УДК 004.896

Разработка функционального поиска для веб-системы

Акентьев Данила Денисович
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В работе рассматривается разработка функционального поиска для веб-системы на основе фреймворка Yii2. Описаны этапы создания и интеграции поисковых строк, методы обработки запросов, защита от SQL-инъекций, а также визуальные и функциональные аспекты конечного продукта.
Ключевые слова: веб-система, функциональный поиск, Yii2, SQL-инъекции, Materialize.

Development of a functional search for a web system

Akentev Danila Denisovich
Sholom-Aleichem Priamursky State University
Student

Abstract
The paper considers the development of a functional search for a web system based on the Yii2 framework. The stages of creating and integrating search strings, query processing methods, protection against SQL injections, as well as visual and functional aspects of the final product are described.
Keywords: web system, functional search, Yii2, SQL injection, Materialize.

1 Введение

1.1 Актуальность

В условиях стремительного развития информационных технологий и увеличения объема данных важным аспектом становится обеспечение эффективного поиска информации в веб-системах. Создание надежного и быстрого механизма поиска является ключевым фактором в улучшении пользовательского опыта и повышении функциональности веб-приложений.

1.2 Обзор исследований

М.Е. Кочитов рассмотрел в своей статье применение пагинации на содержимое вебсайтов для разбивки его на страницы, используя язык разметки HTML, язык программирования PHP и готовый шаблон стилей CSS от Materialize [1]. Про защиту базы данных от sql-инъекций рассказали и показали В.В. Новошинский и А.В. Сапожникова [2]. Рассмотрели все преимущества использования фреймворка yii2, а также проанализировали статистику популярности в своей работе И.И. Тимергалиев, Т.А. Бикулов,

А.Д. Давлетшин, И.Г. Сазгетдинов [3]. Показал и разработал собственный пример про использование плагина activeform для отображения на веб странице интерактивных форм в php фреймворке yii2, автор статьи М.Е. Кочитов [4]. М.Е. Кочитов показал пример использования плагина activerecord и query для управления базой данных в php фреймворке yii2 [5].

1.3 Цель исследования

Целью исследования является разработка и внедрение функционального поиска для веб-системы на базе фреймворка Yii2, обеспечивающего быструю и надежную обработку пользовательских запросов с минимизацией рисков, связанных с безопасностью данных.

2 Материалы и методы

Для достижения поставленных целей использовались следующие методы и инструменты:

- Фреймворк Yii2 для разработки веб-системы.
- PHP для написания серверной логики.
- HTML и CSS, с использованием Materialize для создания пользовательского интерфейса.
- Методы защиты от SQL-инъекций для обеспечения безопасности.
- Тестирование и отладка кода на различных этапах разработки.

3 Результаты и обсуждения

Первоочередной задачей было создать представление для отображения поисковой системы в разделе views/site/search.php и подключение виджетов и помощников фреймворка yii2 (рис.1).

```
use yii\helpers\Html;
use yii\widgets\ActiveForm;
use yii\widgets\LinkPager;
use yii\widgets\Modal;
```

Рисунок 1 – Подключение виджетов и помощников

Далее был добавлен код отображения поисковых строк, а также использование метода get-запроса для передачи значения переменных в контроллер страницы (рис.2).

```
<div class="site-about">
<h1><?= Html::encode($this->title) ?></h1>
    <br>
    <?php $form = ActiveForm::begin(['method' => 'get']); ?>
        <label for="inputName" class="form-label">Название диплома</label>
        <?= Html::textInput('namediplom', $this->params['namediplom'] == '' ? '' : $this->params['namediplom'], [
                                                                                'placeholder' => 'Введите название диплома',
                                                                                'size' => 25,
                                                                                'class' => "form-control",
                                                                                'type' => "text"
            ])
        ?>
        <label for="inputAuthor" class="form-label">Имя автора</label>
        <?= Html::textInput('studentdiplom', $this->params['studentdiplom'] == '' ? '' : $this->params['studentdiplom'], [
                                                                                    'placeholder' => 'Введите автора диплома',
                                                                                    'size' => 25,
                                                                                    'class' => "form-control",
                                                                                    'type' => "text"
            ])
        ?>
        <label for="inputDate" class="form-label">Дата</label>
        <?= Html::textInput('datediplom', $this->params['datediplom'] == '' ? '' : $this->params['datediplom'], [
                                                                                'placeholder' => 'Введите дату диплома',
                                                                                'size' => 25,
                                                                                'class' => "form-control",
                                                                                'type' => "date"
            ])
        ?>
        <br>
        <?= Html::submitButton('Отправить', ['class' => 'btn btn-outline-danger']) ?>
    <?php ActiveForm::end(); ?>
</div>
```

