МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М. Є. Жуковського

«Харківський авіаційний інститут»

Факультет літакобудування

Кафедра 105 «Інформаційних технологій проектування»

**Звіт**

до лабораторної роботи № 5

з дисципліни

«WEB-програмування»

за темою

«Знайомство з Javascript та бібліотекою jQuery.  Побудова валідації форми та динамічних  таблиць.»

Виконав: студент 1 курсу групи № 115

напряму підготовки (спеціальності)

134 Авіаційна та ракетно-космічна техніка

Олександр ЛОСЄВ

Прийняв: викл. 105 Артур КЛОЧКО

Харків – 2025

## ХІД РОБОТИ

Метою цієї лабораторної роботи є формування базових навичок використання javascript  для створення елементів з динамічною поведінкою у браузері. Для досягнення цієї мети  також буде необхідно ознайомитись з бібліотекою jQuery.

**Створення форми авторизації**  
У рамках лабораторної роботи було створено форму авторизації для входу в панель адміністрування блогу. Форма була розміщена по центру сторінки за допомогою класу justify-content-center бібліотеки Bootstrap. Для візуального оформлення використано компонент card, який забезпечує рамку та заголовок форми. Код форми включає поля для введення email та пароля, а також кнопку для відправлення даних. Форма має наступну структуру:

Контейнер з класом container justify-content-center.

Компонент card з заголовком "Login".

Поля введення для email (type="text") та пароля (type="password") з класом form-control.

Кнопка відправлення форми з класом btn btn-primary.

**Підключення бібліотек та файлів**  
Для роботи форми було підключено CSS та JavaScript файли бібліотеки Bootstrap, а також бібліотеку jQuery. Підключення виконано через теги <link> для CSS та <script> для JavaScript файлів у секції <head> HTML-документа. Додатково створено файл main.js для обробки логіки форми, який підключено після бібліотек jQuery та Bootstrap.

**Валідація форми авторизації**  
У файлі main.js реалізовано JavaScript-код для перевірки введених даних форми. Використано методи jQuery:

$('#login-form').click() для обробки події натискання кнопки відправлення форми.

event.preventDefault() для запобігання стандартній поведінці браузера (відправлення даних на сервер до завершення перевірки).

$('#email-address').val() та $('#password').val() для отримання значень полів форми.

Перевірка на порожні поля: якщо поле email або пароль порожнє, додається клас is-invalid для виділення поля червоною рамкою, а змінна error встановлюється в true. Якщо помилок немає, клас is-invalid видаляється.

Якщо помилок немає (error = false), форма відправляється на сервер за допомогою методу $('#login-form').submit().

**Створення сторінки posts.html**  
Було створено окрему сторінку posts.html зі стандартною структурою Bootstrap. На сторінці розміщено таблицю з класами table та table-striped для відображення списку постів. Таблиця містить заголовки колонок (#, Title, Created At) та тіло з ідентифікатором posts для динамічного заповнення.

**Динамічне створення рядків таблиці**  
У файлі posts.js створено масив posts, який містить об’єкти з полями id, title та created\_date. Реалізовано функцію createPostsRows(posts), яка:

Очищає вміст таблиці за допомогою методу $('#posts').html('').

У циклі формує HTML-код рядків таблиці на основі даних масиву posts.

Додає згенерований HTML-код до таблиці за допомогою методу $('#posts').html(rowsHtml).  
Функція викликається в кінці файлу posts.js для початкового відображення таблиці.

**Сортування даних таблиці**  
Додано кнопку <button id="sort-button" class="btn btn-primary">Sort</button> для сортування таблиці за заголовком постів. У файлі posts.js реалізовано обробник події натискання кнопки за допомогою $('#sort-button').click(). Сортування виконано методом posts.sort(), який порівнює значення поля title двох об’єктів і повертає 1 або -1 для визначення порядку. Після сортування викликається функція createPostsRows(posts) для оновлення таблиці.

