

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 информированности 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 динамического веб-приложения, которое поможет пользователю повысить информированность 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 задачи для 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

Функционал приложения поможет пользователю проанализировать интересующий его регион Российской Федерации в целях повышения информированности, а также изучить советы и рекомендации по самым часто нарушаемым статьям уголовного кодекса. Для анализа региона, необходимо выбрать в левом меню приложения интересующий регион, а в меню справа – желаемый тип статистики. Приложение поможет проанализировать данные на основе выбора пользователя и продемонстрирует результат на диаграммах. Графики содержат исходную информацию из датасета, ее процентное соотношение, а также стандартное отклонение.

Диаграмма вариантов использования, написанная на языке диаграмм UML с помощью стороннего сервиса `app.diagrams`, содержит взаимодействие между пользователем и вариантами использования в приложении. Сплошные стрелки без наконечника означают ассоциативную связь. Пользователь может выбрать желаемый регион, интересующую для него опцию и на основе его выбора сформируются гистограммы. Также пользователь может посмотреть рекомендации по выбранному региону на основе рейтинга пяти наиболее часто совершаемых преступлений в данной области. Данная диаграмма вариантов использования представлена на рисунке 3.

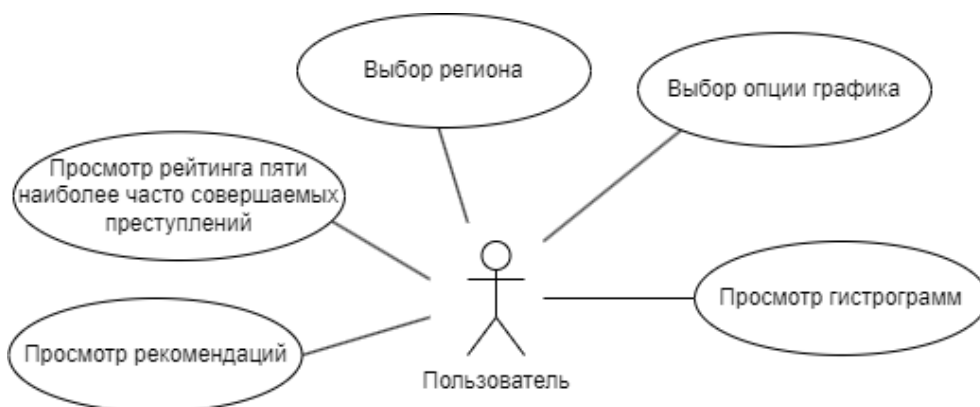


Рисунок 1 – Диаграмма вариантов использования

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.

Для формирования титульной страницы приложения используется компонент `index.php`, который содержит информацию о проекте, ссылки на использованные открытые данные, а также кнопку для перехода к работе на главной странице приложения.

Для формирования главной страницы приложения используется компонент `main.php`, который содержит графики, динамические кнопки для выбора региона и соответствующей для данного региона опции. Также 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-скрипта.

Для формирования страницы с пятью наиболее часто нарушаемых статьями УК РФ используется компонент `recommends.php`, содержащий 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.

Для формирования страницы с рекомендациями по конкретной статье используется компонент `article_recommends.php` с приведенными ниже ссылками на источники использованной литературы. Данные динамически подставляются в зависимости от выбранной ранее статьи.

Основные вычислительные функции для обработки открытых данных реализуются в компонент-утилите `stats.php`. Компонент 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.

Реализация подключения приложения к базе данных происходит в компонент-утилите `config.php`.

Список запросов ко всем основным запросам к базе данных содержится в компонент-утилите `querys.php`.

Список всех доступных регионов Российской Федерации формируется в массиве в компонент-утилите `regions.php`.

Статические рекомендации и советы для каждой статьи уголовного кодекса Российской Федерации представлены в компонент-утилите `articles.php`.

Диаграмма компонентов, отображающая специфику зависимостей между компонентами веб-приложения, написана на языке диаграмм UML с помощью стороннего сервиса `app.diagrams` [5]. 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 компонентов веб-приложения представлена на рисунке 1.

