

How Visual AI Makes Testing a Breeze!

Andrew Knight

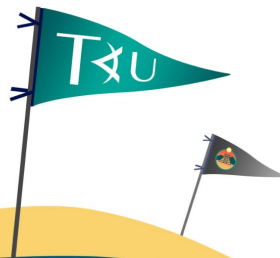
Automation Panda
Applitools Developer Advocate
Test Automation University Director





Become a test automation superstar!

All Courses Free!



testautomationu.applitools.com

Visual AI

Types of Testing



**Testing =
Interaction + Verification**

**Testing =
Interaction + Verification**



You do something!

**Testing =
Interaction + Verification**

You do something!

You make sure it works!

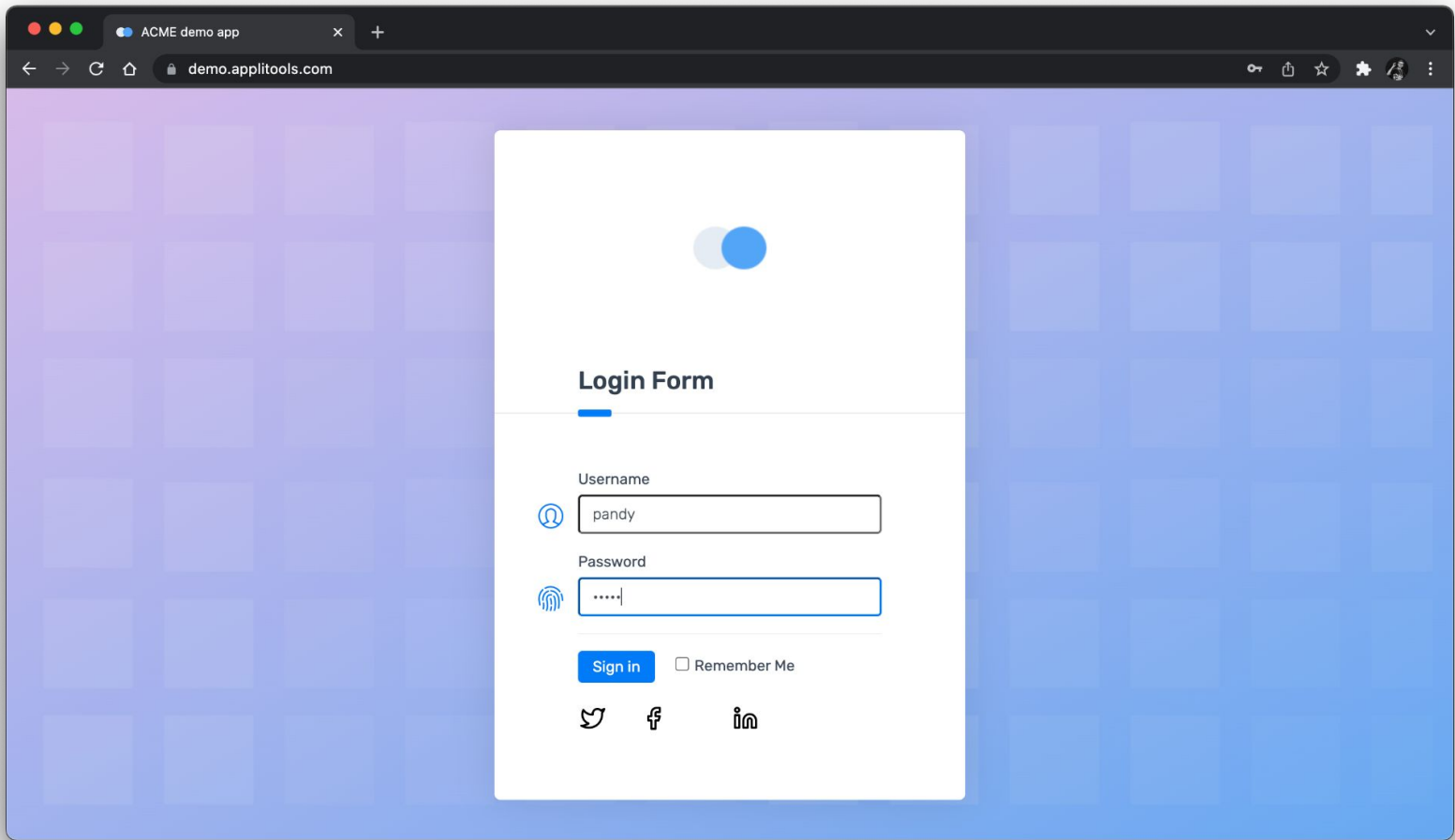
VISUAL
Testing =
AUTONOMOUS
Interaction + Verification

