МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Институт математики и информационных систем
Факультет автоматики и вычислительной техники
Кафедра электронных вычислительных машин

Дата сдачи на проверку:

«__»_____ 2025 г.

Проверено:

«__»_____ 2025 г.

Разработка клиент–серверного веб-приложения на HTML, CSS, JavaScript и Express.js.

Отчёт по лабораторной работе №2

по дисциплине

«Web-программирование»

Разработал студент гр. ИВТб-2301-05-00 _____/Черкасов А. А./

(подпись)

Старший преподаватель _____/Преподаватель/

(подпись)

Киров

2025

Цели лабораторной работы

- изучить структуру клиент–серверных веб-приложений;
- освоить работу с Express.js и обработку POST-запросов;
- закрепить навыки раздельного использования HTML, CSS и JavaScript;
- научиться принимать и обрабатывать данные, отправленные из формы.

Задание

Создать веб-приложение, работающее по архитектуре «клиент–сервер», удовлетворяющее требованиям:

1. На стороне клиента реализовать HTML-страницу с формой отправки данных.
2. Стили разместить в отдельном файле CSS, логику обработки формы — в отдельном JS.
3. На стороне сервера реализовать обработку POST-запроса с использованием Express.js.
4. Сервер должен принимать данные, логировать их и возвращать ответ клиенту.
5. Организовать вывод всех полученных данных в виде массива.
6. Файлы клиента должны раздаваться через статическую директорию.

Ход выполнения работы

1. Создание структуры проекта

Проект был создан в соответствии с типовой структурой веб-приложения:

- `index.html` — основная страница с формой;
- `style.css` — стили оформления;
- `script.js` — логика отправки данных;
- `server.js` — сервер на Express.js;
- статические файлы размещены в папке `/public`.

2. Реализация клиентской части

Была разработана HTML-форма, содержащая несколько полей ввода и кнопку отправки. Все стили вынесены в отдельный файл CSS. JavaScript осуществляет отправку данных методом POST и выводит ответ сервера.

3. Реализация серверной части

Сервер был создан на Express.js. Были подключены модули:

- `express` — основной фреймворк;
- `body-parser` — обработка POST-запросов.

На сервере была реализована логика:

- раздача статических файлов;
- логирование всех входящих запросов (данные формы, query, headers);
- сохранение всех полученных данных в массив;
- ответ клиенту после успешного получения данных;
- вывод массива по запросу `/list`.

Также дополнительно был реализован расширенный лог всего, что происходит в сервере: тело запроса, параметры, заголовки, время получения.

4. Тестирование работы приложения

После запуска сервера через:

```
bun server.js
или
node server.js
```

Приложение было открыто в браузере по адресу:

`http://localhost:8080`

Поля формы были заполнены, данные успешно отправлены и отобразились в консоли, а затем были возвращены пользователю.

5. Скриншот формы

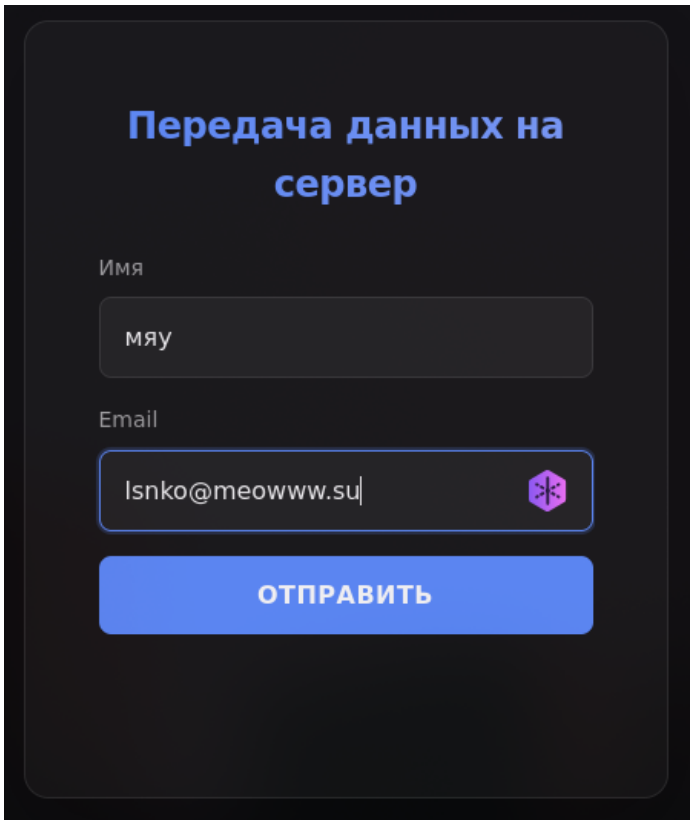

Рисунок 1 — Форма

Вывод

В ходе выполнения лабораторной работы было разработано полноценное клиент–серверное веб-приложение. Была изучена структура веб-проекта, реализован сервер на Express.js, выполнено разделение клиентского кода на HTML, CSS и JavaScript.

