1

Introducing TAME

TAME builds and runs software tests.

Instead of writing tests one at a time, using TAME, testers fill out simple tables and TAME does the boring and repetitive work of forming combinations to create dozens of useful tests at once.

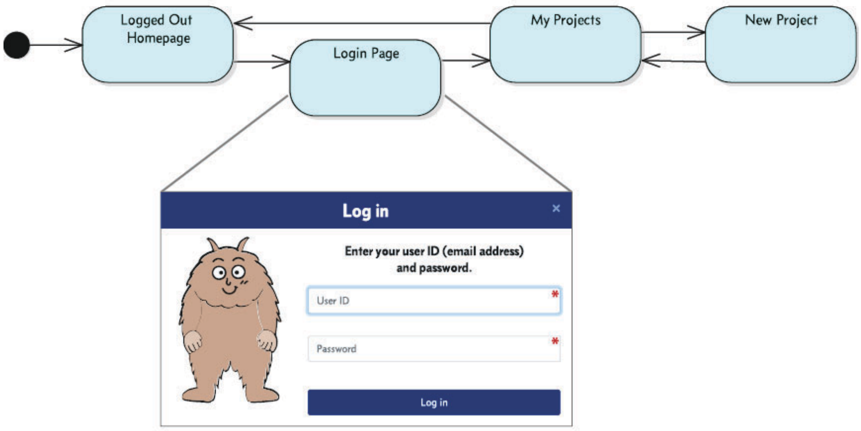
Turn these tests into automated UI tests by simply showing TAME how to locate the UI objects, then TAME runs fully automated tests.

To illustrate how TAME builds and automates software tests, we'll examine a familiar activity—logging into a website. This quick overview will demonstrate core TAME techniques: creating a workbook, identifying inputs and results, defining values, writing directions, and generating test cases. We'll see how TAME can generate scenarios for test planning and review (including the Gherkin language used by Cucumber), how to add automation instructions to a test suite, and how to run the generated tests on multiple platforms.

Choose a Function to Test

While it's common to analyze and design top-down, development and testing usually proceed bottom-up. Systems divide into processes; processes into activities; activities into units—functions, steps, or UI pages.

The whole login activity can be thought of in three steps: a logged out homepage, the login page, and a logged in homepage.



Enter your user ID (email address) and password.

A user starts on the logged out homepage, clicks a link to go to the login page, then a successful login takes the user to the logged in homepage. From there the user can do other activities, such as creating new projects.

Examining the login activity, the login page itself is the most interesting of the three, so we'll start with that page.

Don’t be a Serial Tester

So how many tests would you need to build for a login page? If you’re like many testers and developers, you’ll likely answer that question by creating tests one at a time.

Perhaps you’ll write these as scenarios in a form like this:

```
Scenario: Logged Out Homepage
     Given I am on the logged out homepage
       And there is a user in good standing with ID gary@tametest.net
       And the user gary@tametest.net has password Test.123
      When I click the Log In link in the upper right corner
      Then I am on the Login page

      When I type "gary@tametest.net" into the User ID field
       And I type "Test.123" into the Password field
       And I click the blue Log In button
      Then the login is successful. I'm taken to the logged in homepage

      When I click the Hello link in, then select the Log Out menu item
      Then I am on the logged out homepage
```

```
Scenario: Wrong Password Error
     Given I am on the logged out homepage
       And there is a user in good standing with ID gary@tametest.net
       And the user gary@tametest.net does not have password Bad-456
      When I click the Log In link in the upper right corner
      Then I am on the Login page

      When I type "gary@tametest.net" into the User ID field
       And I type "Bad-456" into the Password field
       And I click the blue Log In button
      Then I am told that I can't log in
```

This Given-When-Then form is a common language known as Gherkin - part of the Cucumber system.[1]

[1] Reference: The Cucumber Book

While Gherkin provides a common, easily understood, human readable form for writing tests, you're still left with having to create each of these tests one by one. If I have a number of scenarios, I'm likely doing a lot of copying and pasting. When it's time to make changes I'll have to go through and inspect those tests and make changes in several different places.