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

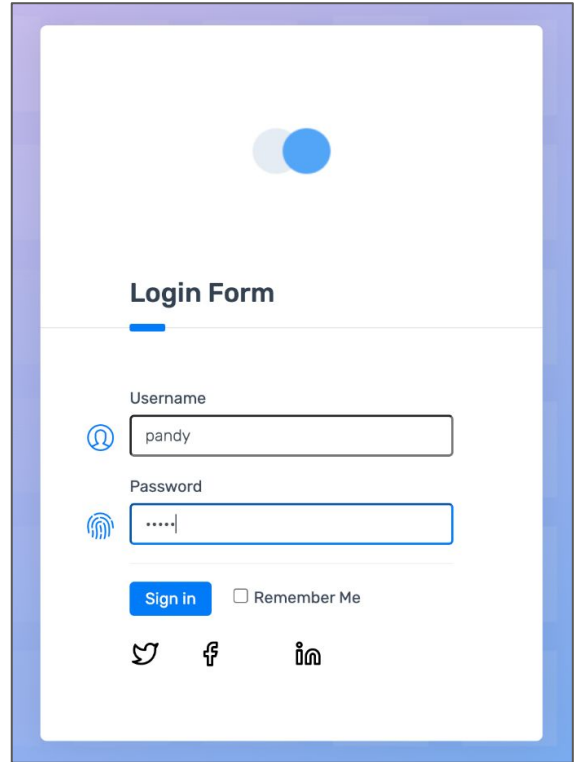
Total Balance	Credit Available	Due Today
\$350 %7 ↓ View Statement >	\$17,800 Request Increase >	\$180 Pay Now >

Recent Transactions

STATUS	DATE	DESCRIPTION	CATEGORY	AMOUNT
● Complete	Today 1:52am	Starbucks coffee	Restaurant / Cafe	+ 1,250 USD
● Declined	Jan 19th 3:22pm	Stripe 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 screenshot of a login form interface. At the top, there is a toggle switch that is currently turned on (blue). Below the toggle is the heading "Login Form" with a blue underline. The form contains two input fields: "Username" with the text "pandy" and "Password" with masked characters "....". Below the password field is a "Sign in" button and a "Remember Me" checkbox. 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));
}

```