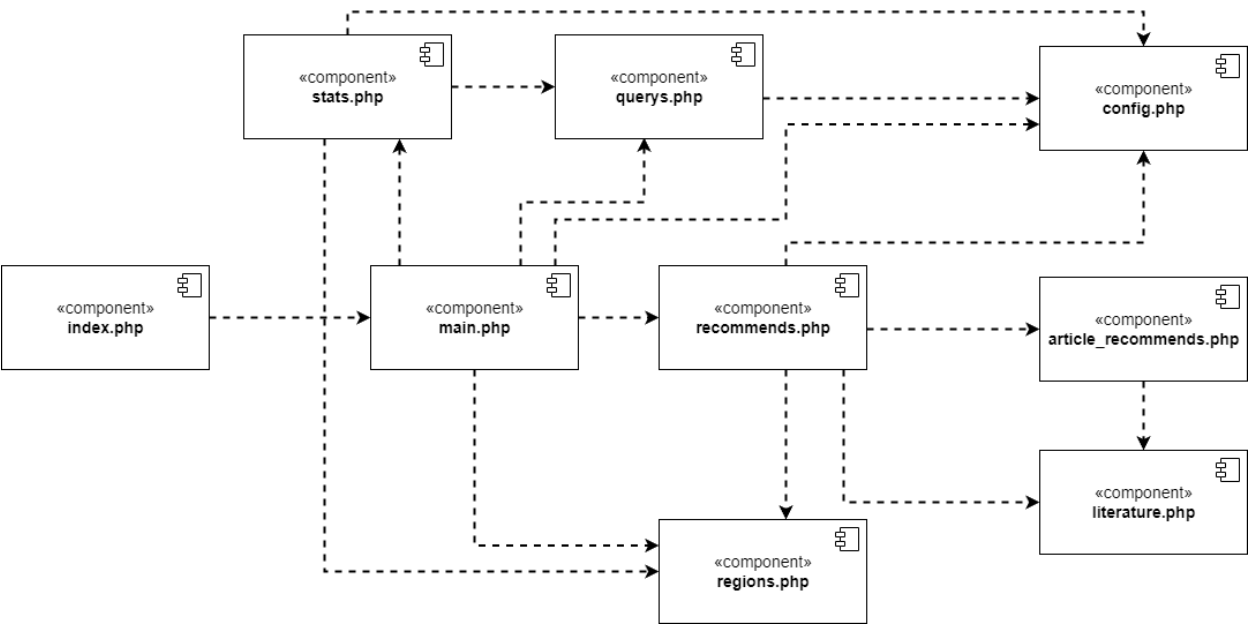


Рисунок 2 — Диаграмма компонентов веб-приложения

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, в которых отсутствуют связи между собой. Реляционная модель базы данных разработана с помощью стороннего сервиса app.diagrams. Модель содержит три независимых отношения и представлена на рисунке 2.

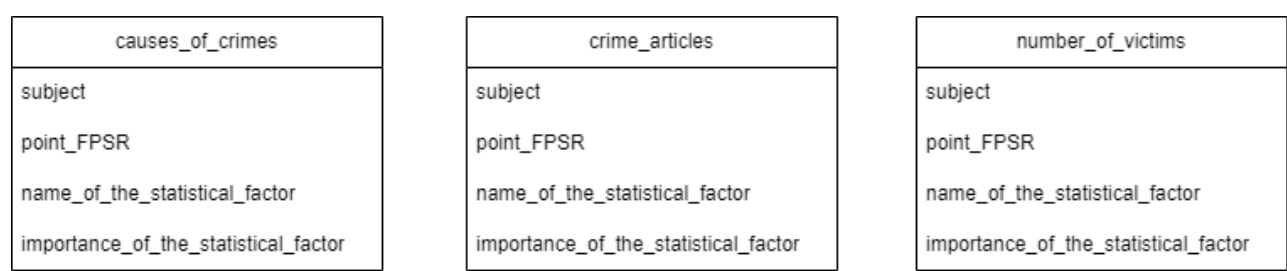


Рисунок 3 – Реляционная модель базы данных

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 [6]. Основное предназначение библиотеки пришлось на использование адаптивной сетки для приложения. Приложение имеет динамический задний фон, написанный на языке JavaScript с помощью встроенного класса Canvas [7]. 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 сверстанных 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 result – результирующий массив, key – название статистического фактора, value – значение статистического фактора (см. приложение А листинг – 1).

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

Функция `count_general_statistics_percent` принимает название региона параметром, высчитывает процентное соотношение атрибутов со всех датасетов, содержащее 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 – общая сумма, n – общее 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) и возвращает массив данных формата, согласно формуле (1) (см. приложение А листинг – 4).