Рисунок 2 – Код поисковых строк с использованием ActiveForm

Для обработки полученных переменных требуется добавить функцию в контроллер сайта (рис.3). Он будет обрабатывать все действия, связанные с этим поиском такие как защита от SQL-инъекций, пагинация и сам поиск по базе данных.

```
public function actionSearch($namediplom = '', $studentdiplom = '', $datediplom = '')
{
    $data = $namediplom;
    $data1 = $studentdiplom;
    $data2 = $datediplom;

    ////////////////////////////////////////////////// Инъекции SQL
    $data = trim($data);
    $data1 = trim($data1);
    $data2 = trim($data2);

    $white = ["ALTER", "DATABASE", "BETWEEN", "CASE", "CHECK", "CREATE", "DATABASE", "INDEX", "REPLACE", "PROCEDURE", "UNIQUE", "VIEW", "DEFAULT",
            "DELETE", "DESC", "DISTINCT", "DROP", "COLUMN", "CONSTRAINT", "EXEC", "EXISTS", "FOREIGN", "FROM", "OUTER", "GROUP", "HAVING", "INNER",
            "INSERT", "JOIN", "LEFT", "LIKE", "LIMIT", "NOT", "NULL", "ORDER", "PRIMARY", "RIGHT", "ROW", "SELECT", "TABLE", "TRUNCATE", "UNION",
            "UPDATE", "VALUES", "WHERE"];
    foreach($white as $key)
    {
        $b1 = [strtoupper($data), strtoupper($data1), strtoupper($data2)];
        if(str_contains($b1[0], $key) || str_contains($b1[1], $key) || str_contains($b1[2], $key)) $this->redirect('?r=site/search');
    }

    static $pattern = "/[!@#$&*.,?-_}{\/+\=\(\)\"\:\;\`\"\\\><|]/";
    if(preg_match($pattern, $data, $matches, PREG_OFFSET_CAPTURE) || preg_match($pattern, $data1, $matches, PREG_OFFSET_CAPTURE)) $this->redirect('?r=site/search');
    //////////////////////////////////////////////////

    $query = Works::find()->joinWith('student');
    if(!empty($data)) $query->andWhere(['LIKE', 'name', '%'.$data.'%', false]);
    if(!empty($data1)) $query->andWhere(['LIKE', 'student.surname', '%'.$data1.'%', false]);
    if(!empty($data2)) $query->andWhere(['LIKE', 'datez', '%'.$data2.'%', false]);

    $pagination = new Pagination(['PageSize' => 10,'totalCount'=> $query->count()]);

    $models = $query->offset($pagination->offset)->limit($pagination->limit)->all();

    \Yii::$app->getView()->params['namediplom'] = $data;
    \Yii::$app->getView()->params['studentdiplom'] = $data1;
    \Yii::$app->getView()->params['datediplom'] = $data2;

    return $this->render('search',['pages' => $pagination,'models'=> $models]);
}
```

Рисунок 3 – Код контроллера поисковой системы

После чего был добавлен код вывода результатов запроса в стиле и оформлении под систему использованием Materialize (рис.4). Использования заданных стилей обусловлена большим количеством символов, хранящихся в этих переменных. Благодаря данным стилям визуальная составляющая этой системы будет улучшена (рис.5).

```
<div class="row">
    <?php foreach($models as $works): ?>
        <div class="col m6">
            <div class="card green lighten-3">
                <div class="card-content black-text" >
                    <span style="overflow: hidden; text-overflow: ellipsis; white-space: nowrap;" class="card-title">
                        <strong>№<?= $works->id_works; ?> <?= Html::encode($works->name); ?></strong>
                    </span>
                    <p style="overflow: hidden; text-overflow: ellipsis; white-space: nowrap;"><?= Html::encode($works->typew); ?> по направлению: <?= Html::encode($works->specialty->name);?></p>
                    <p style="overflow: hidden; text-overflow: ellipsis; white-space: nowrap;">Студент: <?= Html::encode($works->student->surname); ?></p>
                    <p style="overflow: hidden; text-overflow: ellipsis; white-space: nowrap;">Руководитель: <?= Html::encode($works->sotrudnik->surname); ?></p>
                    <p style="overflow: hidden; text-overflow: ellipsis; white-space: nowrap;">Оценка: <?= Html::encode($works->ozenka); ?></p>
                    <p style="overflow: hidden; text-overflow: ellipsis; white-space: nowrap;">Дата: <?= Html::encode($works->datez); ?></p>
                </div>
                <div class="card-action">
                    <?= Html::a('Подробнее', ['site/workdetail', 'id_works' => $works->id_works,], ['class' => 'waves-effect waves-light btn-small']) ?>
                </div>
            </div>
        </div>
    <?php endforeach; ?>
</div>
<div class="pagination"> <? echo LinkPager::widget(['pagination' => $pages]); ?> </div>
```