Были выполнены все требования задания: форма корректно отправляет данные, сервер принимает их, логирует, сохраняет и возвращает ответ клиенту.

Работа позволила закрепить знания по основам веб-разработки и принципам построения архитектуры «клиент–сервер».

Приложение А1. Исходный код index.html

```
<!doctype html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Лабораторная форма</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <div class="container">
    <h1 class="title">Передача данных на сервер</h1>

    <form id="form">
      <div class="form-group">
        <label class="form-label">Имя</label>
        <input class="form-control" name="name" required>
      </div>

      <div class="form-group">
        <label class="form-label">Email</label>
        <input class="form-control" name="email" type="email" required>
      </div>

      <button class="btn">Отправить</button>
    </form>

    <div id="response" class="response"></div>
  </div>

  <script src="script.js"></script>
</body>
</html>
```

Приложение А2. Исходный код script.js

```
const form = document.getElementById('form');
```

```
const responseBox = document.getElementById('response');
const submitBtn = form.querySelector('.btn');

const handleSubmit = async (e) => {
  e.preventDefault();

  const data = new FormData(form);
  showLoadingState();

  try {
    const res = await fetch('/', {
      method: 'POST',
      body: new URLSearchParams(data)
    });

    const text = await res.text();
    showResponse(`Сервер ответил: ${text}`);
    form.reset();

  } catch (error) {
    showResponse(`Ошибка: ${error.message}`);
  } finally {
    resetButtonState();
  }
};

const showLoadingState = () => {
  submitBtn.disabled = true;
  submitBtn.textContent = 'Отправка...';
};

const showResponse = (message) => {
  responseBox.textContent = message;
  responseBox.classList.add('show');

  setTimeout(() => {
    responseBox.classList.remove('show');
```

```
  }, 5000);
};

const resetButtonState = () => {
  submitBtn.disabled = false;
  submitBtn.textContent = 'Отправить';
};

form.addEventListener('submit', handleSubmit);
```