Функция `count_standard_deviation` 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 4.

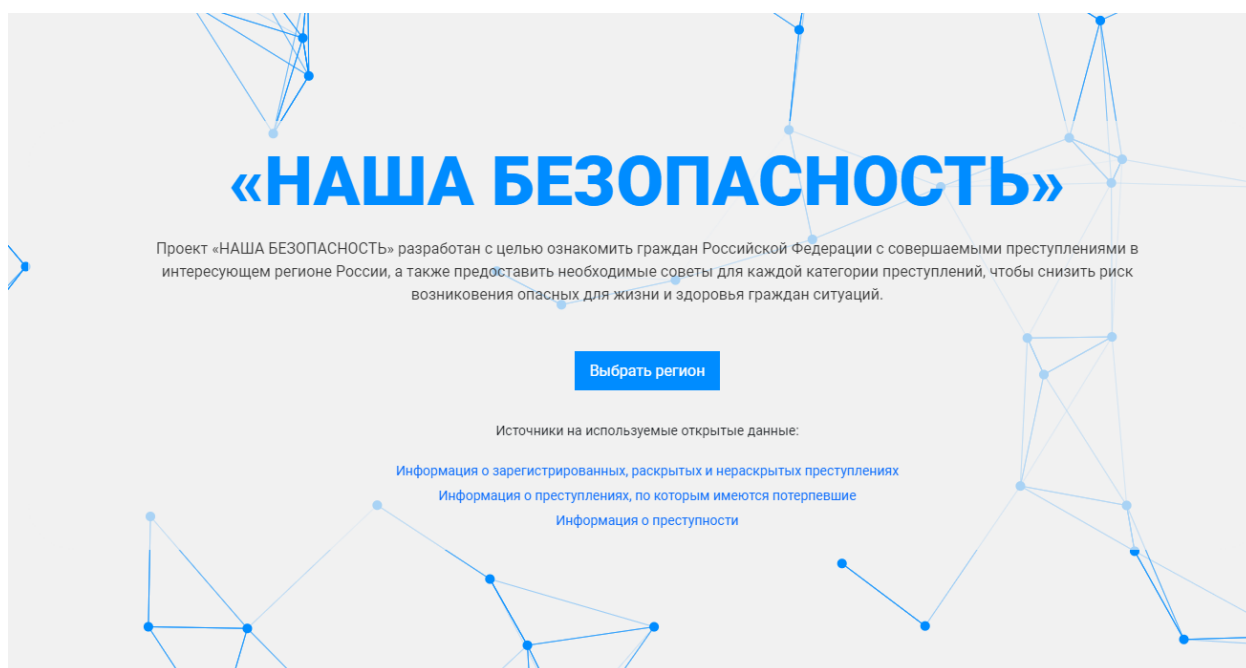


Рисунок 4 – Интерфейс титульной станицы веб-приложения

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 5.

