PROJECT ABSTRACT

# Website Accessibility Checker with WCAG Scoring

*An automated WCAG 2.2 auditing web app that crawls up to 25 pages, runs 58 accessibility checks across the four POUR principles, and produces per-page scores with severity-ranked issues and code-level fix guidance. Delivered built-to-order with source, check-rule documentation, report, PPT and viva kit.*

## Title

Website Accessibility Checker with WCAG Scoring — an automated auditor that turns WCAG 2.2 success criteria into executable checks, page scores, and developer-ready remediation guidance.

## Introduction / Background

One in six people lives with a significant disability, yet most websites still fail basic accessibility: images without alt text, text that fails contrast, dropdowns a keyboard cannot operate. WCAG 2.2 is the accepted standard, but reading the guidelines does not tell a team which of their pages break which rules. Automated auditing bridges that gap: it crawls a site, evaluates each page against machine-testable success criteria, and converts the findings into a score developers can track and a fix list they can execute. This project builds such an auditor — a crawler, a rule engine of 58 checks mapped to WCAG 2.2 AA, and a reporting interface that explains every failure with the exact elements involved.

## Problem Statement

Teams ship inaccessible pages not from indifference but from invisibility: without tooling, nobody knows the site has 48 contrast failures across 9 pages. Manual review is slow and inconsistent, and generic linters check code rather than rendered pages. The project addresses this with a crawler-based auditor that evaluates real rendered pages against 58 automated WCAG 2.2 checks, scores each page on a 0–100 scale, ranks issues by severity, and shows the failing elements with concrete fix steps — making accessibility measurable and actionable.

## Objectives

- Crawl a target site up to a configurable page limit (25-page default) and render pages for analysis.
- Implement 58 automated checks mapped to WCAG 2.2 success criteria across Perceivable, Operable, Understandable and Robust.
- Compute a transparent 0–100 per-page score from severity-weighted issue counts.
- Rank issues by severity (critical, serious, moderate, minor) with affected-page counts.
- Show failing elements per issue with contrast ratios, selectors and code-level fix snippets.
- Provide scan history with score trends so fixes are verifiable over time.

## Proposed Methodology

The system is built in three stages. (1) Crawling: a headless browser visits the seed URL, discovers same-origin links up to the page limit, and captures the rendered DOM plus computed styles per page. (2) Rule engine: 58 checks run against each page — contrast ratios computed from rendered colors, alt-text presence on images, form-label association, keyboard-focus visibility, ARIA roles and names, heading order, and link purpose. (3) Scoring and reporting: issue counts are severity-weighted into a 0–100 score, issues are grouped by criterion with per-element evidence, and fix guidance is generated from templates per rule.

## System Architecture / Working

A scan job fans out page fetches to a worker pool; each worker renders the page, runs the check suite, and stores issues with element selectors and measured values (e.g. 2.9:1 contrast). The scorer aggregates severity-weighted deductions into the page score and site summary. The web UI presents the dashboard (principle cards, recent scans), the report view (score ring, severity distribution, top issues) and the issue-detail view (failing elements, how-to-fix steps, code diff, WCAG reference). Scan history persists so re-scans show score movement.

## Software Requirements

- Python 3 with Django (scan orchestration, scoring, reporting UI)
- Headless Chromium via Playwright (page rendering and style computation)
- PostgreSQL (sites, scans, issues, history)
- Celery with Redis (background scan jobs)
- HTML, CSS and JavaScript (dashboard, report and issue views)
- Docker (one-command deployment)

## Expected Outcomes

- A working auditor that scans a multi-page site and reports a scored, severity-ranked issue list.
- 58 documented checks with their WCAG 2.2 criterion mapping and test logic.
- Per-issue fix guidance with failing-element evidence and code snippets.
- A complete report, PPT and viva Q&A; (crawling, contrast computation, WCAG structure, scoring design).

## Applications

Accessibility audits for college, NGO and small-business websites; the check engine pattern adapts to SEO or performance auditing with different rule sets.

## Future Scope

Keyboard-navigation path simulation for focus-order checks, PDF/CSV export of reports, scheduled re-scans with regression alerts, and axe-core cross-validation of check results.