

The Visual Testing Revolution!



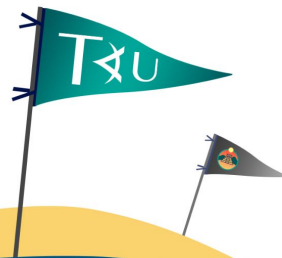
Andrew Knight

Automation Panda
Applitools Developer Advocate
Test Automation University Director



Become a test automation superstar!

All Courses Free!



testautomationu.applitools.com

Visual Testing

Types of Testing



**Testing =
Interaction + Verification**

**Testing =
Interaction + Verification**



You do something!

Testing = Interaction + Verification



You do something!



You make sure it works!

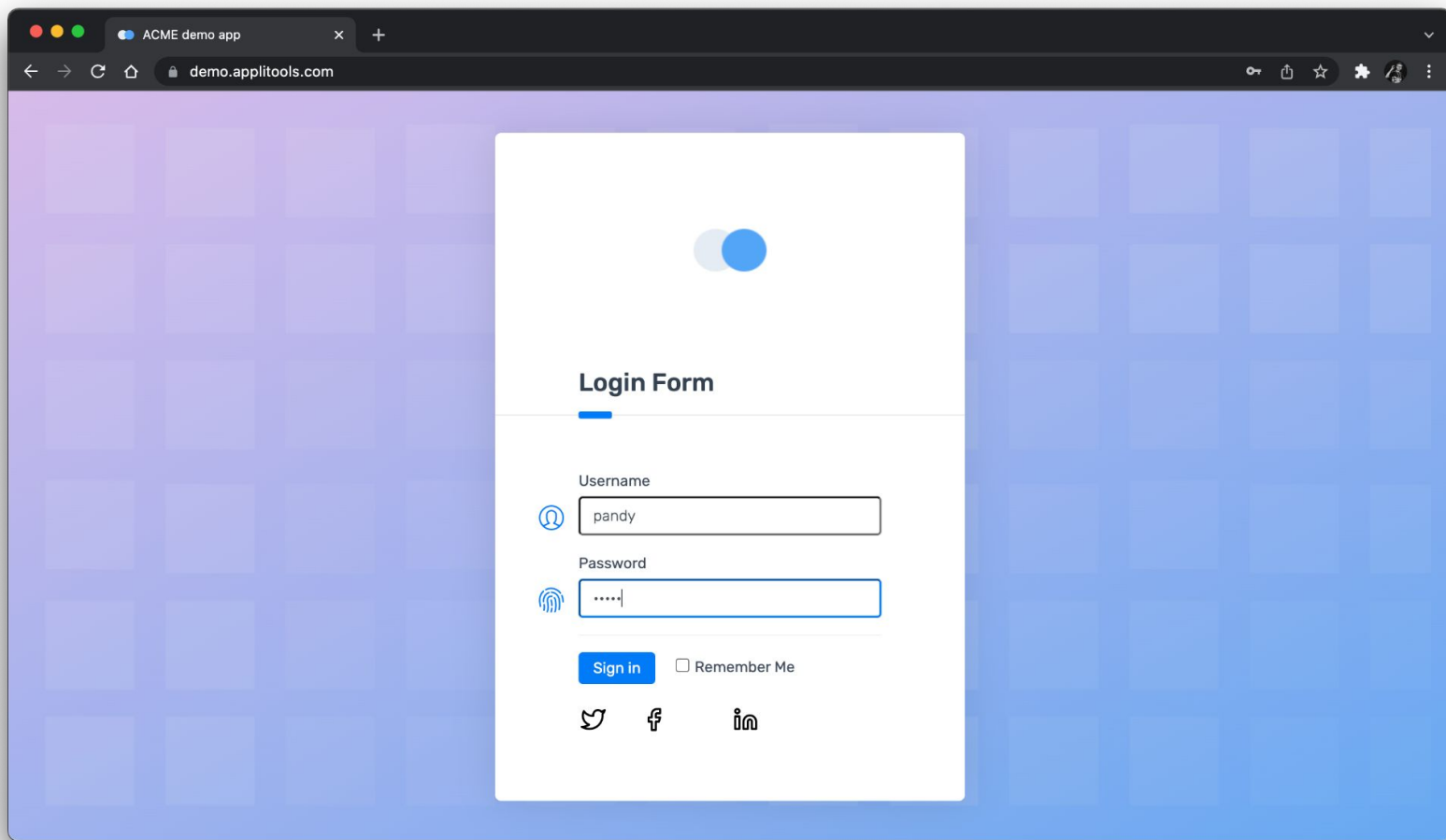
VISUAL **Testing =** **AUTONOMOUS** **Interaction + Verification**

You do something!

You make sure it works!

With snapshots!

Visual testing is **easier**
than traditional test automation.



ACME demo app x +

demo.applitools.com/app.html

ACME

Start typing to search...