Приложение А3. Исходный код style.css

```
:root {
  /* Color palette */
  --bg-primary: #0a0a0a;
  --bg-secondary: #121212;
  --bg-tertiary: #1e1e1e;
  --accent-primary: #5a8bff;
  --accent-secondary: #3a6bef;
  --text-primary: #f5f5f5;
  --text-secondary: #a0a0a0;
  --border-color: rgba(255, 255, 255, 0.1);

  /* Typography */
  --font-main: 'Inter', -apple-system, BlinkMacSystemFont, 'Segoe UI', sans-serif;
  --font-size-base: 1rem;
  --font-size-lg: 1.25rem;
  --font-size-xl: 1.5rem;

  /* Spacing */
  --space-sm: 0.5rem;
  --space-md: 1rem;
  --space-lg: 1.5rem;
  --space-xl: 2rem;
  --space-xxl: 3rem;

  /* Border radius */
```

```
  --radius-sm: 6px;
  --radius-md: 8px;
  --radius-lg: 12px;
  --radius-xl: 16px;

  /* Shadows */
  --shadow-sm: 0 2px 8px rgba(0, 0, 0, 0.2);
  --shadow-md: 0 4px 16px rgba(0, 0, 0, 0.3);
  --shadow-lg: 0 8px 32px rgba(0, 0, 0, 0.4);
  --shadow-xl: 0 16px 64px rgba(0, 0, 0, 0.5);

  /* Transitions */
  --transition-fast: all 0.15s ease;
  --transition-normal: all 0.25s ease;
  --transition-slow: all 0.35s ease;
}

/* Base styles */
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

body {
  font-family: var(--font-main);
  background: var(--bg-primary);
  color: var(--text-primary);
  display: flex;
  justify-content: center;
  align-items: center;
  min-height: 100vh;
  padding: var(--space-xl);
  line-height: 1.6;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
}
```

```css
/* Container styles */
.container {
  width: 100%;
  max-width: 420px;
  background: var(--bg-secondary);
  padding: var(--space-xxl);
  border-radius: var(--radius-xl);
  box-shadow: var(--shadow-xl);
  border: 1px solid var(--border-color);
  backdrop-filter: blur(8px);
  transition: var(--transition-normal);
}

.container:hover {
  box-shadow: var(--shadow-lg);
}

/* Title styles */
.title {
  text-align: center;
  margin-bottom: var(--space-lg);
  font-size: var(--font-size-xl);
  font-weight: 600;
  letter-spacing: -0.02em;
  background: linear-gradient(90deg, var(--accent-primary), #7a9dff);
  -webkit-background-clip: text;
  -webkit-text-fill-color: transparent;
  background-clip: text;
}

/* Form styles */
form {
  display: flex;
  flex-direction: column;
  gap: var(--space-md);
}
```

```css
.form-group {
  display: flex;
  flex-direction: column;
  gap: var(--space-sm);
}

.form-label {
  font-size: 0.875rem;
  color: var(--text-secondary);
  font-weight: 500;
  transition: var(--transition-fast);
}

.form-control {
  padding: var(--space-md);
  border-radius: var(--radius-md);
  border: 1px solid var(--border-color);
  background: var(--bg-tertiary);
  color: var(--text-primary);
  font-family: inherit;
  font-size: var(--font-size-base);
  transition: var(--transition-normal);
}

.form-control:focus {
  outline: none;
  border-color: var(--accent-primary);
  box-shadow: 0 0 0 2px rgba(90, 139, 255, 0.2);
}

/* Button styles */
.btn {
  width: 100%;
  padding: var(--space-md);
  border-radius: var(--radius-md);
  border: none;
```

```
  background: var(--accent-primary);
  color: white;
  font-family: inherit;
  font-size: var(--font-size-base);
  font-weight: 600;
  cursor: pointer;
  transition: var(--transition-normal);
  display: inline-flex;
  justify-content: center;
  align-items: center;
  gap: var(--space-sm);
  text-transform: uppercase;
  letter-spacing: 0.03em;
}

.btn:hover {
  background: var(--accent-secondary);
  transform: translateY(-2px);
}

.btn:active {
  transform: translateY(0);
}

.btn:focus {
  outline: none;
  box-shadow: 0 0 0 3px rgba(90, 139, 255, 0.3);
}

/* Response styles */
.response {
  margin-top: var(--space-lg);
  padding: var(--space-md);
  border-radius: var(--radius-md);
  background: var(--bg-tertiary);
  color: var(--text-primary);
  border: 1px solid var(--border-color);
```

```
  opacity: 0;
  transform: translateY(-10px);
  transition: var(--transition-normal);
  pointer-events: none;
}

.response.show {
  opacity: 1;
  transform: translateY(0);
  pointer-events: auto;
}

/* Responsive styles */
@media (max-width: 768px) {
  .container {
    padding: var(--space-xl);
  }

  .title {
    font-size: 1.3rem;
  }
}

@media (max-width: 480px) {
  body {
    padding: var(--space-md);
  }

  .container {
    border-radius: var(--radius-lg);
    padding: var(--space-lg);
  }
}

/* Scrollbar styles */
::-webkit-scrollbar {
  width: 10px;
```

```
  height: 10px;
}

::-webkit-scrollbar-track {
  background: var(--bg-secondary);
}

::-webkit-scrollbar-thumb {
  background: var(--accent-primary);
  border-radius: var(--radius-sm);
}

::-webkit-scrollbar-thumb:hover {
  background: var(--accent-secondary);
}

/* Accessibility styles */
:focus-visible {
  outline: 2px solid var(--accent-primary);
  outline-offset: 2px;
}
```

Приложение А4. Исходный код server.js

```
const express = require("express");
const path = require("path");
const bodyParser = require("body-parser");

// init
const app = express();
const port = 8080;

console.log("[INIT] Starting server...");
console.log("[PATH] __dirname =", __dirname);
console.log("[PATH] static dir =", path.resolve(__dirname, "public"));

// логируем каждый запрос
```

```
app.use((req, res, next) => {
  console.log(`[REQ] ${req.method} ${req.url}`);
  next();
});

// логируем раздачу статики
app.use(
  express.static(path.resolve(__dirname, "public"), {
    setHeaders(res, filePath) {
      console.log(`[STATIC] Sent file: ${filePath}`);
    },
  })
);

app.use(bodyParser.urlencoded({ extended: true }));

const list = [];

// логируем GET /list
app.get("/list", (req, res) => {
  console.log("[GET /list] returning list:", list);
  res.send(list);
});

// логируем отдачу index.html
app.get("/", (req, res) => {
  const file = path.resolve(__dirname, "index.html");
  console.log("[GET /] sending", file);
  res.sendFile(file);
});

// логируем POST
app.post("/", (req, res) => {
  console.log("[POST /] BODY =", req.body);
  list.push(req.body);
  console.log("[POST /] list now =", list);
  res.send("OK");
```

```
});

// запуск сервера
app.listen(port, () => {
  console.log(`Server running at http://localhost:${port}`);
});
```