**Самостійна робота**

Для виконання самостійної роботи було доопрацьовано функцію сортування таблиці, щоб вона дозволяла сортувати дані за будь-якою колонкою (id, title, created\_date) у порядку зростання або зменшення. Для цього:

**Створено форму вибору параметрів сортування**:

Додано HTML-форму з двома випадаючими списками (<select>):

Перший список дозволяє обрати колонку для сортування (id, title, created\_date).

Другий список визначає порядок сортування (asc для зростання, desc для зменшення).

Додано кнопку <button id="sort-button" class="btn btn-primary">Sort</button> для запуску сортування.

**Модифіковано функцію сортування**:

У файлі posts.js реалізовано функцію, яка отримує значення полів форми за допомогою

$('#sort-button').click(function(){

  let field = $('#sort-field').val()

  let direction = $('#sort-direction').val()

  posts.sort(function(a, b) {

    let aValue = a[field]

    let bValue = b[field]

    if (field === 'created\_date') {

      aValue = new Date(aValue)

      bValue = new Date(bValue)

    }

    if (aValue > bValue) {

      return direction === 'asc' ? 1 : -1

    } else if (aValue < bValue) {

      return direction === 'asc' ? -1 : 1

    } else {

      return 0

    }

  })

  createPostsRows(posts)

})

Після сортування викликається createPostsRows(posts) для оновлення таблиці.

**Розроблена нова сторінка register.html, яка реалізує форму реєстрації за прикладом у завданні. Також для неї зроблен “register.js”**

Ця форма має 3 поля (email адрес, пароль та повтор паролю). Також для неї зроблена валідація яка перевіряє, що користувач ввів усі данні форми та поле пароль і повтор паролю збігаються.

<html >

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Реєстрація</title>

  <link rel="stylesheet" href="css/bootstrap.min.css">

</head>

<body>

  <div class="container justify-content-center mt-5">

    <div class="card">

      <div class="card-header">Register</div>

      <div class="card-body">

        <form id="register-form">

          <div class="form-group row">

            <label for="email" class="col-md-4 col-form-label text-md-right">E-Mail Address</label>

            <div class="col-md-6">

              <input type="text" id="email" class="form-control" name="email">

              <div class="invalid-feedback">Please enter email.</div>

            </div>

          </div>

          <div class="form-group row">

            <label for="password" class="col-md-4 col-form-label text-md-right">Password</label>

            <div class="col-md-6">

              <input type="password" id="password" class="form-control" name="password">

              <div class="invalid-feedback">Please enter password.</div>

            </div>

          </div>

          <div class="form-group row">

            <label for="confirm-password" class="col-md-4 col-form-label text-md-right">Confirm Password</label>

            <div class="col-md-6">

              <input type="password" id="confirm-password" class="form-control" name="confirm-password">

              <div class="invalid-feedback">Passwords do not match.</div>

            </div>

          </div>

          <div class="col-md-6 offset-md-4">

            <button type="submit" class="btn btn-primary" id="register-button">Register</button>

            <button type="submit" class="btn btn-link" id="Go to login">Go to login</button>

        </div>

        </form>

      </div>

    </div>

  </div>

   <script src="js/jquery-3.7.1.min.js"></script>

   <script src="js/bootstrap.min.js"></script>

   <script src="js/register.js"></script>

</body>

</html>

**“register.js”**

$('#register-button').click(function(event){

  event.preventDefault()

  let email = $('#email').val()

  let password = $('#password').val()

  let confirmPassword = $('#confirm-password').val()

  let error = false

  if (email === '') {

    $('#email').addClass('is-invalid')

    error = true

  } else {

    $('#email').removeClass('is-invalid')

  }

  if (password === '') {

    $('#password').addClass('is-invalid')

    error = true

  } else {

    $('#password').removeClass('is-invalid')