7

Jack Gomez
CUSTOMER

Your nearest branch closes in: 30m 5s

Add Account Make Payment

Financial Overview

Total Balance
\$350 %7 ↓
[View Statement](#) >

Credit Available
\$17,800
[Request Increase](#) >

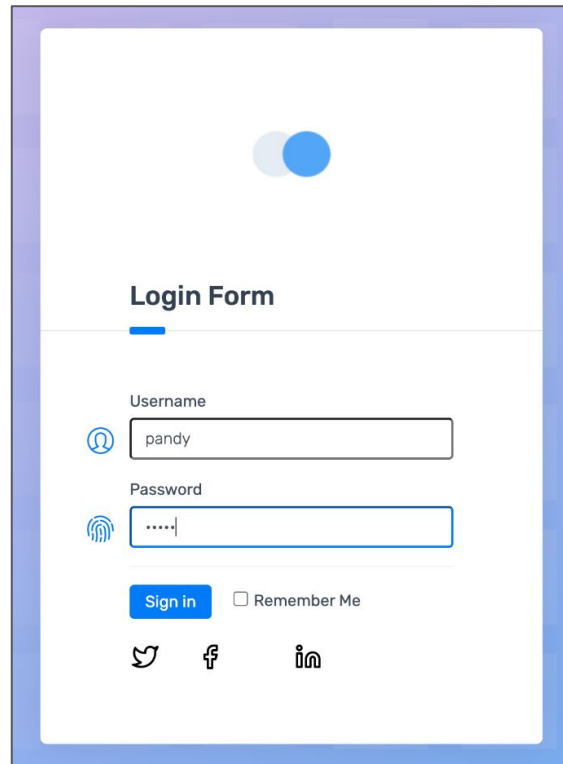
Due Today
\$180
[Pay Now](#) >

Recent Transactions

STATUS	DATE	DESCRIPTION	CATEGORY	AMOUNT
Complete	Today 1:52am	Starbucks coffee	Restaurant / Cafe	+ 1,250 USD
Declined	Jan 19th 3:22pm	stripe Stripe Payment Processing	Finance	+ 952.23 USD
Pending	Yesterday 7:45am	MailChimp Services	Software	- 320 USD
Pending	Jan 23rd 2:7pm	Shopify product	Shopping	+ 17.99 USD

Login test case

1. Load the login page.
2. Verify that the login page loads correctly.
3. Log into the app.
4. Verify that the main page loads correctly.



A login form interface is shown, enclosed in a blue border. At the top right, there is a toggle switch with a blue circle. Below it, the text "Login Form" is displayed in bold, followed by a horizontal line. The form contains two input fields: "Username" with a blue circular icon to its left and the text "pandy" inside, and "Password" with a blue fingerprint icon to its left and masked text "...." inside. Below the password field is a "Sign in" button with a blue background and white text, followed by a checkbox and the text "Remember Me". At the bottom, there are three social media icons: Twitter, Facebook, and LinkedIn.

```
public class LoginTest {  
  
    private WebDriver driver;  
    private WebDriverWait wait;  
  
    @Test  
    public void login() {  
        loadLoginPage();  
        verifyLoginPage();  
        performLogin();  
        verifyMainPage();  
    }  
  
    // ...  
}
```

```
private void loadLoginPage() {  
    driver.get("https://demo.applitools.com");  
}
```

```
private void waitForAppearance(By locator) {  
    wait.until(d -> d.findElements(locator).size() > 0);  
}  
  
private void verifyLoginPage() {  
    waitForAppearance(By.cssSelector("div.logo-w"));  
    waitForAppearance(By.id("username"));  
    waitForAppearance(By.id("password"));  
    waitForAppearance(By.id("log-in"));  
    waitForAppearance(By.cssSelector("input.form-check-input"));  
}
```

```
private void performLogin() {  
    driver.findElement(By.id("username")).sendKeys("andy");  
    driver.findElement(By.id("password")).sendKeys("i<3pandas");  
    driver.findElement(By.id("log-in")).click();  
}
```



```

private void verifyMainPage() {
    // Check various page elements
    waitForAppearance(By.cssSelector("div.logo-w"));
    waitForAppearance(By.cssSelector("div.element-search.autosuggest-search-activator > input"));
    waitForAppearance(By.cssSelector("div.avatar-w img"));
    waitForAppearance(By.cssSelector("ul.main-menu"));
    waitForAppearance(By.xpath("//a/span[.='Add Account']"));
    waitForAppearance(By.xpath("//a/span[.='Make Payment']"));
    waitForAppearance(By.xpath("//a/span[.='View Statement']"));
    waitForAppearance(By.xpath("//a/span[.='Request Increase']"));
    waitForAppearance(By.xpath("//a/span[.='Pay Now']"));

    // Check time message
    assertTrue(Pattern.matches(
        "Your nearest branch closes in:( \\d+[hms])+",
        driver.findElement(By.id("time")).getText()));

    // Check menu element names
    var menuElements = driver.findElements(By.cssSelector("ul.main-menu li span"));
    var menuItems = menuElements.stream().map(i -> i.getText().toLowerCase()).toList();
    var expected = Arrays.asList("card types", "credit cards", "debit cards", "lending", "loans", "mortgages");
    assertEquals(expected, menuItems);

    // Check transaction statuses
    var statusElements = driver.findElements(By.xpath("//td[./span[contains(@class, 'status-pill')]]/span[2]"));
    var statusNames = statusElements.stream().map(n -> n.getText().toLowerCase()).toList();
    var acceptableNames = Arrays.asList("complete", "pending", "declined");
    assertTrue(acceptableNames.containsAll(statusNames));
}

```

```

private void verifyMainPage() {
    // Check various page elements
    waitForAppearance(By.cssSelector("div.logo-w"));
    waitForAppearance(By.cssSelector("div.element-search.autosuggest-search-activator > input"));
    waitForAppearance(By.cssSelector("div.avatar-w img"));
    waitForAppearance(By.cssSelector("ul.main-menu"));
    waitForAppearance(By.xpath("//a/span[.='Add Account']"));
    waitForAppearance(By.xpath("//a/span[.='Make Payment']"));
    waitForAppearance(By.xpath("//a/span[.='View Statement']"));
    waitForAppearance(By.xpath("//a/span[.='Request Increase']"));
    waitForAppearance(By.xpath("//a/span[.='Pay Now']"));

    // Check time message
    assertTrue(Pattern.matches(
        "Your nearest branch closes in:( \\d+[hms])+",
        driver.findElement(By.id("time")).getText()));

    // Check menu element names
    var menuElements = driver.findElements(By.cssSelector("ul.main-menu li span"));
    var menuItems = menuElements.stream().map(i -> i.getText().toLowerCase()).toList();
    var expected = Arrays.asList("card types", "credit cards", "debit cards", "lending", "loans", "mortgages");
    assertEquals(expected, menuItems);

    // Check transaction statuses
    var statusElements = driver.findElements(By.xpath("//td[./span[contains(@class, 'status-pill')]]/span[2]"));
    var statusNames = statusElements.stream().map(n -> n.getText().toLowerCase()).toList();
    var acceptableNames = Arrays.asList("complete", "pending", "declined");
    assertTrue(acceptableNames.containsAll(statusNames));
}