Рисунок 4 – Код вывода результатов поиска

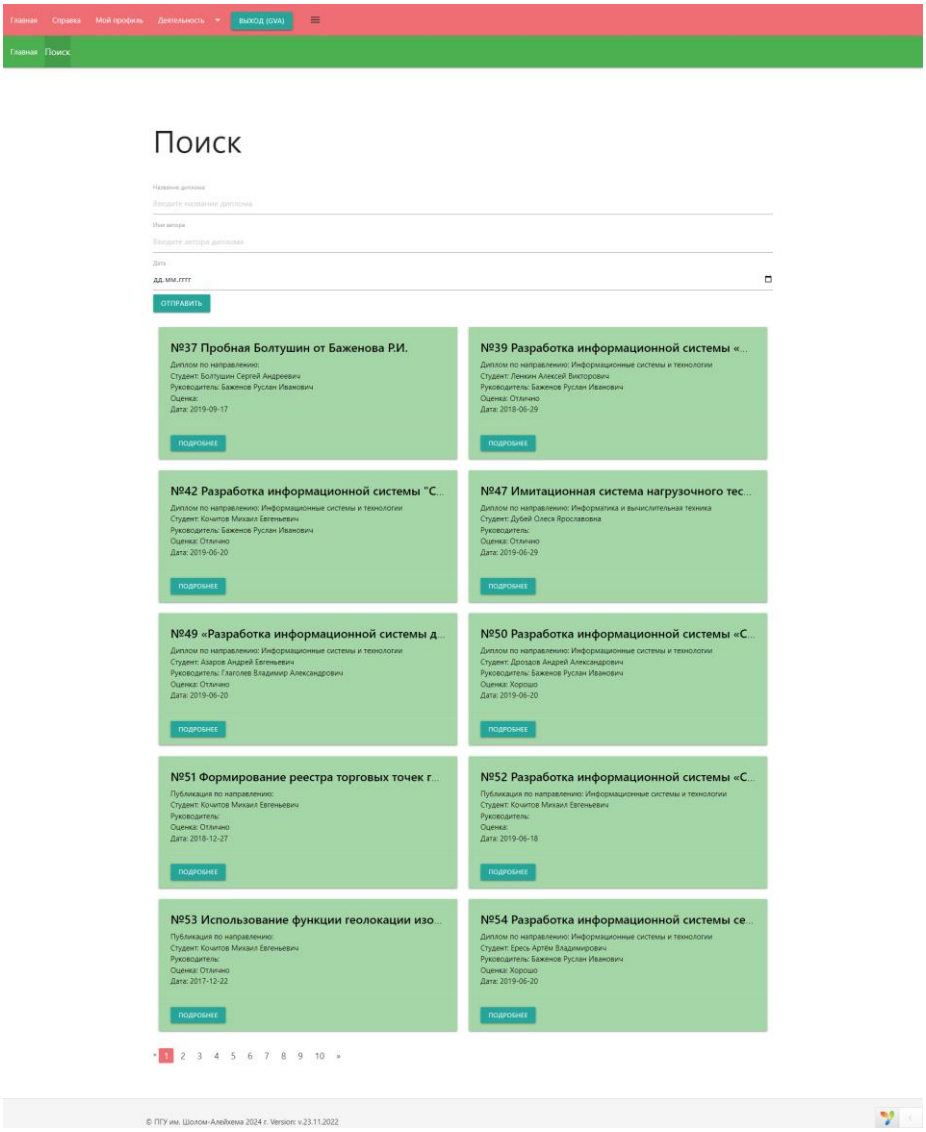

Рисунок 5 – Визуальный вид
поисковой системы

Так же добавлено подробное описание публикации с помощью кнопки «Подробнее», которая позволяет ознакомится с аннотацией диплома, а также позволяет скачать диплом при авторизации на сайте (рис.6). Но для скачивания диплома есть 2 условия:

