# Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования «Рыбинский государственный авиационный технический университет имени П. А. Соловьева»

Кафедра математического и программного обеспечения электронно-вычислительных средств

## ЛАБОРАТОРНАЯ РАБОТА №3

по дисциплине «Web-программирование» на тему

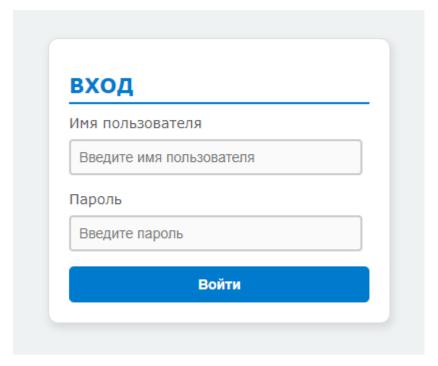
«HTML / CSS»

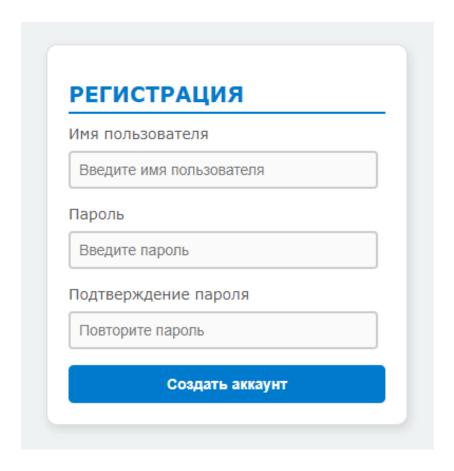
 Студент группы ИПБ-22
 Ушаков М. С.

 Преподаватель ассистент
 Пруктишина В. А.

В ходе выполнения лабораторной работы необходимо:

1. Написать формы регистрации и входа на HTML Форма регистрации и входа:





# Содержание файла Reg\_Login.html:

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <meta name="description" content="Форма регистрации и входа на сайте">
  <meta name="author" content="Ушаков">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Пользовательские формы</title>
  <link rel="stylesheet" href="styles.css">
</head>
<!-- Блок формы входа -->
<div class="form-box">
  <h1 class="form-header">Вход</h1>
  <form>
    <label for="signin-username">Имя пользователя</label>
    <input type="text" id="signin-username" placeholder="Введите имя пользователя" required>
    <label for="signin-password">Пароль</label>
    <input type="password" id="signin-password" placeholder="Введите пароль" required>
    <button type="submit">Войти</button>
  </form>
</div>
<body>
  <!-- Блок формы регистрации -->
  <div class="form-box">
    <h1 class="form-header">Регистрация</h1>
    <form>
      <label for="signup-username">Имя пользователя</label>
      <input type="text" id="signup-username" placeholder="Введите имя пользователя" required>
      <label for="signup-password">Пароль</label>
      <input type="password" id="signup-password" placeholder="Введите пароль" required>
      <label for="signup-confirm">Подтверждение пароля</label>
      <input type="password" id="signup-confirm" placeholder="Повторите пароль" required>
      <button type="submit">Создать аккаунт</button>
    </form>
  </div>
</body>
</html>
```

## Содержание файла Styles.css:

```
body {
 font-family: "Verdana", sans-serif;
 background-color: #eef2f3;
 margin: 0;
 padding: 50px;
 display: grid;
 place-items: center;
 height: 100vh;
 box-sizing: border-box;
}
h1 {
 font-size: 26px;
 text-align: left;
 color: #333333;
 margin-bottom: 15px;
}
/* Заголовки для форм */
.form-header {
 font-size: 20px;
 text-transform: uppercase;
 color: #007acc;
 margin-bottom: 10px;
 border-bottom: 2px solid #007acc;
 padding-bottom: 5px;
/* Стиль контейнеров форм */
.form-box {
 width: 300px;
 padding: 20px;
 background-color: #ffffff;
 border: 1px solid #dddddd;
 border-radius: 10px;
 box-shadow: 2px 4px 8px rgba(0, 0, 0, 0.1);
 margin-bottom: 20px;
label {
 font-size: 14px;
 color: #555555;
 display: block;
 margin-bottom: 8px;
}
input {
 width: 91%;
 padding: 8px;
 margin-bottom: 15px;
 border: 2px solid #ccccc;
 border-radius: 4px;
```