```

```

private void verifyMainPage() {
    // Check various page elements
    waitForAppearance(By.cssSelector("div.logo-w"));
    waitForAppearance(By.cssSelector("div.element-search.autosuggest-search-activator > input"));
    waitForAppearance(By.cssSelector("div.avatar-w img"));
    waitForAppearance(By.cssSelector("ul.main-menu"));
    waitForAppearance(By.xpath("//a/span[.='Add Account']"));
    waitForAppearance(By.xpath("//a/span[.='Make Payment']"));
    waitForAppearance(By.xpath("//a/span[.='View Statement']"));
    waitForAppearance(By.xpath("//a/span[.='Request Increase']"));
    waitForAppearance(By.xpath("//a/span[.='Pay Now']"));

    // Check time message
    assertTrue(Pattern.matches(
        "Your nearest branch closes in:( \\d+[hms])+",
        driver.findElement(By.id("time")).getText()));

    // Check menu element names
    var menuElements = driver.findElements(By.cssSelector("ul.main-menu li span"));
    var menuItems = menuElements.stream().map(i -> i.getText().toLowerCase()).toList();
    var expected = Arrays.asList("card types", "credit cards", "debit cards", "lending", "loans", "mortgages");
    assertEquals(expected, menuItems);

    // Check transaction statuses
    var statusElements = driver.findElements(By.xpath("//td[./span[contains(@class, 'status-pill')]]/span[2]"));
    var statusNames = statusElements.stream().map(n -> n.getText().toLowerCase()).toList();
    var acceptableNames = Arrays.asList("complete", "pending", "declined");
    assertTrue(acceptableNames.containsAll(statusNames));
}

