функция вычисления стандартного отклонения для конкретного датасета.

```

function count_dispersion($query) {
    $result = array();
    global $connect;
    global $regions;
    // -- Return arrow to start of query string result
    mysqli_data_seek($query, 0);
    $total_sum = 0;
    $total_count = 0;
    $total_sum_of_the_statistical_factors = array();
    $average_of_the_statistical_factors = array();
    $dispersions = array();
    // -- Count total sum and count of the statistical factor
    while ($row = mysqli_fetch_assoc($query)) {
        if ($row["subject"] != "Всего по России") {
            if
(isset($total_sum_of_the_statistical_factors[$row["name_of_the_s
tatistical_factor"]])) {

$total_sum_of_the_statistical_factors[$row["name_of_the_statisti
cal_factor"]] += $row["importance_of_the_statistical_factor"];
            } else {

$total_sum_of_the_statistical_factors[$row["name_of_the_statisti
cal_factor"]] = $row["importance of the statistical factor"];

```

```

        }
    }
}
// -- Count average for each statistical factor
foreach ($total_sum_of_the_statistical_factors as $key =>
$value) {
    $average_of_the_statistical_factors[$key] =
round($total_sum_of_the_statistical_factors[$key] /
(count($regions) - 1), 2);
}
// -- Return arrow to start of query string result
mysqli_data_seek($query, 0);
// -- Count dispersion for each statistical factor
$numerators = array();
$result = 0;
while ($row = mysqli_fetch_assoc($query)) {
    if ($row["subject"] != "Всего по России") {
        foreach ($average_of_the_statistical_factors as $key
=> $value) {
            if ($key ==
$row["name_of_the_statistical_factor"]) {
                if (isset($numerators[$key])) {
                    $numerators[$key] +=
pow($row["importance_of_the_statistical_factor"] - $value, 2);
                    $result = sqrt($numerators[$key] /
(count($regions) - 1));
                    $dispersions[$key] = round($result, 2);
                } else {
                    $numerators[$key] =
pow($row["importance_of_the_statistical_factor"] - $value, 2);
                }
            }
        }
    }
}
return $dispersions;
}

```