Руководство программиста для веб-приложение “Планировщик задач”

1Введение

Это приложение для создания заметок, с возможностью удаления, редактирования и фильтрации.

Серверная часть написано с использланием PHP, и СУБД с языком MySQL, фронтенд с использованием JavaScript jQuery для отправки AJAX-запросов. Данные хранятся в базе данных. Приложение также использует Bootstrap для стилизации.

Основные функции:

Регистрация и авторизация пользователя;

Добавление, удаление, редактирование задачи;

Асинхронная обработка запросов через AJAX;

Фильтрация задач на стороне пользователя;

Поиск задач по ключевым словам из описания задачи.

Технологический стек:

Фронтенд: HTML, CSS, JavaScript, jQuery, Bootstrap.

- Бэкенд: PHP 8+, MySQL.

- Запросы: AJAX для взаимодействия клиента с сервером.

2 Архитектура приложения:

2.1 Общая схема архитектуры:

Клиент отправляет AJAX-запросы из браузера на серверные скрипты PHP для обработки данных, таких как аутентификация, работа с базой данных и управление контентом. Ответы возвращаются в формате JSON для обработки на стороне клиента.

2.2 Взаимодействие компонентов

1. Пользователь взаимодействует с веб-интерфейсом, отправляя запросы (например, добавление заметки или фильтрация данных).

2. Запросы отправляются с помощью jQuery AJAX на серверные PHP-файлы.

3. PHP обрабатывает запросы и взаимодействует с базой данных MySQL.

4. Ответ возвращается в формате JSON или HTML и обрабатывается JavaScript для обновления интерфейса.

2.3 Основные модули

-signin\_db.php, sigup\_db.php: обработка регистрации и входа пользователей.

- test.php: создание, редактирование и удаление задач.

- connect.php: модуль для подключения к базе данных MySQL.

3 Разработка фронтенда:

3.1 Установка окружения:

Необходимо запустить локальный сервер и подключить библиотеки.

 <!-- bootstrap -->

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet"

        integrity="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+ALEwIH" crossorigin="anonymous">

 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"

        integrity="sha384-YvpcrYf0tY3lHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcIdslK1eN7N6jIeHz"

        crossorigin="anonymous"></script>

    <!-- jquery -->

    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>

3.2 Структура проекта

/css

style.css – стили приложения

/db

connect.php – подключение к базе данных

signin\_db.php- обработка авторизации

signup\_db.php – обработка регистрации

viewall.php – вывод всех задач

/

test.php – главная рабочая страница

3.3 Взаимодейтсвие с сервером через AJAX

Пример отправки AJAX-запроса при доавление нвоой задачи

 function createTask() {

        $('#createForm').on('submit', function (event) {

            event.preventDefault();

            $.ajax({

                url: '/db/newNote.php',

                type: 'POST',

                data: $(this).serialize(),

                success: function (response) {

                    console.log(response);

                    if (response.status == 'ok') {

                        // Закрываем модальное окно только после успешного ответа

                        $('#staticBackdrop').modal('hide');

                        $('#staticBackdrop input[type="text"]').val('');

                        getAllTaskByUserId();

                    }

                },

                error: function (xhr, status, error) {

                    console.error('Произошла ошибка: ' + status);

                }

            });

        });

    }

3.3 Стилизация

Для стилизации используется CSS, Bootsrap

4 Разработка бэкенда

4.1 Подключение к базе данных

<?php

$con = mysqli\_connect('localhost', 'root', '', 'todoList');

?>

4.2 Обработка данных на сервере

Пример обработки добавления новой заметки

<?php

header('Content-Type: application/json');

require\_once "connect.php";

$title = isset($\_POST['title']) ? $\_POST['title'] : false;

$descr = isset($\_POST['descr']) ? $\_POST['descr'] : false;

$id\_user = isset($\_POST['id\_user']) ?intval($\_POST['id\_user']) : false;

if ($title and $descr) {

    $sql = mysqli\_query($con, "INSERT INTO `tasks`(`id\_user`, `title`, `description`, `is\_complited`) VALUES ($id\_user,'$title','$descr',0)");

    if ($sql) {

        $id\_task = mysqli\_insert\_id($con); // Get the last inserted ID

        $query = mysqli\_fetch\_assoc(mysqli\_query($con, "SELECT \* FROM `tasks` WHERE id\_tasks = $id\_task")); // Fetch a single row

        $response = [

            'id\_task'=> $query['id\_tasks'],

            'title'=> $query['title'],

            'description'=> $query['description'],

            'is\_complited'=> $query['is\_complited'],

            'created\_at'=> $query['created\_at'],

            'updated\_at'=> $query['updated\_at'],

            'status' => 'ok'

        ];

    } else {

        $response = [

            'status' => 'false',

            'error' => mysqli\_error($con) // Include error message

        ];

    }

    $json = json\_encode($response);

    echo $json;

}else {

    $response = [

        'status' => 'false'

    ];

    $json = json\_encode($response);

    echo $json;

}

?>

4.3 Авторизация пользователя

<?php

require\_once "connect.php";

session\_start();

$login = isset($\_POST['login'])?$\_POST['login']:false;

$pass = isset($\_POST['pass'])?$\_POST['pass']:false;

if($login and $pass){

    $check = mysqli\_query($con, "SELECT \* FROM `user`");

    if(mysqli\_num\_rows($check)!=0){

        $sql = mysqli\_query($con, "SELECT \* FROM `user` WHERE `login` = '$login'");

        if($sql){

            $user = mysqli\_fetch\_assoc($sql);

            if($login = $user['login']){

                $\_SESSION['id\_user'] = $user['id\_user'];

                header("Location: ../test.php");

            }

        }

    }

}

?>

4.4 Структура базы данных

CREATE TABLE `tasks` (

  `id\_tasks` int NOT NULL,

  `id\_user` int NOT NULL,

  `title` varchar(200) NOT NULL,

  `description` text NOT NULL,

  `is\_complited` tinyint(1) NOT NULL DEFAULT '0',

  `created\_at` timestamp NOT NULL DEFAULT CURRENT\_TIMESTAMP,

  `updated\_at` timestamp NOT NULL DEFAULT CURRENT\_TIMESTAMP

)

CREATE TABLE `user` (

  `id\_user` int NOT NULL,

  `login` varchar(50) NOT NULL,

  `pass` varchar(50) NOT NULL

)

5 Тестирование

5. Тестирование

5.1 Тестирование на клиенте

1. Проверьте корректность работы форм и их отправку через AJAX.

2. Убедитесь, что сервер корректно возвращает JSON-ответы, которые обрабатываются на клиенте.

5.2 Тестирование на сервере

1. Проверьте работу скриптов PHP с помощью тестовых данных.

2. Убедитесь, что обработка ошибок на сервере работает корректно (например, при отсутствии обязательных полей).

6. Развертывание

6.1 Настройка сервера

1. Убедитесь, что сервер поддерживает PHP 8+ и MySQL.

6.2 Развертывание на хостинге

1. Скопируйте файлы проекта на сервер.

2. Настройте файл конфигурации базы данных (`connect.php`) с корректными данными подключения.

3. Импортируйте структуру базы данных (SQL-файлы) на сервер.