```

```

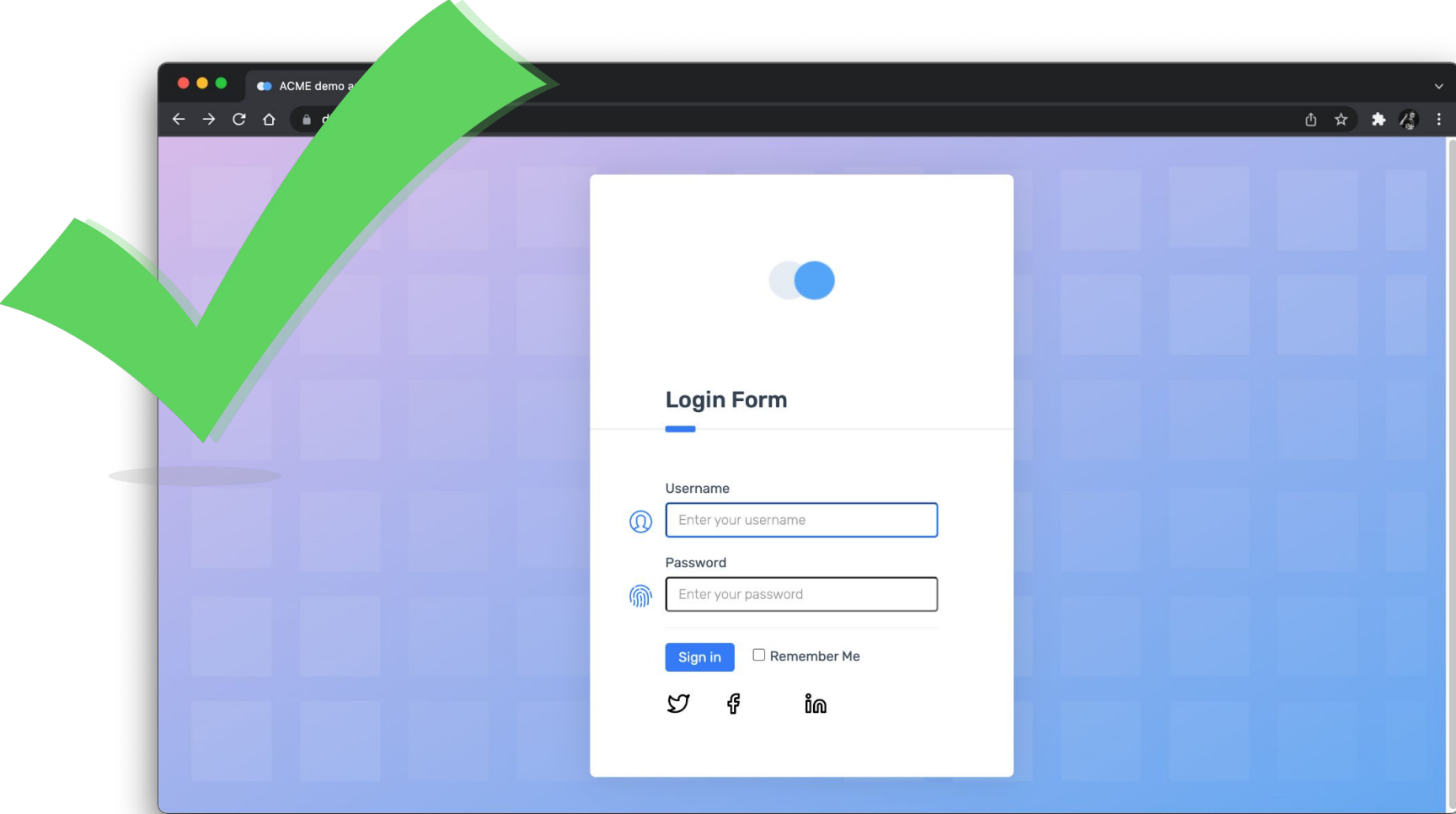
private void verifyMainPage() {
    // Check various page elements
    waitForAppearance(By.cssSelector("div.logo-w"));
    waitForAppearance(By.cssSelector("div.element-search.autosuggest-search-activator > input"));
    waitForAppearance(By.cssSelector("div.avatar-w img"));
    waitForAppearance(By.cssSelector("ul.main-menu"));
    waitForAppearance(By.xpath("//a/span[.='Add Account']"));
    waitForAppearance(By.xpath("//a/span[.='Make Payment']"));
    waitForAppearance(By.xpath("//a/span[.='View Statement']"));
    waitForAppearance(By.xpath("//a/span[.='Request Increase']"));
    waitForAppearance(By.xpath("//a/span[.='Pay Now']"));

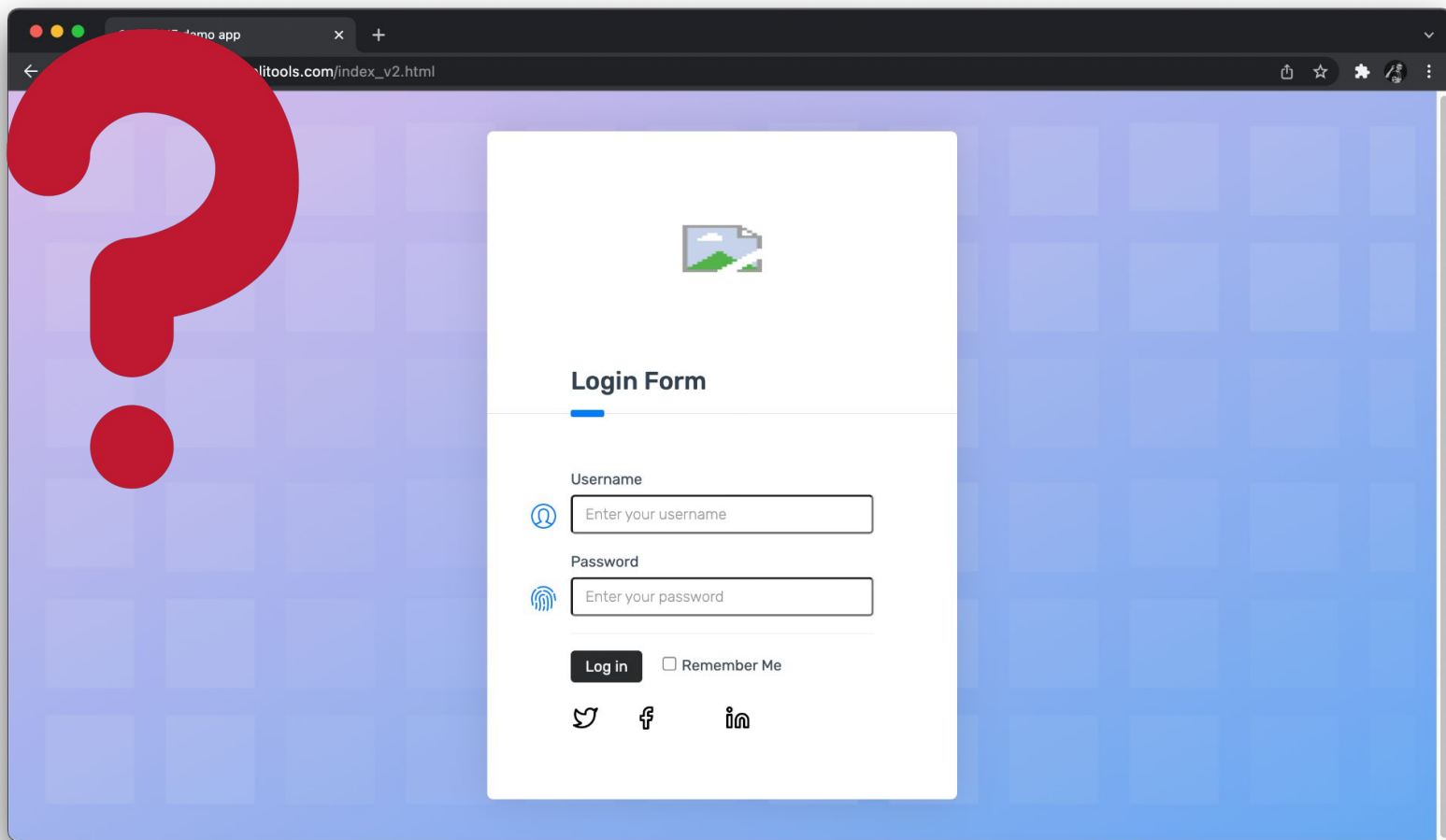
    // Check time message
    assertTrue(Pattern.matches(
        "Your nearest branch closes in:( \\d+[hms])+",
        driver.findElement(By.id("time")).getText()));

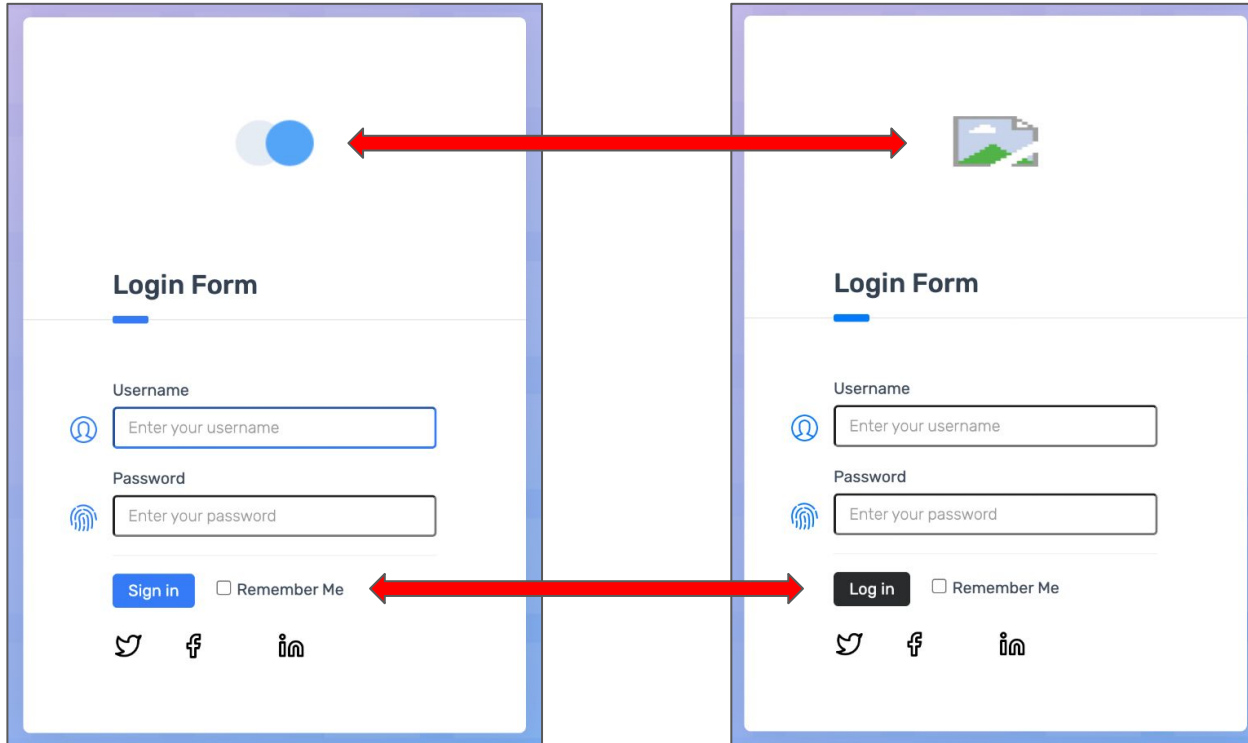
    // Check menu element names
    var menuElements = driver.findElements(By.cssSelector("ul.main-menu li span"));