```

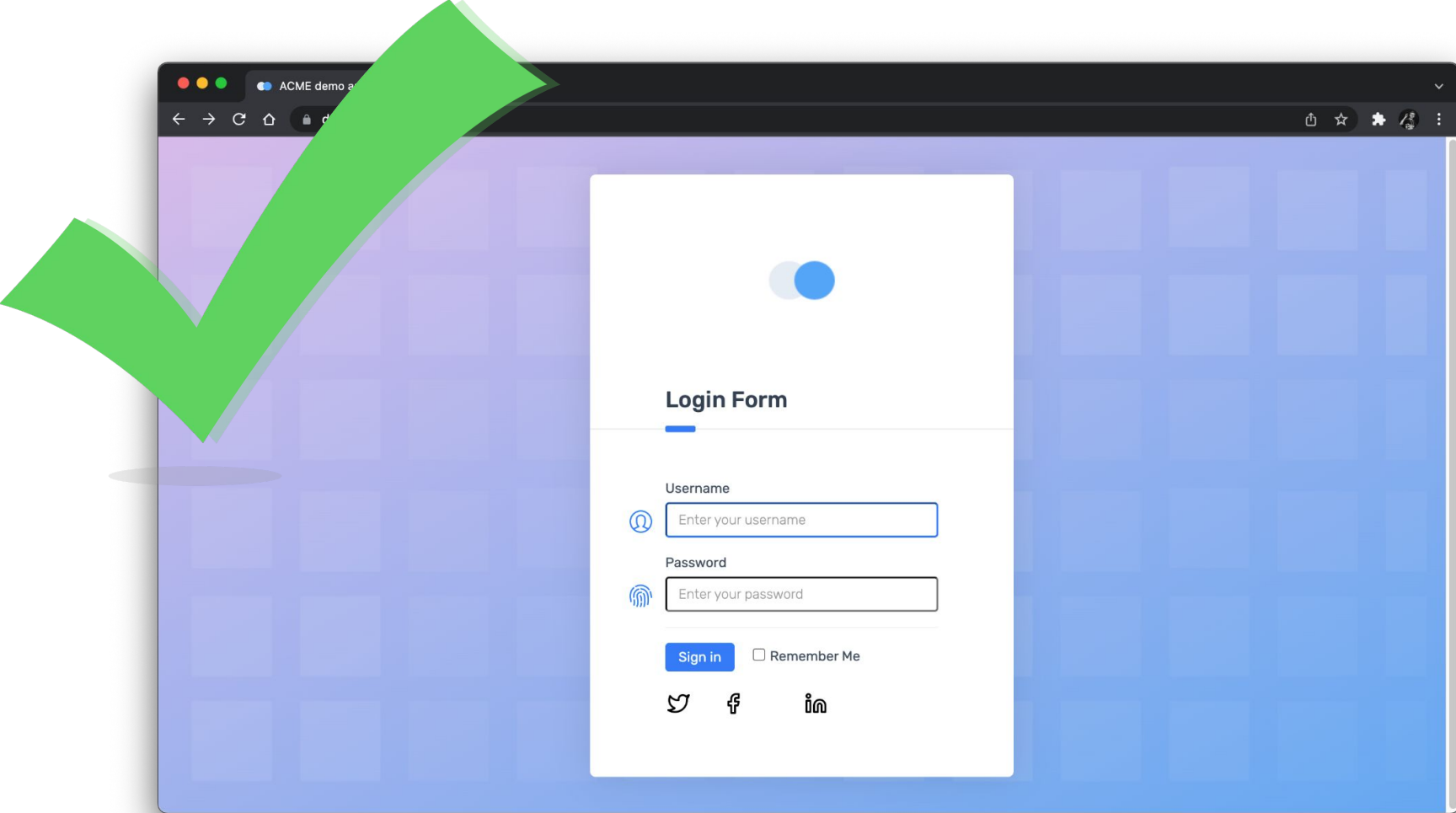
private void verifyMainPage() {
    // Check various page elements
    waitForAppearance(By.cssSelector("div.logo-w"));
    waitForAppearance(By.cssSelector("div.element-search.autosuggest-search-activator > input"));
    waitForAppearance(By.cssSelector("div.avatar-w img"));
    waitForAppearance(By.cssSelector("ul.main-menu"));
    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']"));