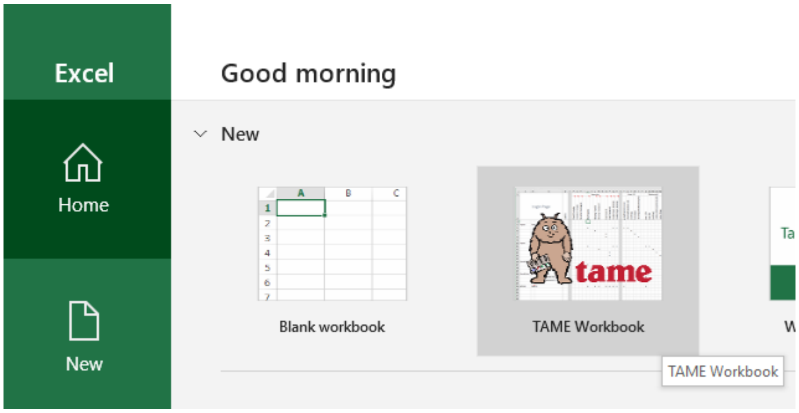
And then there's the problem of turning those into automated executable tests. Often this involves testers (or automation engineers) reading the test scenarios, then recording or writing tests in a programming language.

TAME greatly simplifies the process of building the many test cases needed to completely verify a software system.

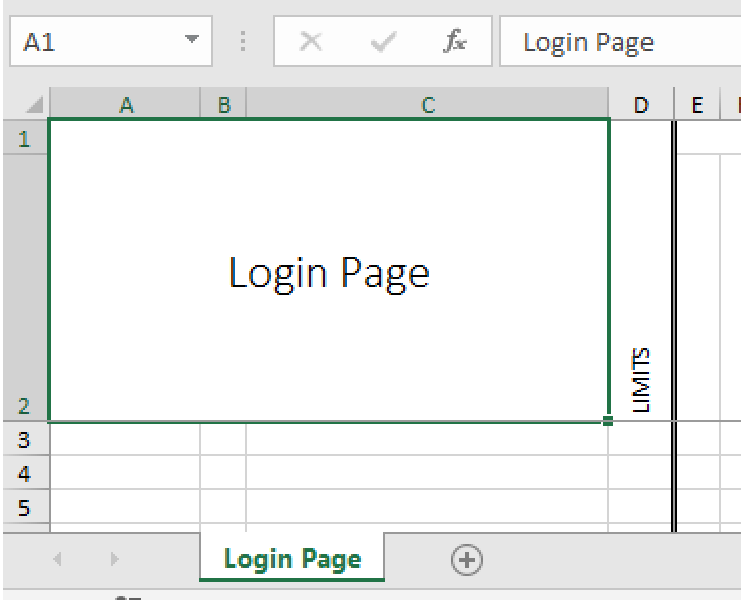
Create a Workbook

To build tests for the login page, create a TAME workbook.

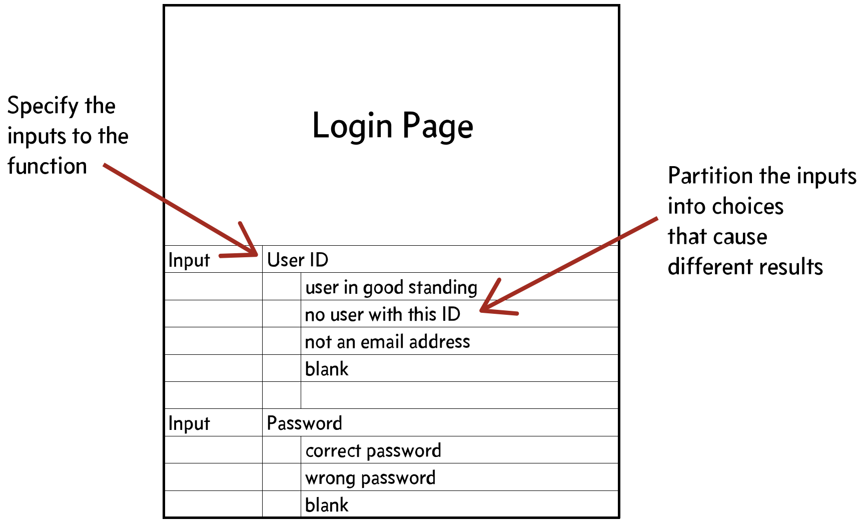
If you have Excel and the TAME plugin, just select the TAME Workbook template.



Name the worksheet by typing “Login Page” into cell A1. Note how this changes the name of the tab as well.



The login page has two inputs, the user ID and the password. Create input categories for each. Then partition the inputs into choices representing different inputs that will produce different behavior.