    var menuItems = menuElements.stream().map(i -> i.getText().toLowerCase()).toList();
    var expected = Arrays.asList("card types", "credit cards", "debit cards", "lending", "loans", "mortgages");
    assertEquals(expected, menuItems);

    // Check transaction statuses
    var statusElements = driver.findElements(By.xpath("//td[./span[contains(@class, 'status-pill')]]/span[2]"));
    var statusNames = statusElements.stream().map(n -> n.getText().toLowerCase()).toList();
    var acceptableNames = Arrays.asList("complete", "pending", "declined");
    assertTrue(acceptableNames.containsAll(statusNames));
}

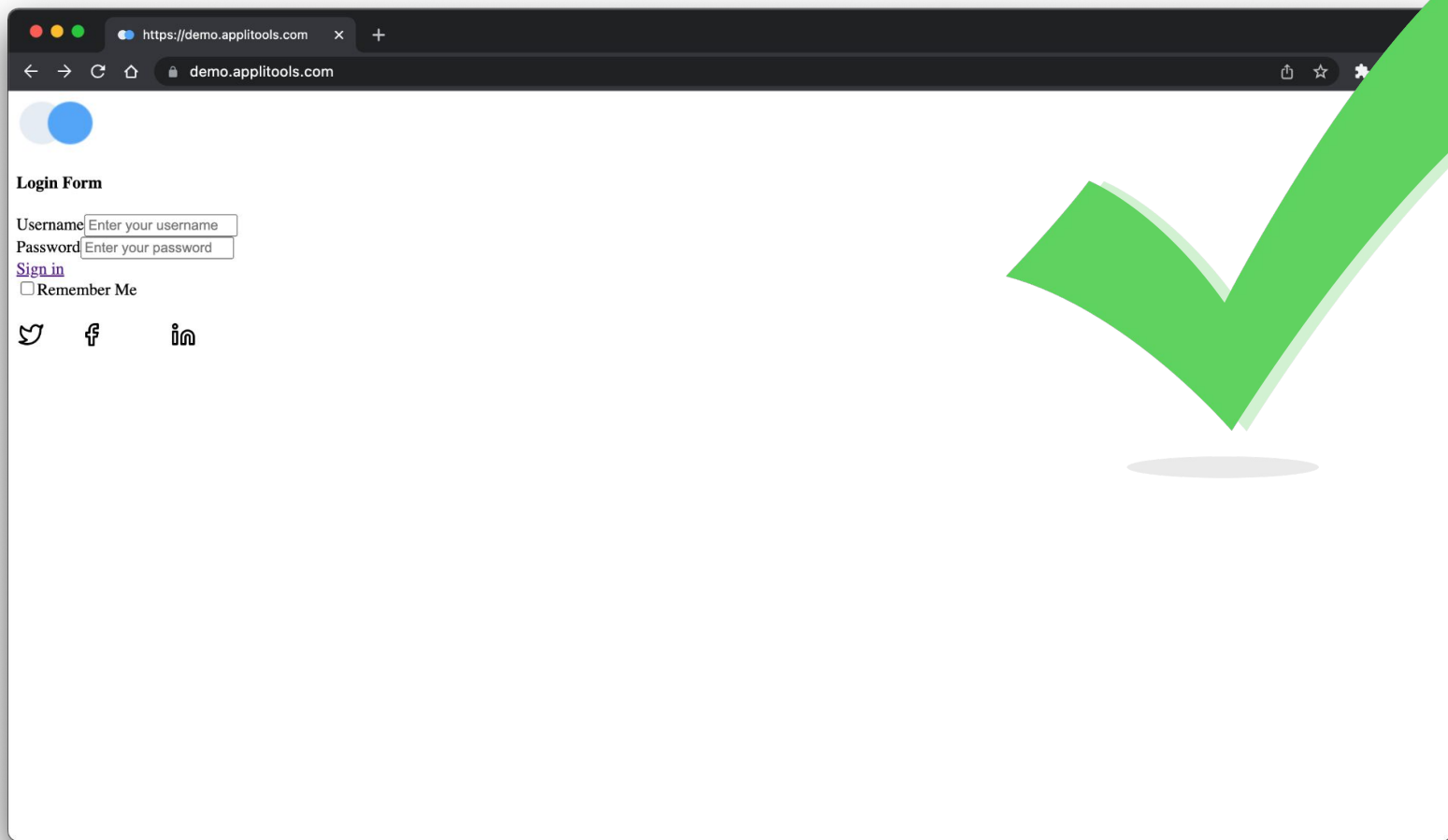
```







```
private void waitForAppearance(By locator) {  
    wait.until(d -> d.findElements(locator).size() > 0);  
}  
  
private void verifyLoginPage() {  
    waitForAppearance(By.cssSelector("div.logo-w"));  
    waitForAppearance(By.id("username"));  
    waitForAppearance(By.id("password"));  
    waitForAppearance(By.id("log-in"));  
    waitForAppearance(By.cssSelector("input.form-check-input"));  
}
```

