МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ «ПОЛОЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ ЕВФРОСИНИИ ПОЛОЦКОЙ»

Факультет информационных технологий

Кафедра технологий программирования

ЛАБОРАТОРНАЯ РАБОТА №9

по дисциплине «Веб - технологии»

Тема: «Создание To-Do List с использованием jQuery»

|  |  |
| --- | --- |
| Выполнил: | студент группы 23-ИТ-1  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_В.Ю. Страпко |
| Проверил: | ассистент кафедры ТП  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.А. Макарычева |

|  |
| --- |
|  |

**Цель работы:**научиться работать с событиями и локальным хранилищем для реализации динамического списка задач.

1. **Краткая теория**

Todo List – это список дел, которые вам нужно выполнить или того, что вы хотите сделать. Традиционно их пишут на листке бумаги и организовывают в порядке приоритета. При выполнении задачи, её обычно вычеркивают из списка. Но такой списосок можно вести не только на листке бумаги, но и в электронном виде, например, браузере.

1. **Практическая часть**

**Задание:** Реализуйте To-Do List с функционалом:

− Добавление новой задачи в список.

− Отметка задачи как выполненной с изменением стиля.

− Удаление задачи из списка.

− Сохранение списка задач в localStorage для восстановления при перезагрузке страницы.

Дополнительно: добавьте возможность сортировки задач по их статусу.

Код главного файла main.js – страницы представлен в листинге 2.1.

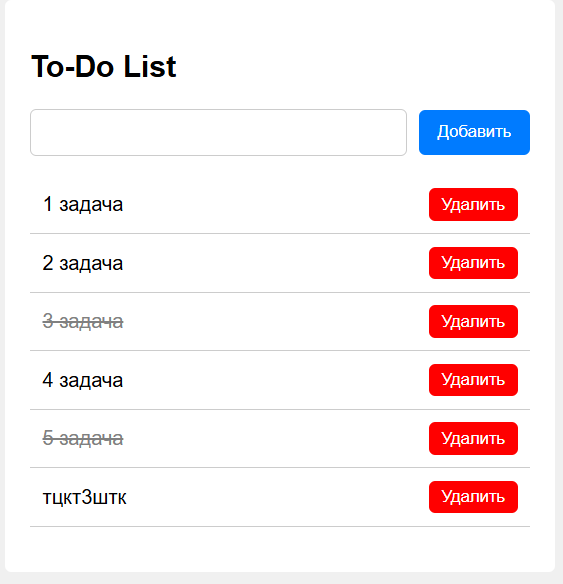
Листинг 2.1 – файл main.js.

1. $(document).ready(function() {
2. function loadTasks() {
3. const tasks = JSON.parse(localStorage.getItem('tasks')) || [];
4. tasks.forEach(task => {
5. addTaskToList(task.text, task.completed);
6. });
7. }
8. function addTaskToList(taskText, completed = false) {
9. const listItem = $('<li></li>');
10. const taskSpan = $('<span></span>').text(taskText);
11. if (completed) {
12. taskSpan.addClass('completed');
13. }
14. const deleteButton = $('<button>Удалить</button>');
15. deleteButton.click(function() {
16. $(this).parent().remove();
17. saveTasks();
18. });
19. listItem.append(taskSpan).append(deleteButton);
20. taskSpan.click(function() {
21. $(this).toggleClass('completed');
22. saveTasks();
23. });
24. $('#todoList').append(listItem);
25. }
26. function saveTasks() {
27. const tasks = [];
28. $('#todoList li').each(function() {
29. const taskText = $(this).find('span').text().trim();
30. const completed = $(this).find('span').hasClass('completed');
31. tasks.push({ text: taskText, completed });
32. });
33. localStorage.setItem('tasks', JSON.stringify(tasks));
34. }
35. loadTasks();
36. $('#addTask').click(function() {
37. const taskText = $('#taskInput').val().trim();
38. if (taskText) {
39. addTaskToList(taskText);
40. $('#taskInput').val('');
41. saveTasks();
42. }
43. });
44. $('#taskInput').keypress(function(e) {
45. if (e.which === 13) {
46. $('#addTask').click();
47. }
48. });
49. });

В данном листинге было реализованно следующее:

Этот код реализует простое приложение для управления задачами (To-Do List) с использованием jQuery. При загрузке страницы он извлекает сохраненные задачи из localStorage и отображает их в виде списка. Пользователь может добавлять новые задачи, отмечать их как выполненные, а также удалять их. Каждое действие (добавление, изменение состояния или удаление) обновляет локальное хранилище, чтобы сохранить текущее состояние задач даже при перезагрузке страницы. Ввод новых задач возможен как через кнопку, так и по нажатию клавиши Enter.

**3.Скриншоты результата**

****

**Рисунок 3.1 – To-do List**

**Вывод:** в ходе данной лабораторной работы были изучены принципы работы с событиями и локальным хранилищем для реализации динамического списка задач.