  }

  if (confirmPassword !== password || confirmPassword === '') {

    $('#confirm-password').addClass('is-invalid')

    error = true

  } else {

    $('#confirm-password').removeClass('is-invalid')

  }

  if (!error) {

    $('#register-form').submit()

  }

})

**Висновки**

У ході виконання лабораторної роботи було вивчено основи роботи з бібліотекою jQuery для створення інтерактивних веб-елементів. Реалізовано форму авторизації з валідацією даних, яка використовує методи jQuery для обробки подій та маніпуляції HTML-елементами. Створено сторінку реєстрації з валідацією та сторінкуз таблицею для відображення списку постів із можливістю динамічного формування рядків. Розроблено функцію сортування таблиці за заданою колонкою та порядком, що дозволило гнучко керувати відображенням даних. Використання бібліотеки Bootstrap забезпечило зручне та привабливе оформлення сторінки. Робота допомогла закріпити навички роботи з JavaScript, jQuery, Bootstrap та основами веб-розробки.

**Відповіді на контрольні питання**

1. **Для чого необхідна бібліотека jQuery?**  
   Бібліотека jQuery спрощує роботу з JavaScript, надаючи зручні методи для маніпуляції DOM, обробки подій, виконання AJAX-запитів та роботи з анімаціями. Вона забезпечує кросбраузерну сумісність та зменшує обсяг коду.
2. **Як за допомогою jQuery знайти необхідний HTML-елемент?**  
   Для пошуку HTML-елемента використовується селектор jQuery, наприклад, $('#id') для пошуку за ідентифікатором, $('.class') за класом або $('tag') за тегом. Наприклад: $('#email-address') знаходить елемент із id email-address.
3. **Як за допомогою jQuery додати чи видалити клас елементу?**  
   Для додавання класу використовується метод addClass(), наприклад: $('#email-address').addClass('is-invalid'). Для видалення — метод removeClass(), наприклад: $('#email-address').removeClass('is-invalid').
4. **Як за допомогою jQuery задати функцію, яка спрацьовує, коли користувач натискає на кнопку?**  
   Використовується метод click(), наприклад: $('#login-button').click(function() { alert('Button clicked!'); }). Функція, передана в click(), виконується при натисканні на елемент.
5. **Як підключити JavaScript-файл на сторінку?**  
   JavaScript-файл підключається за допомогою тегу <script> із атрибутом src, наприклад: <script src="js/main.js"></script>. Тег розміщується в секції <head> або в кінці <body>.
6. **Для чого потрібна команда** event.preventDefault()**?**  
   Команда event.preventDefault() запобігає стандартній поведінці браузера, наприклад, відправленню форми на сервер або переходу за посиланням, дозволяючи обробити подію вручну.
7. **Як зробити валідацію HTML-форми за допомогою JavaScript?**  
   Валідація форми виконується шляхом перевірки значень полів (val() у jQuery), додавання/видалення класів для візуального позначення помилок (addClass, removeClass) та використання умовних конструкцій для перевірки умов (наприклад, чи не порожнє поле). Якщо помилок немає, форма відправляється (submit()).
8. **Як за допомогою jQuery задати HTML-код елементу?**  
   Для встановлення HTML-коду використовується метод html(), наприклад: $('#posts').html('<tr><td>1</td><td>Title</td></tr>') замінює вміст елемента з id posts.
9. **Як за допомогою jQuery отримати значення поля форми?**  
   Значення поля форми отримується за допомогою методу val(), наприклад: $('#email-address').val() повертає текст, введений у поле з id email-address.
10. **Як працює функція** sort **для JavaScript-масиву?**  
    Функція sort() сортує елементи масиву, використовуючи функцію порівняння, яка повертає 1, -1 або 0 залежно від порядку елементів. Наприклад, array.sort((a, b) => a.title > b.title ? 1 : -1) сортує масив об’єктів за полем title за зростанням.