```
font-size: 14px;
 background-color: #fafafa;
 transition: border-color 0.3s ease;
input:focus {
 border-color: #007acc;
 outline: none;
 background-color: #ffffff;
button {
 width: 100%;
 padding: 10px;
 font-size: 14px;
 font-weight: bold;
 background-color: #007acc;
 color: #ffffff;
 border: none;
 border-radius: 5px;
 cursor: pointer;
 transition: background-color 0.3s ease, transform 0.2s;
button:hover {
 background-color: #005f99;
 transform: scale(1.02);
}
```

# 2. Ответить на вопросы:

#### 1. Что такое лендинг?

Лендинг (Landing Page) — это одностраничный сайт, основной целью которого является конверсия посетителей в покупателей, подписчиков или клиентов.

#### Особенности:

- 1. Фокус на одной цели (регистрация, покупка, скачивание и т.д.).
- 2. Минимум отвлекающих элементов.
- 3. Используются яркие кнопки, продающий текст, визуальные элементы и форма захвата данных.

## 2. SPA, MPA и PWA — что это?

SPA (Single Page Application)

Одностраничное приложение, которое загружается единожды, а последующие изменения интерфейса происходят динамически без перезагрузки страницы. Примеры: Gmail, Trello.

#### Плюсы:

- 1. Быстродействие.
- 2. Плавность работы.

## Минусы:

- 1. SEO сложнее, чем у MPA.
- 2. Сложность в реализации.

MPA (Multi Page Application)

Многостраничное приложение, где каждая страница загружается с сервера. Примеры: интернет-магазины, классические корпоративные сайты.

#### Плюсы:

- 1. Простота реализации.
- 2. Хорошая SEO-оптимизация.

## Минусы:

- 1. Дольше загрузка между страницами.
- 2. Сложность в поддержке больших приложений.

PWA (Progressive Web Application)

Прогрессивное веб-приложение, которое совмещает лучшие черты веба и мобильных приложений. Примеры: Starbucks, Twitter Lite.

#### Плюсы:

- 1. Работает оффлайн.
- 2. Возможность установки на устройство.

# Минусы:

1. Ограничения функционала на iOS.

## 3. Почему лучше разбирать сразу фреймворк, а не чистый JS?

Разбирать фреймворк (например, React, Vue, Angular) сразу может быть выгодно, так как:

- 1. Практичность: Фреймворки используются в реальных проектах, что быстрее подводит к работе над реальными задачами.
- 2. Скорость: они упрощают рутинные задачи (например, управление состоянием, работа с DOM).
- 3. Сообщество: Фреймворки имеют обширную документацию, шаблоны и готовые решения.

# 4. Что такое roadmap frontend?

Roadmap Frontend — это дорожная карта для изучения фронтенда, состоящая из последовательных шагов (основы HTML/CSS/JS, фреймворки, инструменты сборки и т.д.).

Можно ли стать фронтенд-разработчиком за 30 минутный видеоролик? Нет. Однако видео может быть хорошей отправной точкой для понимания основных концепций. Становление специалистом требует месяцев или лет практики.

# 5. Принципы S.O.L.I.D, KISS и YAGNI

- S.O.L.I.D это набор принципов проектирования, которые помогают создавать более понятный и поддерживаемый код:
  - 1. Single Responsibility Principle (SRP): Каждый класс или модуль должен иметь только одну причину для изменения. То есть, каждый элемент программы должен быть ответственен только за одну задачу.
  - 2. Open-Closed Principle (OCP): Код должен быть открыт для расширения, но закрыт для изменений. То есть, добавление нового функционала не должно требовать изменений в уже существующем коде.
  - 3. Liskov Substitution Principle (LSP): Объекты наследующих классов должны быть полностью заменяемы объектами базового класса без нарушения корректности работы программы.
  - 4. Interface Segregation Principle (ISP): Интерфейсы должны быть узкоспециализированными. Модули не должны зависеть от интерфейсов, которые они не используют.
  - 5. Dependency Inversion Principle (DIP): Модули верхнего уровня должны зависеть от абстракций, а не от конкретных реализаций. Это позволяет уменьшить зависимость кода от конкретных классов и улучшить его тестируемость.

KISS (Keep It Simple, Stupid) — принцип, который говорит, что код должен быть как можно более простым. Не стоит усложнять решение задачи, если есть более простые и понятные варианты. Простота делает код более читаемым и поддерживаемым.

YAGNI (You Aren't Gonna Need It) — принцип, который предполагает, что не стоит писать код, который может понадобиться в будущем, но на данный момент не решает никаких текущих задач. Лучше фокусироваться только на тех функциях, которые действительно требуются.

#### 6. OWASP и CORS — что это?

OWASP (Open Web Application Security Project) — это глобальная организация, которая занимается изучением, анализом и улучшением безопасности веб-приложений. Она разрабатывает рекомендации, методологии и инструменты для защиты приложений от различных угроз.

CORS (Cross-Origin Resource Sharing) — это механизм безопасности, который регулирует доступ веб-приложений к ресурсам, расположенным на других доменах. Например, если приложение пытается запросить данные с внешнего сервера, браузер по умолчанию блокирует такие запросы, если сервер не разрешил доступ к данным с этого источника. CORS позволяет серверу отправлять специальные заголовки, которые разрешают браузерам выполнять такие запросы, гарантируя при этом безопасность.