VA | U.S. Department of Veterans Affairs

# Visual Regression Testing

**VA.gov | Testing Tools Team**

**Joe Tice, PM, VSP Testing Tools Team**
joe.tice@oddball.io

September 2, 2021

# Agenda

- What is the problem?
- What is the solution?
- What options are available?
- Value proposition
- Questions / discussion

# What is the problem?

## Problem Statement:

VFS Teams do not currently have a means to verify that changes are not indirectly causing regressions elsewhere in the platform. VSP personnel do not have any means of continuously scanning for visual regressions on the platform as changes are integrated.

How might we verify that newly integrated changes are not inadvertently causing regressions on the platform?

How might we collect information about visual regressions into a report or dashboard?

***User Research Pain Point: VFS Teams recommend that we have visual regression testing***

*(Source: Testing Tools User Research Survey - Q1 2021)*

- *"Provide some help/advice on  [or better yet, tools for automated Regression-testing"*
- *"Maybe BrowserStack's Percy could help automated some visual testing?"*
- *"Percy.io for visual testing is pretty cool"*

# What is the solution?

## Hypothesis:

Given the ability to observe the impact to regressions on the platform immediately upon integrating their changes, VFS teams can fix regressions with context and precision.

A tighter feedback loop for visual regressions will empower teams to resolve issues more quickly.

**Visual Regression Testing will provide what we need to address the problem.**

# What is Visual Regression Testing?

## An Overview:

Regression Testing involves validating that new changes that are introduced to our code do not have unintended impacts to the overall system.

Visual Regression Testing serves the same purpose, but is more specifically focused on the interface that is presented to end users. We do this by comparing baseline reference images of the user interface (UI) to newly generated screenshots captured using the code being integrated.

# What does Visual Regression Testing Provide?

## Benefits:

1. Automatically identify issues in the CI/CD pipeline before Veterans see the problem in production
2. Precise comparisons of visual changes in multiple browsers with no slowdown for additional browsers
3. GitHub Integration facilitates triage/output of only genuine issues
4. Leverage existing E2E test specs to execute visual testing with snapshots in key stages of participating team's products
5. Percy's webhooks will provide trackable metrics to push data to BigQuery for dashboard integration

# Our Research into Options...

We reviewed 15 solutions and approaches to Visual Regression Testing, of those, Percy was the clear choice

| Pros | Cons |
| --- | --- |
| Mature organization w/ a significant cloud presence | Has a financial cost |
| Support for parallelization | Can not verify visual regressions w/ Internet Explorer<br>*(**Note** - this was not a requirement for this solution, but was a nice-to-have, so did not affect our decision-making)* |
| Support for GitHub Actions | |
| Support for webhooks<br>*(useful for pushing updates to dashboards)* | |
| Robust review/approval mechanism | |

# Percy Value proposition

- **Provides:**

## Speed

High-speed rendering - "...snapshotting without impacting the performance of your existing CI speed."



Parallelization - "... run in parallelized CI services or parallel test runners."

## Ease of Use

Efficient reviews - "...it's easy to review visual changes in even the most complex applications."

Commit Synchronization - "...reviews are always in sync with your workflow."

## Precision

Snapshot stabilization - "Our proprietary snapshot stabilization technology minimizes false positives."

Pixel-by-pixel diffs - "Percy detects the smallest changes in your UI."

## Integration

CI service support - "Leverage existing workflows and tests by integrating your CI/CD service..."



Workflow integrations - "...easy to incorporate visual reviews and maintain the source of truth on every commit."

- **Cost: $23,988/year**
  - 500,000 screenshots/month *(Based on current projections of the number of screenshots needed)*

# Next steps

*If approved…*

- Purchase the tool
- Implement in GHA workflow & support
- Identify metrics to track
- Create Domo cards to display metrics

# Questions / Discussion