Step 1/2: Login page

Test: A visual login test

VIEW

HIGHLIGHT DIFFS

ANNOTATIONS

AUTO MAINTENANCE

Scope: Default

ACCESSIBILITY

Baseline | 1/2 Login page

Strict | Linux | Chrome 98.0 | 800x600 | Desktop

Checkpoint | 1/2 Login page

Strict | Linux | Chrome 98.0 | 800x600 | Desktop

100%

+

-

Step 1/2: Login page

Test: A visual login test

VIEW

HIGHLIGHT DIFFS

ANNOTATIONS

AUTO MAINTENANCE

Scope: Default

ACCESSIBILITY

Baseline | 1/2 Login page

Strict | Linux | Chrome 98.0 | 800x600 | Desktop

Checkpoint | 1/2 Login page

Strict | Linux | Chrome 98.0 | 800x600 | Desktop

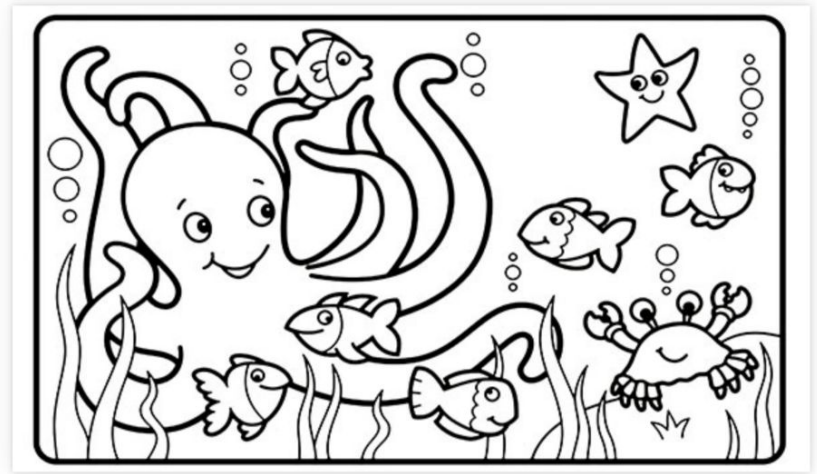
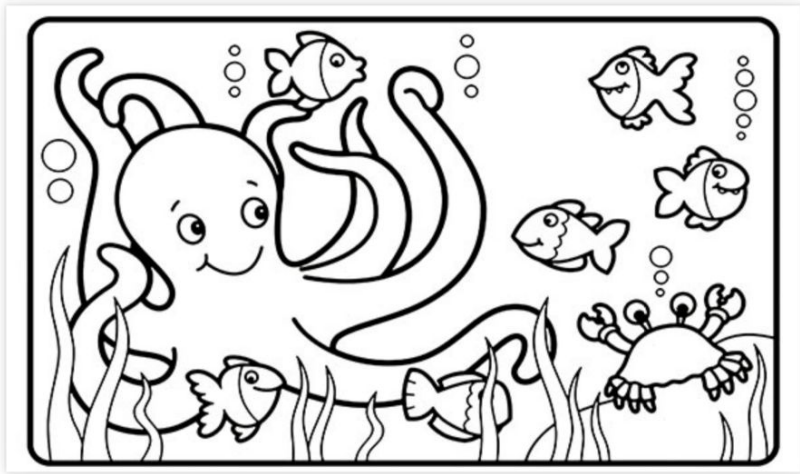
Advantage #1:

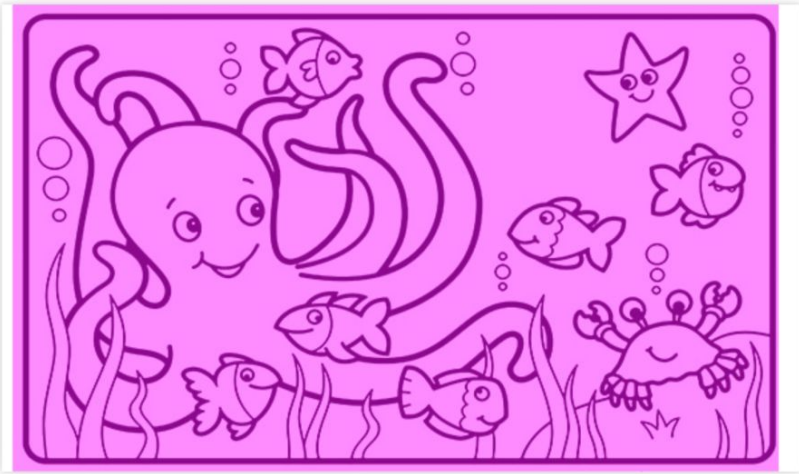
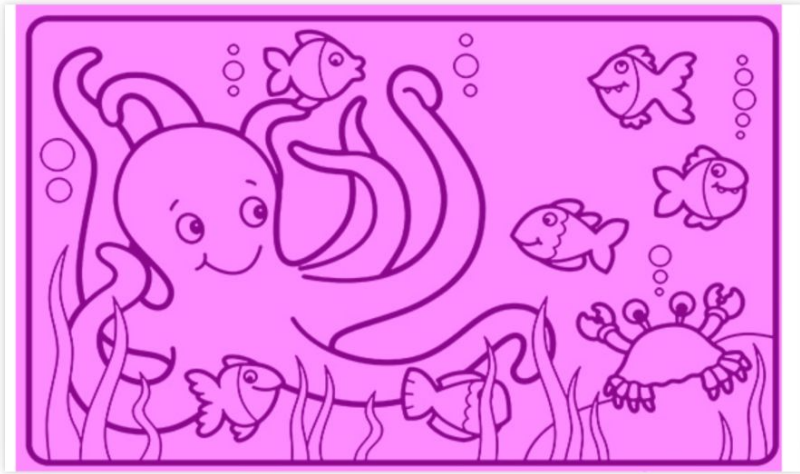
Visual testing covers
everything on a page.