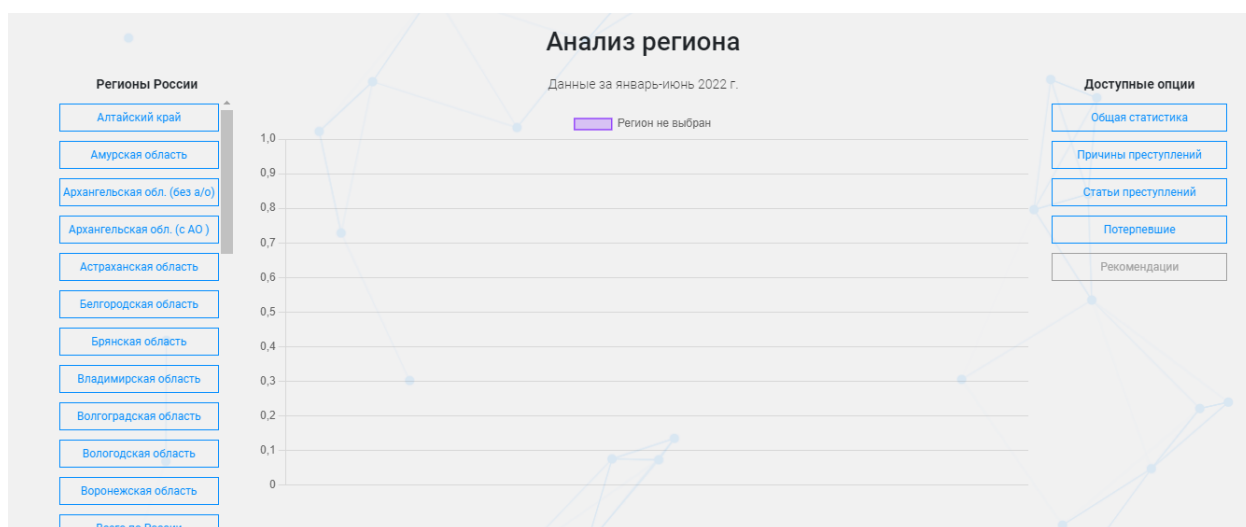


Рисунок 5 – Интерфейс главной страницы веб-приложения с невыбранными параметрами

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 6 и рисунке 7.