1. Пользователь должен быть авторизован под любой учетной записью (студент, преподаватель, администратор).
2. Если файл диплома присутствует на сервере.

```
<div class="card-action">
    <?= Html::a('Подробнее', ['site/workdetail', 'id_works' => $works->id_works,], ['class' => 'waves-effect waves-light btn-small']) ?>
</div>
```

Рисунок 6 – Код кнопки «Подробнее»

После чего был реализован вывод всех основных данных о работе (рис.7). Внешний вид страницы был сделан под стили системы и организован через Materialize.

```
<div class="work-detail">
    <h1><?= Html::encode($this->title) ?></h1>
    <div class="work-detail card green lighten-3">
        <div class="card-content black-text">
            <p><strong>Название:</strong> <?= Html::encode($work->name) ?></p>
            <p><strong>Тип работы:</strong> <?= Html::encode($work->typew) ?></p>
            <p><strong>Направление:</strong> <?= Html::encode($work->specialty->name) ?></p>
            <p><strong>Студент:</strong> <?= Html::encode($work->student->surname /* . ' ' . $work->student->firstname */) ?></p>
            <p><strong>Руководитель:</strong> <?= Html::encode($work->sotrudnik->surname) ?></p>
            <p><strong>Оценка:</strong> <?= Html::encode($work->ozenka) ?></p>
            <p><strong>Дата:</strong> <?= Html::encode($work->datez) ?></p>
            <br>
            <?
            if (Yii::$app->user->isGuest) {
                // Если пользователь не авторизован, выводим неактивную кнопку
                echo Html::a(
                    '<i class="material-icons left">file_download</i> Скачать ' . $work->typew,
                    '#',
                    [
                        'class' => 'btn waves-effect waves-light tooltipped disabled',
                        'disabled' => true,
                    ]
                );
            } else {
                if ($work->filework !== null) {
                    echo Html::a(
                        '<i class="material-icons left">file_download</i> Скачать ' . $work->typew,
                        ['works/download', 'id' => $work->id_works],
                        [
                            'class' => 'btn waves-effect waves-light tooltipped',
                            'target' => '_blank',
                        ]
                    );
                } else {
                    // Если файл диплома отсутствует, выводим неактивную кнопку
                    echo Html::a(
                        '<i class="material-icons left">file_download</i> Скачать ' . $work->typew,
                        '#',
                        [
                            'class' => 'btn waves-effect waves-light tooltipped disabled',
                            'disabled' => true,
                        ]
                    );
                }
            }
            ?>
        </div>
    </div>
```

Рисунок 7 – Код отображения основной информации о работе.

Прежде чем выводить аннотацию на страницу, нужно было обработать текст для улучшения качества читаемости. Сначала разбили текст на абзацы с помощью php кода (рис.8), чтобы обернуть каждый абзац в тег <p> и сделать отступ в начале строки (рис.9). Внешний вид страницы (рис.10).

```
// Разбиваем аннотацию на абзацы
$paragraphs = explode("\n", $work->annotaciy);
```

Рисунок 8 – Код разделения

абзацев

```
<div class="work-detail card green lighten-4">
    <div class="card-content black-text">
        <div class="annotation" style="text-align: justify;">
        <strong>Аннотация:</strong>
            <?php foreach ($paragraphs as $paragraph): ?>
                <!-- Каждый абзац оборачиваем в тег <p> и применяем стиль для отступа -->
                <p style="text-indent: 1.5em"><?= Html::encode($paragraph) ?></p>
            <?php endforeach; ?>
        </div>
    </div>
</div>
```