Specify the inputs to the function

Login Page

Input	User ID	
		user in good standing
		no user with this ID
		not an email address
		blank
Input	Password	
		correct password
		wrong password
		blank

Partition the inputs into choices that cause different results

The other part of building tests is to define the expected results. One result is a successful login. This requires entering a good user ID and a good password before clicking the Login button. Define the result by naming it in the top row under the Checks header, then marking cells for the two input choices and the event.

Entering the user ID of a user who does not exist—entering an email address that doesn't match a registered user —will produce an error regardless of the choice for the password. In this case, a User ID choice is marked, but no choice is selected for the password.

Other error results include:

- a good user ID but the wrong password for the user
- leaving the user ID blank
- leaving the password blank

Each of these are marked with sets of choices that lead to the result.

Finally, create an Event for the Login button—the user behavior that triggers the Login.

List the different expected results

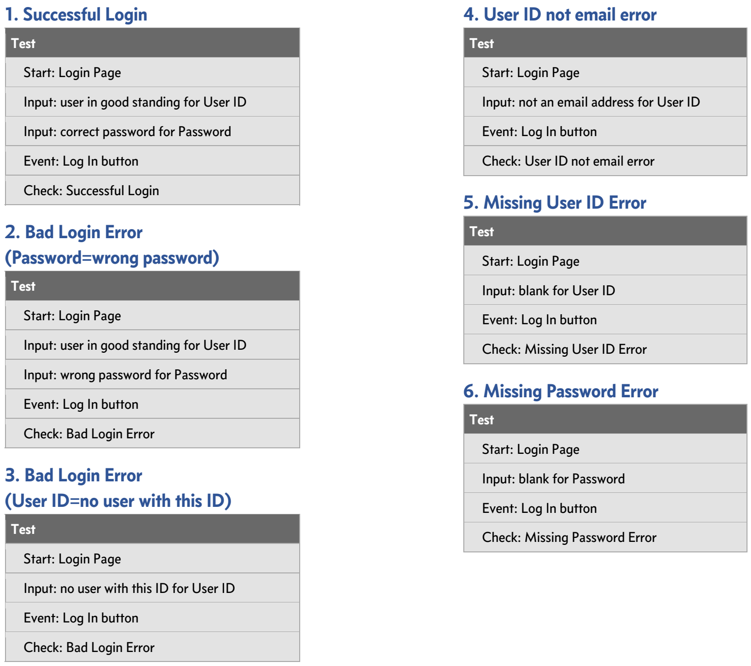
Login Page			LIMITS	Checks									
					Successful Login		Bad Login Error			User ID not email error	Missing User ID Error	Missing Password Error	
Input	User ID												
		user in good standing			x		x						
		no user with this ID						x					
		not an email address								x			
		blank									x		
Input	Password												
		correct password			x								
		wrong password					x						
		blank										x	
Event	Log In button				x		x	x		x	x	x	

"X marks the spot" to define which choices lead to which results

Name the events that trigger the function's behavior

Generate Tests

Save the workbook and click the Generate Tests button. The Test Viewer opens to show six different scenarios.



1. Successful Login

Test
Start: Login Page
Input: user in good standing for User ID
Input: correct password for Password
Event: Log In button
Check: Successful Login

2. Bad Login Error (Password=wrong password)

Test
Start: Login Page
Input: user in good standing for User ID
Input: wrong password for Password
Event: Log In button
Check: Bad Login Error

3. Bad Login Error (User ID=no user with this ID)

Test
Start: Login Page
Input: no user with this ID for User ID
Event: Log In button
Check: Bad Login Error

4. User ID not email error

Test
Start: Login Page
Input: not an email address for User ID
Event: Log In button
Check: User ID not email error

5. Missing User ID Error

Test
Start: Login Page
Input: blank for User ID
Event: Log In button
Check: Missing User ID Error