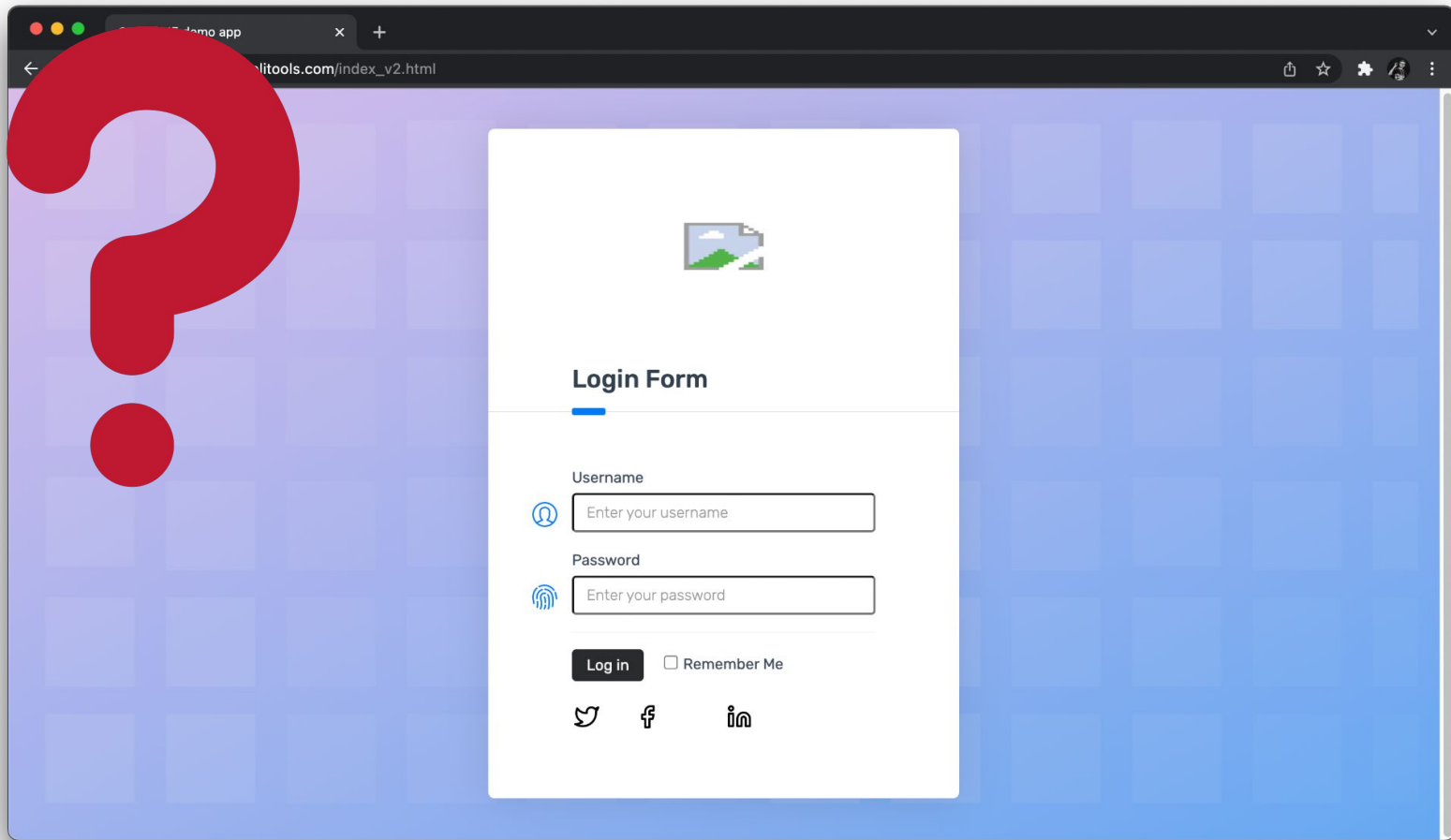
    // Check time message
    assertTrue(Pattern.matches(
        "Your nearest branch closes in:( \\d+[hms])+",
        driver.findElement(By.id("time")).getText()));

