Пример небольшой части руководства программиста для простого веб-приложения, которое представляет собой планировщик задач.

Руководство программиста для веб-приложения "Todo List"

1. Введение

Todo List— это веб-приложение для создания и ведения личных заметок. Пользователи могут регистрироваться, создавать заметки, редактировать их, а так управлять их статусом.

Основные функции:

* регистрация и аутентификация пользователей;
* создание, редактирование и удаление заметок;
* фильтрация и сортировка заметок;
* изменение статуса;
* поиск заметок.

Технологический стек:

* Фронтенд: HTML, CSS,JavaScript,jQurey, Bootstrap;
* Бэкенд: PHP, MySQL;
* Запросы: AJAX;

2. Архитектура приложения

2.1 Общая схема

Клиент отправляет AJAX-запросы из браузера к серверным PHP-скриптам для выполнения задач, таких как аутентификация, взаимодействие с базой данных и управление контентом. Ответы возвращаются в формате JSON, чтобы клиент мог обработать их на своей стороне.

2.2 Взаимодействие компонентов

* Пользователь взаимодействует с интерфейсом через браузер, отправляя запросы.
* Серверная часть обрабатывает запросы с помощью AJAX и взаимодействует с PHP для получения или записи данных.
* PHP обрабатывает запросы и взаимодействует с MySQL.
* Результаты операций возвращаются в виде JSON-ответов или HTML.

2.3 Основные модели данных

signin\_db.php, signup\_db.php обработка регистрации и входа пользователей.

newtask\_bd.php, update-status.php, edit\_db.php, delete\_db.php создание, редактирование, удаление заметки.

connect.php соединение с базой данных

3. Разработка фронтенда

3.1 Установка окружения

Необходимо запустить локальный сервер и подключить библиотеки (рисунок 1).



3.2 Структура проекта

```plaintext

/db

signin\_db.php связь с базой данных для регистрации

signup\_db.php связь с базой данных для авторизации

create-tsak.php компонент для добавления задач

delete.php компонент для удаления задач

edit.php компонент для редактирования задач

update.php компонент для изменения статуса задачи

//

connect.php связь с базой данных sql

header.php файл с шапкой сайта

index.php главная страница сайта

sign-out.php выход из личного кабинета

personal.php страница личного кабинета

sign-up.php страница регистрации

sign-in.php страница авторизации

/css

style.css стили страниц

/js

jquery.js библиотека jquery

```

3.3 Взаимодействие через AJAX

Пример отправки AJAX-запроса при смене статуса задачи (рисунок 2).

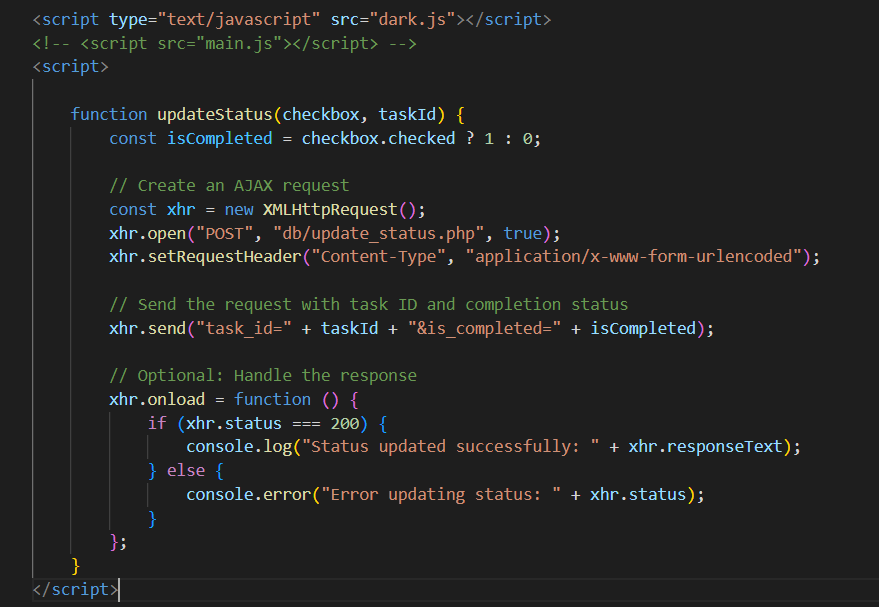


Рисунок 2 – Пример отправки AJAX-запроса.