Рисунок 9 – Код вывода аннотации
с использованием стилей

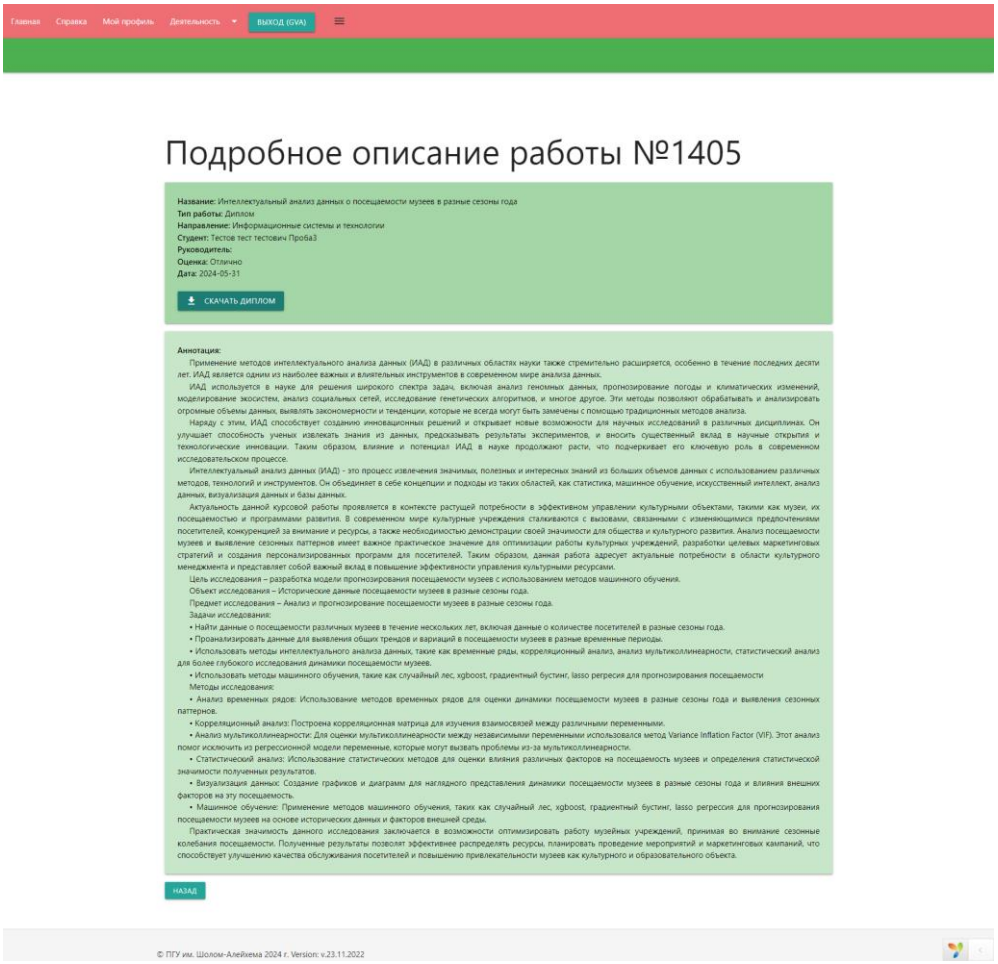

Рисунок 10 – Внешний вид страницы
подробной информации

4 Выводы

В результате проведенного исследования была разработана функциональная система поиска для веб-приложения на базе фреймворка Yii2. Проект включал в себя создание и интеграцию поисковых строк, методы обработки запросов и защиты от SQL-инъекций, а также визуальное оформление интерфейса с использованием фреймворка Materialize. Основные выводы работы включают:

1. Эффективность использования Yii2: Фреймворк Yii2 показал высокую производительность и удобство в реализации сложных поисковых функций. Его возможности позволяют быстро и эффективно создавать веб-системы с высокими требованиями к функциональности и безопасности.
2. Защита от SQL-инъекций: В процессе разработки был успешно реализован механизм защиты от SQL-инъекций, что обеспечило высокую степень безопасности данных. Это достигается за счет использования подготовленных запросов и других встроенных средств Yii2.
3. Интеграция и визуализация: Использование фреймворка Materialize для оформления интерфейса позволило создать современный и удобный для пользователя дизайн. Визуальные компоненты и стили Materialize значительно улучшили пользовательский опыт.
4. Практическая значимость: Разработанная система поиска может быть легко адаптирована и использована в различных веб-приложениях, требующих эффективного и быстрого поиска по базе данных. Это делает данный подход полезным для широкого круга разработчиков и проектов.

Библиографический список

1. Кочитов М.Е. Применение пагинации на содержимое вебсайтов // Постулат. 2018. Т.7. №. 33. С.27.
2. Новошинский В.В., Сапожникова А.В. Защита баз данных от sql-инъекций // Современные проблемы радиоэлектрики и телекоммуникаций. 2018. №.1. С. 234.
3. Тимергалиев И.И., Бикулов Т.А., Давлетшин А.Д. Преимущество использования фреймворка yii2 для разработки веб приложений // Вести научных достижений. 2020. №. 7. С. 10-13
4. Кочитов М.Е. Использование плагина activeform для отображения на веб странице интерактивных форм в php фреймворке yii2 // Постулат. 2020. Т. 1. №. 51. С. 70
5. Кочитов М.Е. Использование плагинов activerecord и query для управления базами данных в php фреймворке yii2 // Постулат. 2020. Т. 1. №. 51. С. 40.