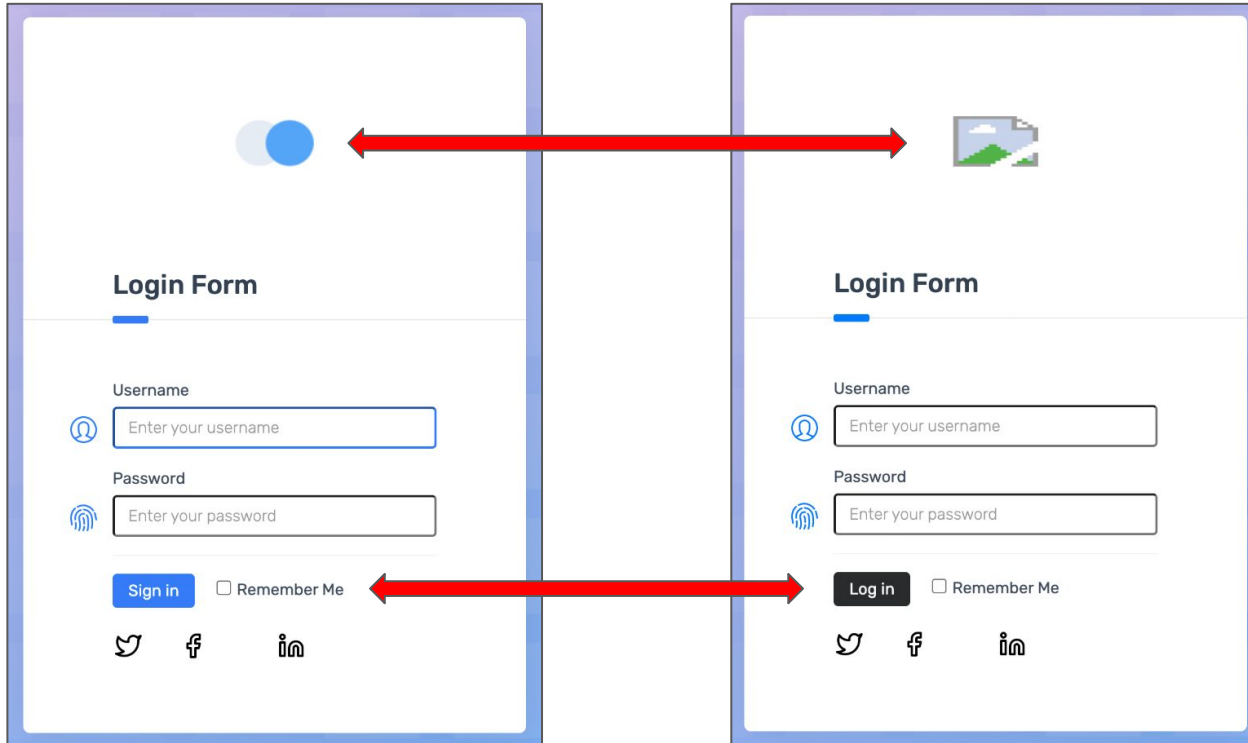
    // Check menu element names
    var menuElements = driver.findElements(By.cssSelector("ul.main-menu li span"));
    var menuItems = menuElements.stream().map(i -> i.getText().toLowerCase()).toList();
    var expected = Arrays.asList("card types", "credit cards", "debit cards", "lending", "loans", "mortgages");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

```

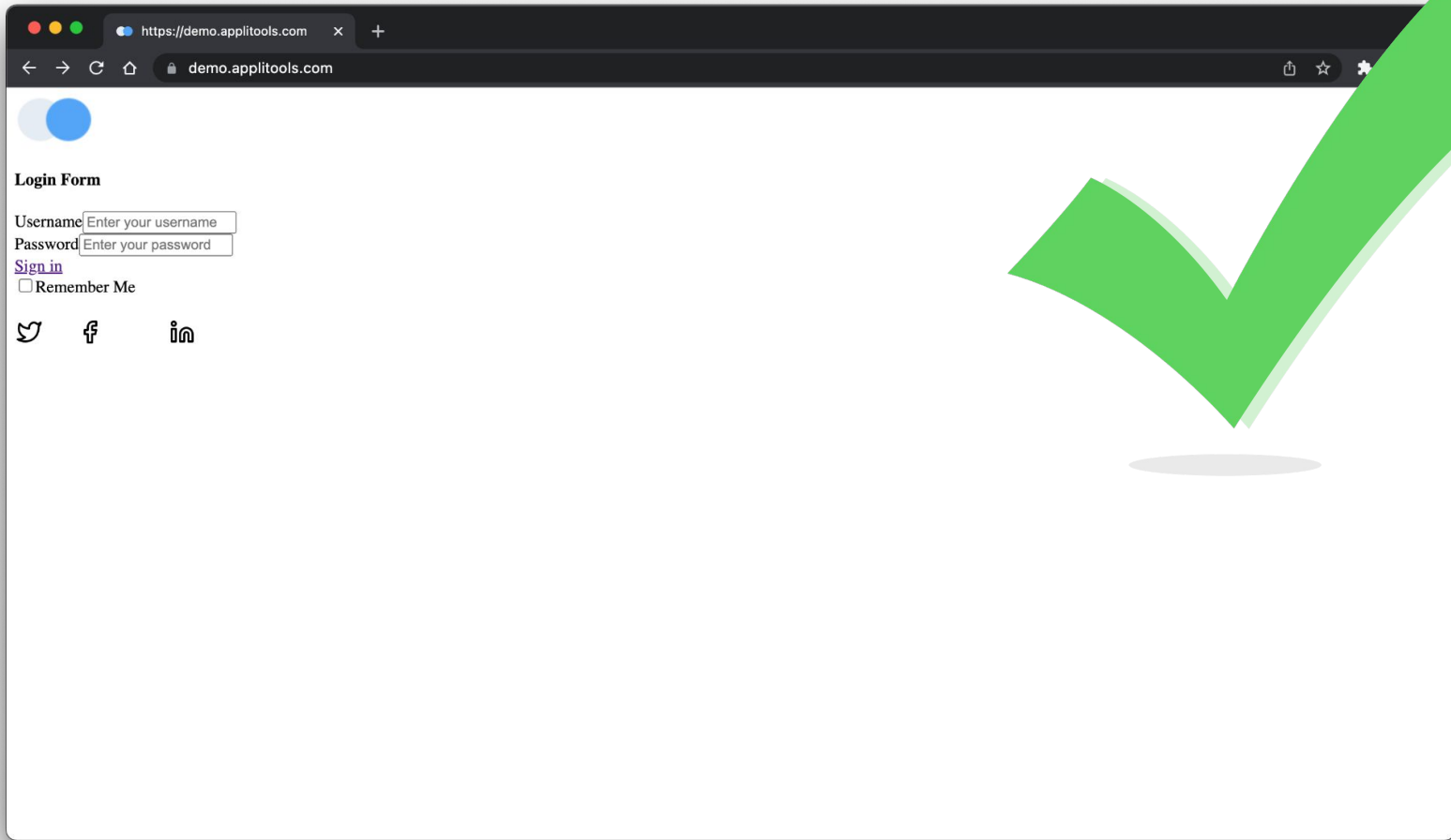