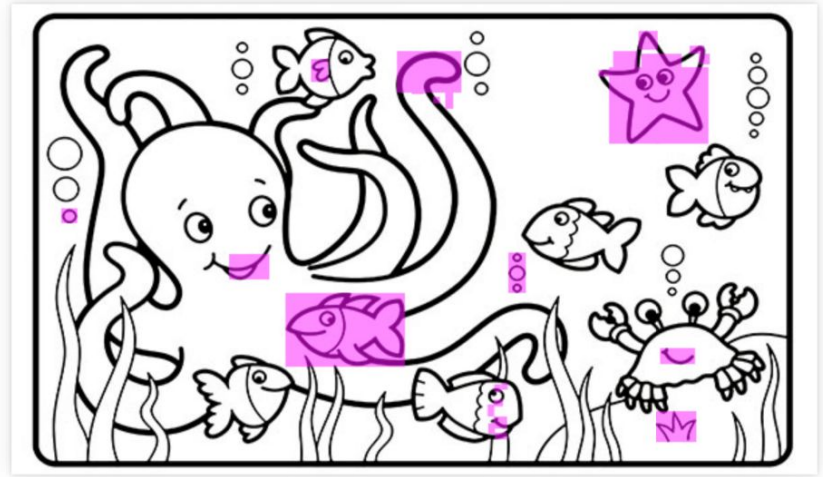
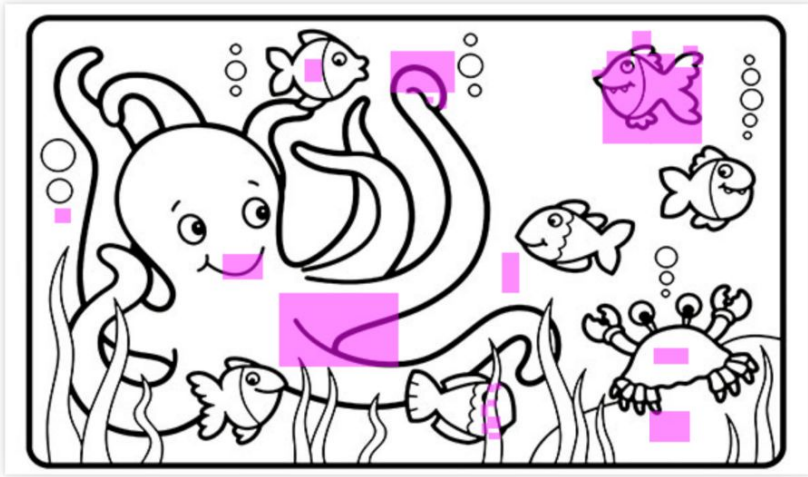


John Henry statue and the Great Bend Tunnel

(Image source: <https://www.nps.gov/neri/planyourvisit/the-legend-of-john-henry-talcott-wv.htm>)