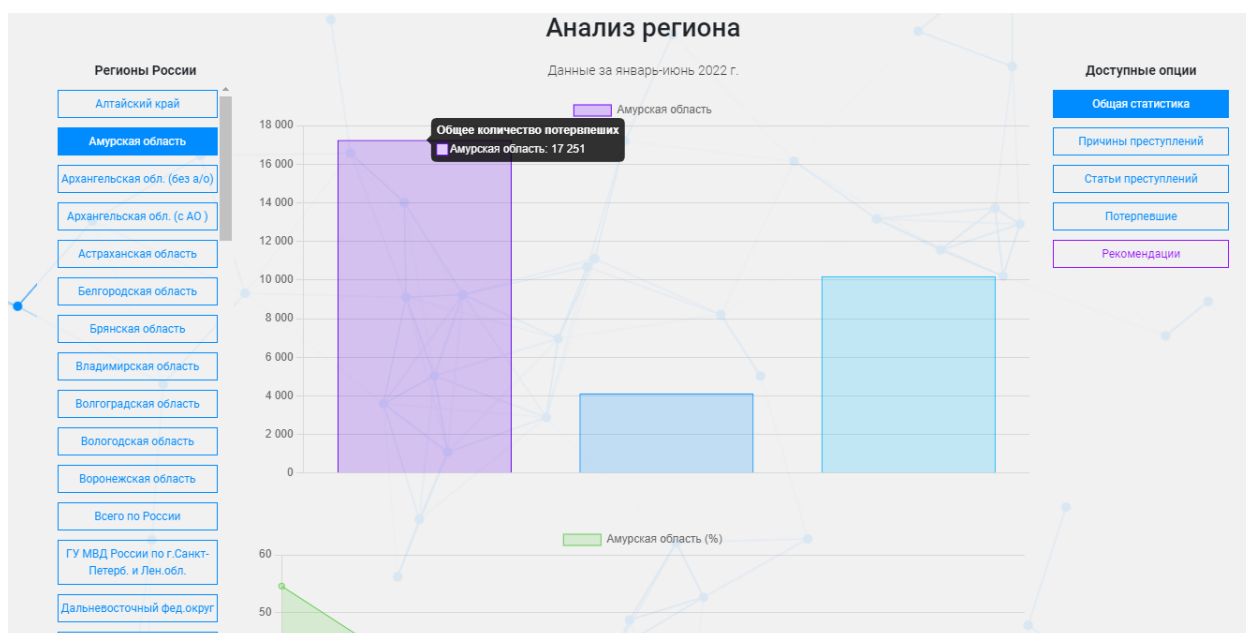


Рисунок 6 – Интерфейс главной страницы веб-приложения с выбранными параметрами

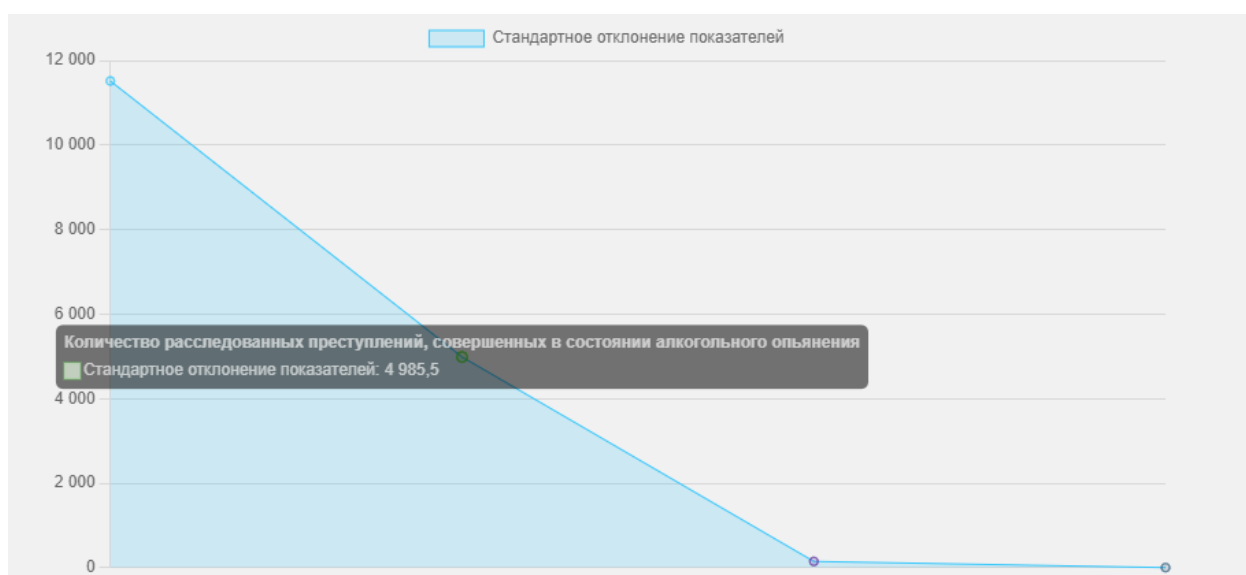


Рисунок 7 – График стандартного отклонения показателей

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 8.

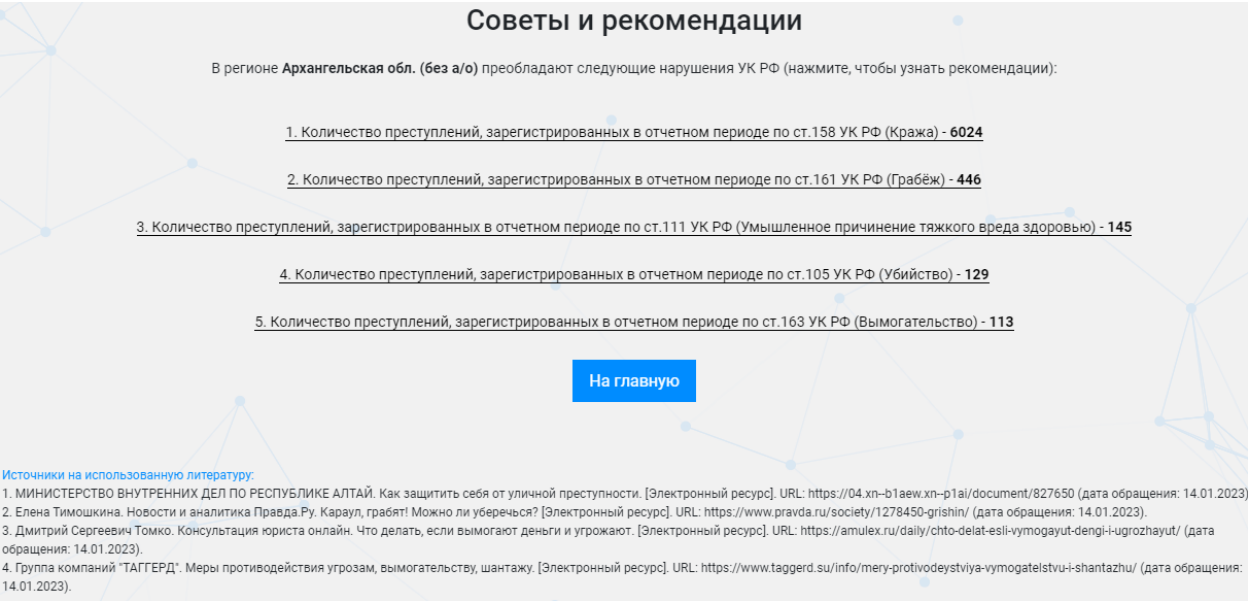


Рисунок 8 – Интерфейс страницы с наиболее часто нарушаемыми статьями

После выбора интересующей статьи, пользователь попадает на страницу с рекомендациями и советами по данному уголовному кодексу. Интерфейс страницы с рекомендациями представлен на рисунке 9.