```
private void waitForAppearance (By locator) {
    wait.until(d -> d.findElements(locator).size() > 0);
}

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 ACCESSIBILITY

☰ → ☰ 📄 ↶ A/B 🖥️ 👁️ ⋮ ⏪ 🔄 ⏩ ⚠️ 💬 📏 IGNORE Scope: Default 🚫 👍 👎

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 ACCESSIBILITY

Scope: Default

Baseline | 1/2 Login page
Strict | Linux | Chrome 98.0 | 800x600 | Desktop

Checkpoint | 1/2 Login page
Strict | Linux | Chrome 98.0 | 800x600 | Desktop

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%

+

-

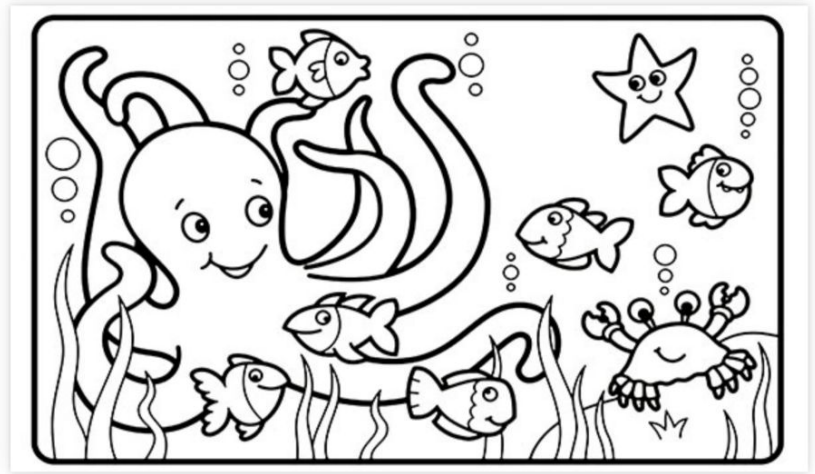
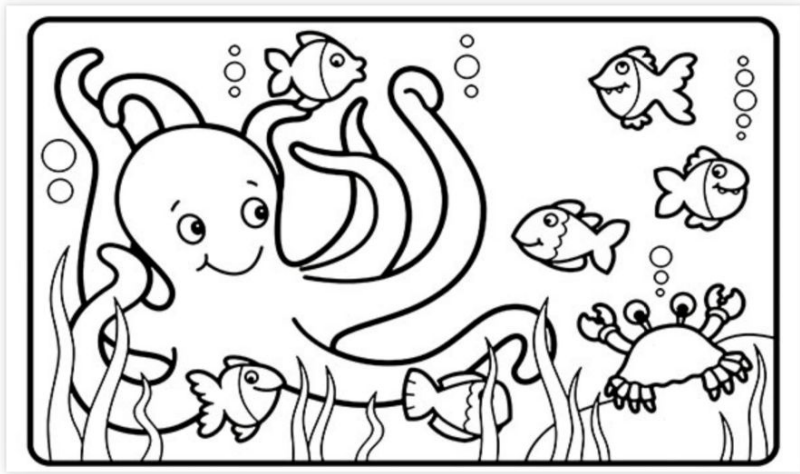
Advantage #1:

