# Техническая документация: Веб-инструмент проверки доступности сайтов

## 1. Цель и задачи проекта

Цель данного проекта — разработка веб-инструмента, который позволит проводить автоматизированную проверку сайтов   
малого и среднего бизнеса на соответствие стандартам доступности для пользователей с ограниченными возможностями.   
Инструмент должен анализировать HTML-страницы на наличие нарушений рекомендаций WCAG (Web Content Accessibility Guidelines),   
включая проверку альтернативного текста у изображений, цветового контраста, структуры заголовков, навигации с клавиатуры   
и корректности ARIA-атрибутов.

Дополнительные задачи:  
- Создание удобного веб-интерфейса для ввода URL и отображения отчёта о найденных нарушениях.  
- Использование исключительно открытых и бесплатных инструментов.  
- Возможность сохранения истории проверок в базе данных PostgreSQL.  
- Генерация отчётов с рекомендациями по исправлению нарушений.

## 2. Архитектура системы

Система построена по клиент-серверному принципу и состоит из четырёх основных компонентов:  
1. Пользовательский интерфейс (Frontend)  
2. Серверная часть (Backend)  
3. Модуль анализа доступности  
4. База данных

Пользователь взаимодействует с системой через веб-интерфейс. При вводе URL анализируемого сайта запрос   
отправляется на сервер через REST API. Сервер обрабатывает запрос, инициирует модуль анализа, который с помощью   
headless-браузера (Puppeteer) загружает страницу и передаёт её библиотеке axe-core для анализа на соответствие WCAG.   
Результаты проверки сохраняются в PostgreSQL и возвращаются пользователю в виде отчёта.

### Текстовое представление архитектурной диаграммы:

[Пользователь (браузер)]   
 ↓   
[Frontend (Vue.js + Bootstrap)]   
 ↓ (HTTP POST /api/analyze)  
[Backend (Flask или Express)]   
 ↓ (Puppeteer + axe-core)  
[Модуль анализа доступности]   
 ↓ (результаты JSON)  
[PostgreSQL (хранение истории и отчётов)]

## 3. Компоненты системы

Frontend (Vue.js + Bootstrap):  
- Обеспечивает интерфейс для пользователя.  
- Реализует ввод URL и отображение отчёта.  
- Использует адаптивный дизайн и динамическое обновление данных.  
  
Backend (Flask или Node.js + Express):  
- Реализует REST API для связи с фронтендом.  
- Запускает Puppeteer, внедряет axe-core и получает JSON с результатами анализа.  
- Сохраняет результаты в базе данных и возвращает ответ пользователю.  
  
Модуль анализа (axe-core + Puppeteer):  
- Загружает веб-страницу, исполняет JavaScript и анализирует DOM.  
- Проверяет контраст, alt-теги, структуру заголовков, ARIA и доступность с клавиатуры.  
- Формирует список ошибок и рекомендаций в формате JSON.  
  
База данных (PostgreSQL):  
- Хранит историю анализов, параметры сайтов, отчёты и логи.  
- Обеспечивает доступ к предыдущим проверкам для сравнения динамики доступности.

## 4. Логика работы инструмента

1. Пользователь вводит адрес сайта и нажимает кнопку "Проверить".  
2. Frontend отправляет запрос POST /api/analyze на сервер.  
3. Сервер вызывает Puppeteer, который загружает страницу в headless-браузере.  
4. axe-core сканирует DOM и выявляет нарушения стандартов WCAG.  
5. Результаты сохраняются в PostgreSQL.  
6. Пользователь получает отчёт с ошибками и рекомендациями.

## 5. Проверки на соответствие WCAG AA

Инструмент проводит автоматическую проверку по ключевым критериям WCAG уровня AA:  
- Проверка наличия и корректности alt-текстов.  
- Анализ контрастности текста и фона.  
- Проверка правильного использования заголовков и семантики.  
- Анализ доступности клавиатурной навигации.  
- Проверка ARIA-атрибутов и ролей.  
- Анализ корректности форм, меток и подсказок.  
- Проверка мультимедиа на наличие субтитров и описаний.  
- Проверка корректного указания языка страницы и её частей.  
- Проверка предотвращения «ловушек фокуса» и доступности всех элементов управления.  
- Контроль за анимацией, мерцанием и автообновлением контента.

### Пример формируемого отчёта:

1. Ошибка: Изображение без альтернативного текста (WCAG 1.1.1)  
 Рекомендация: Добавить атрибут alt="Описание изображения".  
  
2. Ошибка: Контраст текста и фона ниже 4.5:1 (WCAG 1.4.3)  
 Рекомендация: Изменить цвет текста или фона.  
  
3. Ошибка: Элемент без видимого фокуса (WCAG 2.4.7)  
 Рекомендация: Добавить CSS-правило :focus.

## 6. Глоссарий

WCAG — Web Content Accessibility Guidelines, международный стандарт по обеспечению доступности сайтов.  
ARIA — Accessible Rich Internet Applications, технология для описания поведения динамических элементов.  
Axe-core — open-source библиотека для автоматизированной проверки доступности.  
Puppeteer — инструмент для автоматизации браузера Chrome.  
Headless-браузер — браузер без графического интерфейса, используемый для анализа страниц.  
Контраст — отношение яркости между цветами текста и фона.  
Screen reader — программа, озвучивающая содержимое страницы для незрячих пользователей.  
REST API — способ взаимодействия клиент-серверных приложений по HTTP.