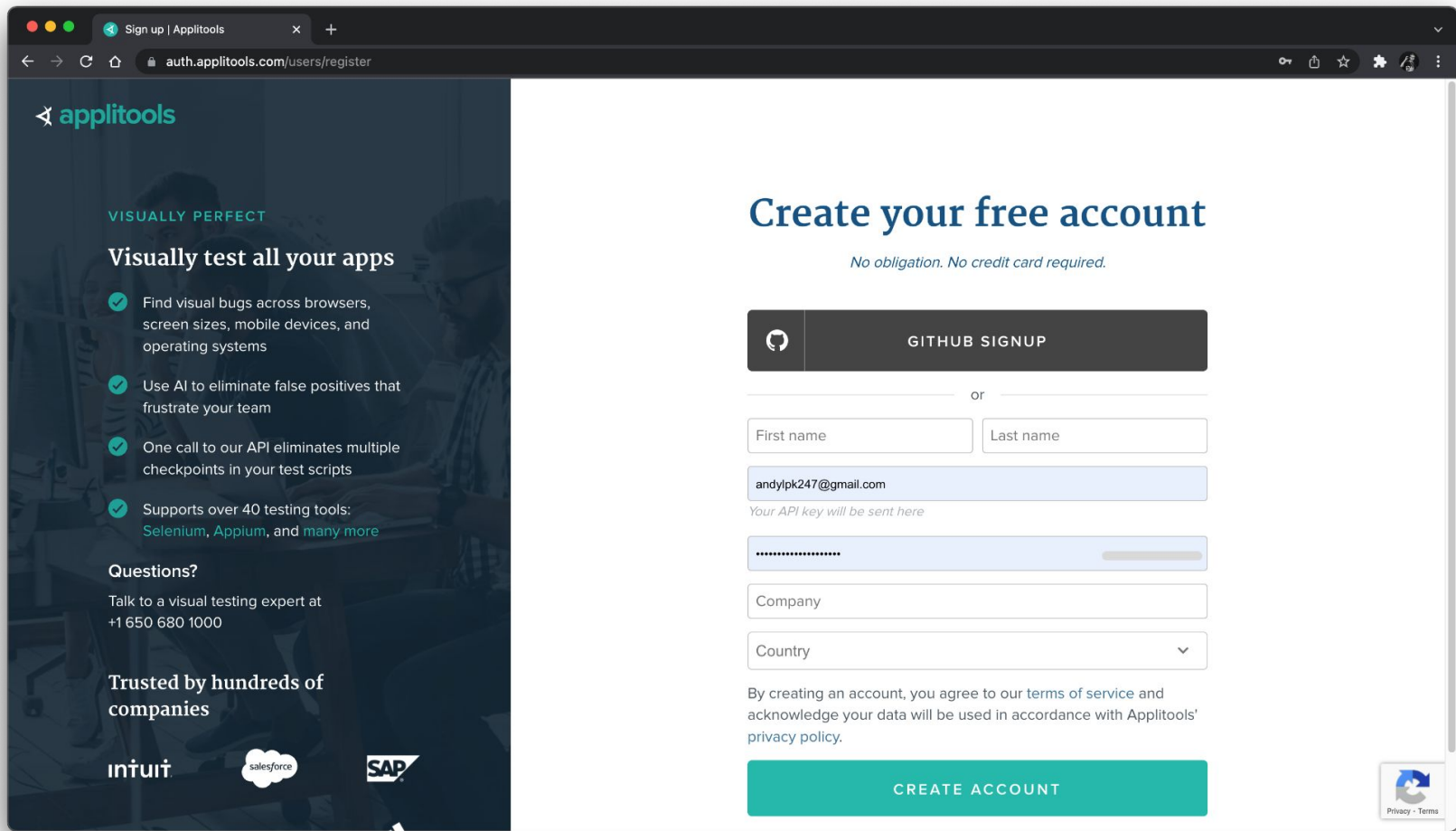






Advantage #2:

Visual AI focuses on
meaningful changes to avoid noise.



```
<dependency>
  <groupId>com.applitools</groupId>
  <artifactId>eyes-selenium-java3</artifactId>
  <version>3.211.0</version>
  <scope>test</scope>
</dependency>
```

```
private WebDriver driver;  
private VisualGridRunner runner;  
private Eyes eyes;  
  
@BeforeEach  
public void setUpVisualAI() {  
    driver = new ChromeDriver();  
    runner = new VisualGridRunner();  
    eyes = new Eyes(runner);  
  
    Configuration config = eyes.getConfiguration();  
    config.setApiKey(System.getenv("APPLITOLS_API_KEY"));  
    config.setBatch(new BatchInfo("A Visual Testing Revolution"));  
    config.addBrowser(800, 600, BrowserType.CHROME);  
  
    eyes.setConfiguration(config);  
}
```

```
@Test
public void login() {
    try {
        eyes.open(driver, "Applitools Demo App", "A visual login test");

        loadLoginPage();
        verifyLoginPage();
        performLogin();
        verifyMainPage();

        eyes.closeAsync();
    }
    finally {
        eyes.abortAsync();
    }
}
```

```
private void verifyLoginPage() {  
    // waitForAppearance(By.cssSelector("div.logo-w"));  
    // waitForAppearance(By.id("username"));  
    // waitForAppearance(By.id("password"));  
    // waitForAppearance(By.id("log-in"));  
    // waitForAppearance(By.cssSelector("input.form-check-input"));  
  
    eyes.check(Target.window().fully().withName("Login page"));  
}
```

```

private void verifyMainPage() {
    // Check various page elements
    waitForAppearance(By.cssSelector("div.logo-w"));
    waitForAppearance(By.cssSelector("div.element-search.autosuggest-search-activator > input"));
    waitForAppearance(By.cssSelector("div.avatar-w img"));
    waitForAppearance(By.cssSelector("ul.main-menu"));
    waitForAppearance(By.xpath("//a/span[.='Add Account']"));
    waitForAppearance(By.xpath("//a/span[.='Make Payment']"));
    waitForAppearance(By.xpath("//a/span[.='View Statement']"));
    waitForAppearance(By.xpath("//a/span[.='Request Increase']"));
    waitForAppearance(By.xpath("//a/span[.='Pay Now']"));

    // Check time message
    assertTrue(Pattern.matches(
        "Your nearest branch closes in:( \\d+[hms])+",
        driver.findElement(By.id("time")).getText()));

    // Check menu element names
    var menuElements = driver.findElements(By.cssSelector("ul.main-menu li span"));
    var menuItems = menuElements.stream().map(i -> i.getText().toLowerCase()).toList();
    var expected = Arrays.asList("card types", "credit cards", "debit cards", "lending", "loans", "mortgages");
    assertEquals(expected, menuItems);

    // Check transaction statuses
    var statusElements = driver.findElements(By.xpath("//td[./span[contains(@class, 'status-pill')]]/span[2]"));
    var statusNames = statusElements.stream().map(n -> n.getText().toLowerCase()).toList();
    var acceptableNames = Arrays.asList("complete", "pending", "declined");
    assertTrue(acceptableNames.containsAll(statusNames));
}

```

```

eyes.check(Target.window().fully()
    .withName("Login page"));

```

Advantage #3:

A snapshot is worth a thousand assertions.

Let's demo!

<https://github.com/AutomationPanda/visual-testing-revolution>



Advantage #4:

Visual snapshots enable lightning-fast
cross-browser testing.

When should a team adopt visual testing?

From the **start**.

Advantage #5:

Visual testing makes
functional testing **easier.**

5 Key Advantages of Visual Testing

1. Visual testing covers **everything** on a page.
2. Visual AI focuses on **meaningful changes** to avoid noise.
3. A **snapshot** is worth a thousand assertions.
4. Visual snapshots enable lightning-fast **cross-browser testing**.
5. Visual testing makes functional testing **easier**.



<https://automationpanda.com/2022/06/01/modernizing-software-quality-assurance-with-visual-testing/>

<https://github.com/AutomationPanda/visual-testing-revolution>



The Visual Testing Revolution!



Andrew Knight

Automation Panda
Applitools Developer Advocate
Test Automation University Director