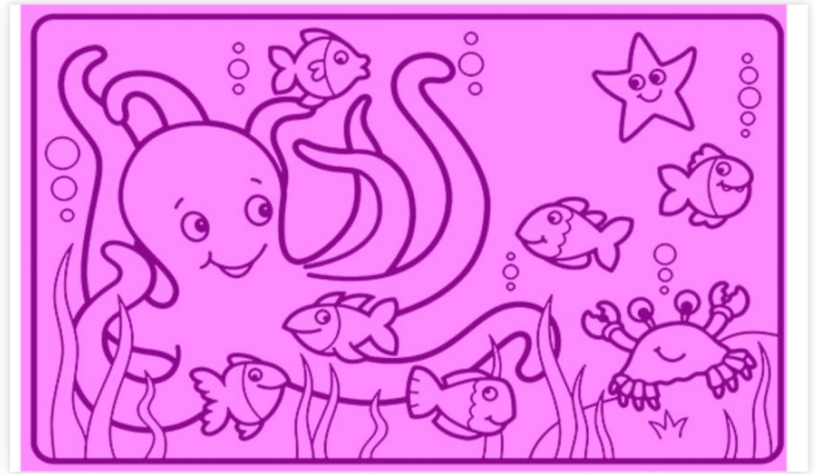
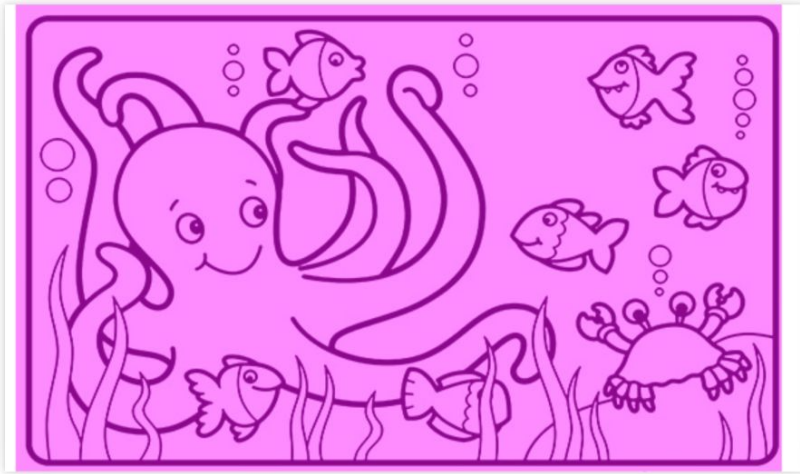
Visual testing covers
everything on a page.