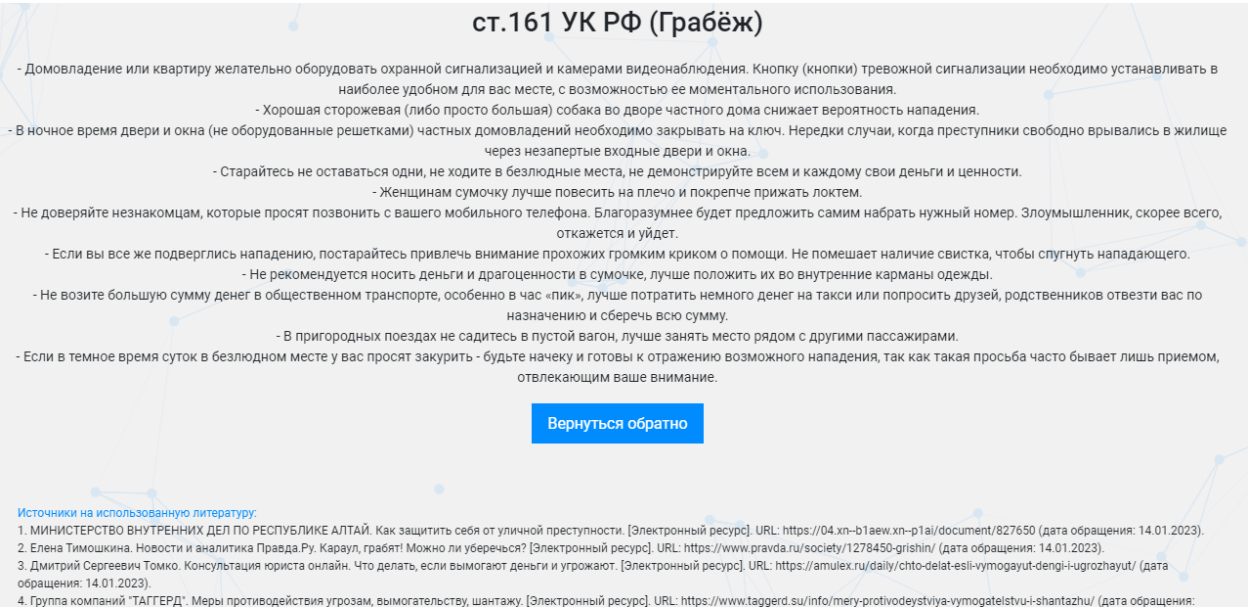


Рисунок 9 – Интерфейс страницы с рекомендациями по выбранной статье

Заключение

Результатом работы является динамическое веб-приложение, помогающее анализировать статистику преступности в регионах России в целях повышения информированности пользователей. Проект внес свой вклад в борьбу с преступностью в регионах страны. В будущем приложение будет дорабатываться, расширяться и улучшаться путем внедрения большего числа гистограмм с результатами статистической обработки 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. App diagrams net. [Электронный ресурс].
URL: <https://app.diagrams.net/> (дата обращения: 10.01.2023).
6. Введение. [Электронный ресурс]. URL: <https://bootstrap5.ru/docs/getting-started/introduction> (дата обращения 10.01.2023).
7. 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 {
```

Разрыв листинга ...

... продолжение листинга

```
$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"];
        }
    }
    Разрыв листинга ...
}
```

... продолжение листинга

```
    }
    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"];
            }
        }
    }
}
```

Разрыв листинга...

... продолжение листинга

```
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_standard_deviation($query) {
    $result = array();

    global $connect;

    global $regions;

    // -- Return arrow to start of query string result
    mysqli_data_seek($query, 0);
    $total_sum = 0;

```

Разрыв листинга...

... продолжение листинга

```
$total_count = 0;

$total_sum_of_the_statistical_factors = array();
$average_of_the_statistical_factors = array();

$standard_deviations = array();

// -- Count total sum 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 standard deviation 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);

```

Разрыв листинга...

... продолжение листинга

```
$result = sqrt($numerators[$key] / (count($regions) - 1));  
          $standard_deviations[$key] =  
round($result, 2);  
          } else {  
            $numerators[$key] =  
pow($row["importance_of_the_statistical_factor"] - $value, 2);  
          }  
        }  
      }  
    }  
  }  
  return $standard_deviations;  
}
```