3.4 Стилизация

Стилизация компонентов реализована с помощью Bootstrap и кастомных CSS-модулей. В проекте подключен Bootstrap, но для дополнительных правок используются SCSS:

```css

main.scss /

.search-bar {

    display: flex;

    justify-content: space-between;

    margin-bottom: 20px;

}

.search-container {

    display: flex;

    align-items: center;

    position: relative;

    width: 100%;

}

.search-container input {

    width: 100%;

    padding: 10px 40px 10px 15px;

    border-radius: 8px;

    border: 1px solid #ccc;

    font-size: 14px;

    outline: none;

}

```

4. Разработка бэкенда

4.1 Установка окружения

Подключение к базе данных в файле connect.php:

$con = mysqli\_connect("localhost","root","","todolist");

4.2 Маршруты API

Пример обработки добавления новой заметки (рисунок 3).

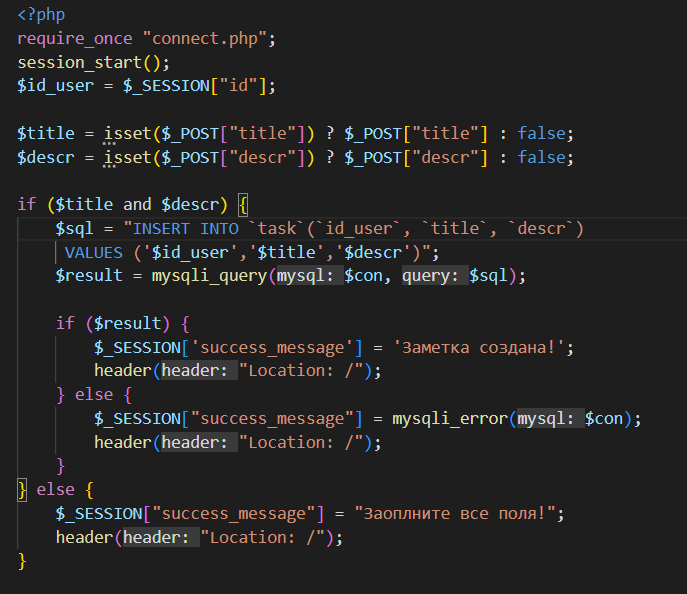


Рисунок 3 – Добавление заметки.

4.3 Авторизация пользователя

Авторизация пользователя (рисунок 4).



Рисунок 4 – Авторизация.

4.4 Модели данных

CREATE TABLE `task` (

  `id` int NOT NULL,

  `id\_user` int NOT NULL,

  `title` varchar(50) NOT NULL,

  `descr` text NOT NULL,

  `is\_completed` varchar(10) CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci NOT NULL DEFAULT '0',

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

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

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

--

-- Структура таблицы `users`

--

CREATE TABLE `users` (

  `id` int NOT NULL,

  `username` varchar(50) NOT NULL,

  `password\_hash` varchar(200) CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

5. Тестирование

5.1 Тестирование на клиенте

* Проверьте корректность работы форм и их отправку через AJAX;
* Убедитесь, что сервер корректно возвращает JSON-ответы, которые обрабатываются на клиенте.

5.2 Тестирование на сервере

* Проверьте работу скриптов PHP с помощью тестовых данных;
* Убедитесь, что обработка ошибок на сервере работает корректно (например, при отсутствии обязательных полей).

6. Развертывание

6.1 Настройка сервера

Убедитесь, что сервер поддерживает PHP 8+ и MySQL.

6.2 Развертывание на хостинге

* Скопируйте файлы проекта на сервер;
* Настройте файл конфигурации базы данных (`connect.php`) с корректными данными подключения;
* Импортируйте структуру базы данных (SQL-файлы) на сервер.