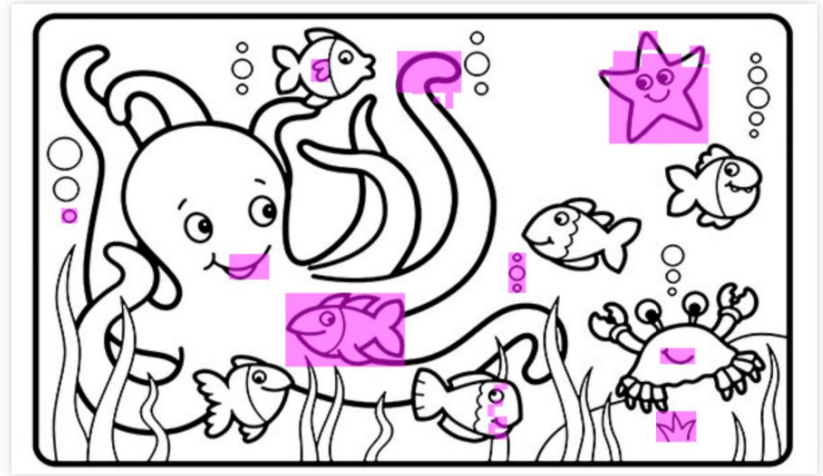
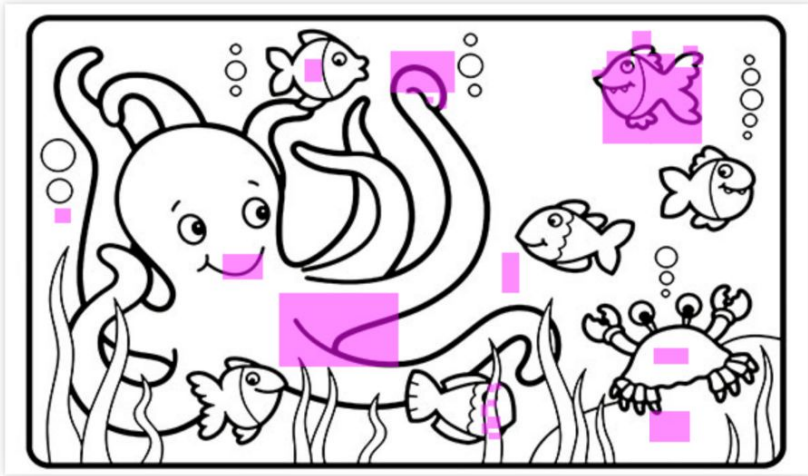


John Henry statue and the Great Bend Tunnel

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





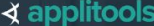


Advantage #2:

Visual AI focuses on
meaningful changes to avoid noise.

Sign up | Applitools

auth.applitools.com/users/register



applitools




VISUALLY PERFECT

Visually test all your apps

- ✓ Find visual bugs across browsers, screen sizes, mobile devices, and operating systems
- ✓ Use AI to eliminate false positives that frustrate your team
- ✓ One call to our API eliminates multiple checkpoints in your test scripts
- ✓ Supports over 40 testing tools: [Selenium](#), [Appium](#), and [many more](#)


Questions?
Talk to a visual testing expert at
+1 650 680 1000

Trusted by hundreds of companies



Create your free account

No obligation. No credit card required.

 **GITHUB SIGNUP**

or

First name Last name


Your API key will be sent here

Company

Country

By creating an account, you agree to our [terms of service](#) and acknowledge your data will be used in accordance with Applitools' [privacy policy](#).

CREATE ACCOUNT



```
<dependency>
  <groupId>com.applitools</groupId>
  <artifactId>eyes-selenium-java3</artifactId>
  <version>3.213.0</version>
  <scope>test</scope>
</dependency>
```

```
private static boolean headless;
private static Configuration config;
private static VisualGridRunner runner;

@BeforeAll
public static void setUpConfigAndRunner() {
    headless = Boolean.parseBoolean(System.getenv().getOrDefault("HEADLESS", "false"));
    runner = new VisualGridRunner(new RunnerOptions().testConcurrency(5));

    config = new Configuration();
    config.setApiKey(System.getenv("APPLITOLS_API_KEY"));
    config.setBatch(new BatchInfo("A Visual Testing Revolution"));
    config.addBrowser(800, 600, BrowserType.CHROME);
}
```

```
@BeforeEach
public void setUpVisualAI (TestInfo testInfo) {

    // Initialize Selenium WebDriver
    driver = new ChromeDriver (new ChromeOptions().setHeadless (headless));

    // Initialize Eyes
    eyes = new Eyes (runner);
    eyes.setConfiguration (config);

    // Open Eyes to start visual testing
    eyes.open (driver, "Applitools Demo App", testInfo.getDisplayName());
}
```

```
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**.

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

How Visual AI Makes Testing a Breeze!

Andrew Knight

Automation Panda
Applitools Developer Advocate
Test Automation University Director