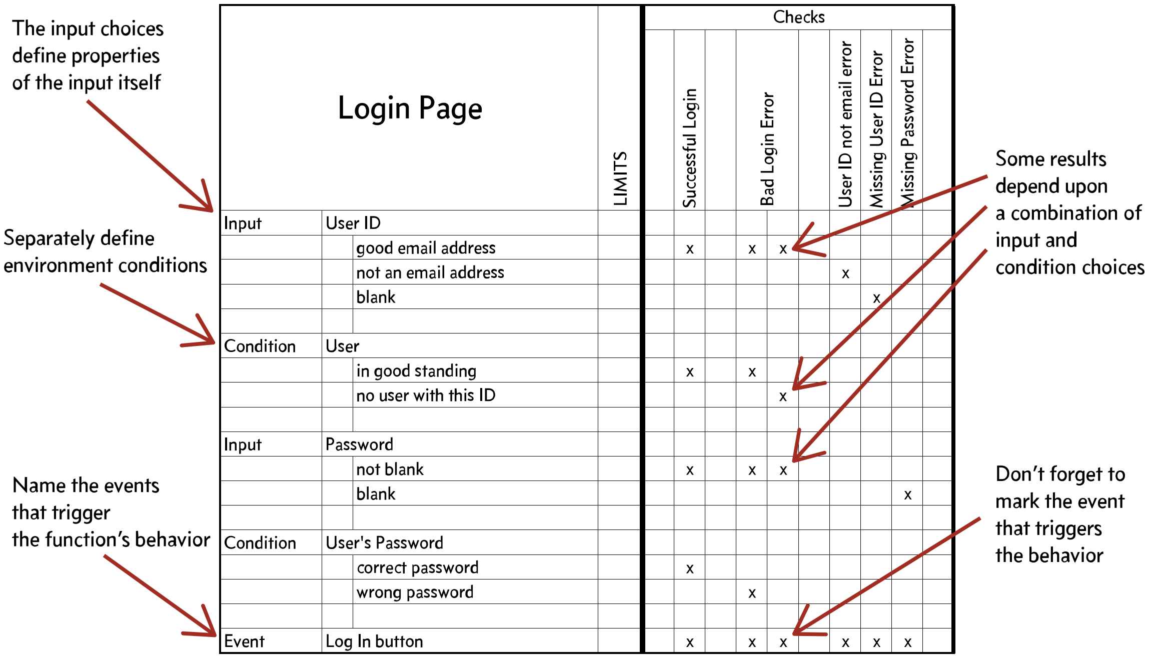
6. Missing Password Error

Test
Start: Login Page
Input: blank for Password
Event: Log In button
Check: Missing Password Error

Conditions

Are there certain properties of the environment or the objects within it that contribute to different outcomes? For example, it's not just enough to have a user ID in the form of an email address: it must also be the user ID of a real known user.

Likewise, the it's not enough to have a nonblank password: it has to be the right password for the user.



Login Page			LIMITS	Checks									
					Successful Login		Bad Login Error			User ID not email error	Missing User ID Error	Missing Password Error	
Input	User ID												
		good email address			x		x	x					
		not an email address								x			
		blank									x		
Condition	User												
		in good standing			x		x						
		no user with this ID						x					
Input	Password												
		not blank			x		x	x					
		blank										x	
Condition	User's Password												
		correct password			x								
		wrong password					x						
Event	Log In button				x		x	x		x	x	x	

The input choices define properties of the input itself

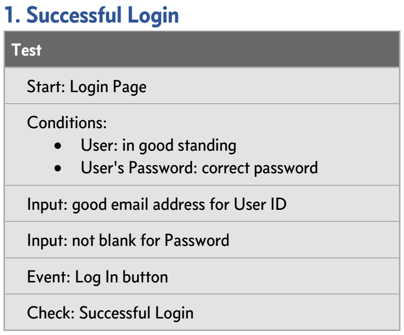
Separately define environment conditions

Name the events that trigger the function's behavior

Some results depend upon a combination of input and condition choices

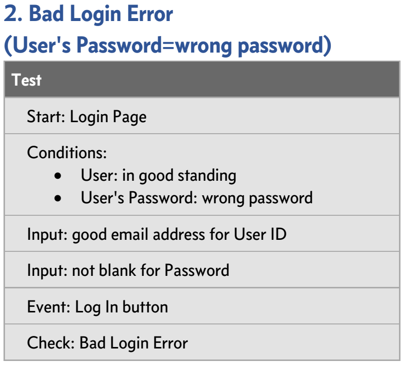
Don't forget to mark the event that triggers the behavior

While we could have just made these into additional choices of the inputs, defining these distinct environment conditions also signals what needs to be true and what needs to be set up before the tests run.



1. Successful Login

Test
Start: Login Page
Conditions: • User: in good standing • User's Password: correct password
Input: good email address for User ID
Input: not blank for Password
Event: Log In button
Check: Successful Login



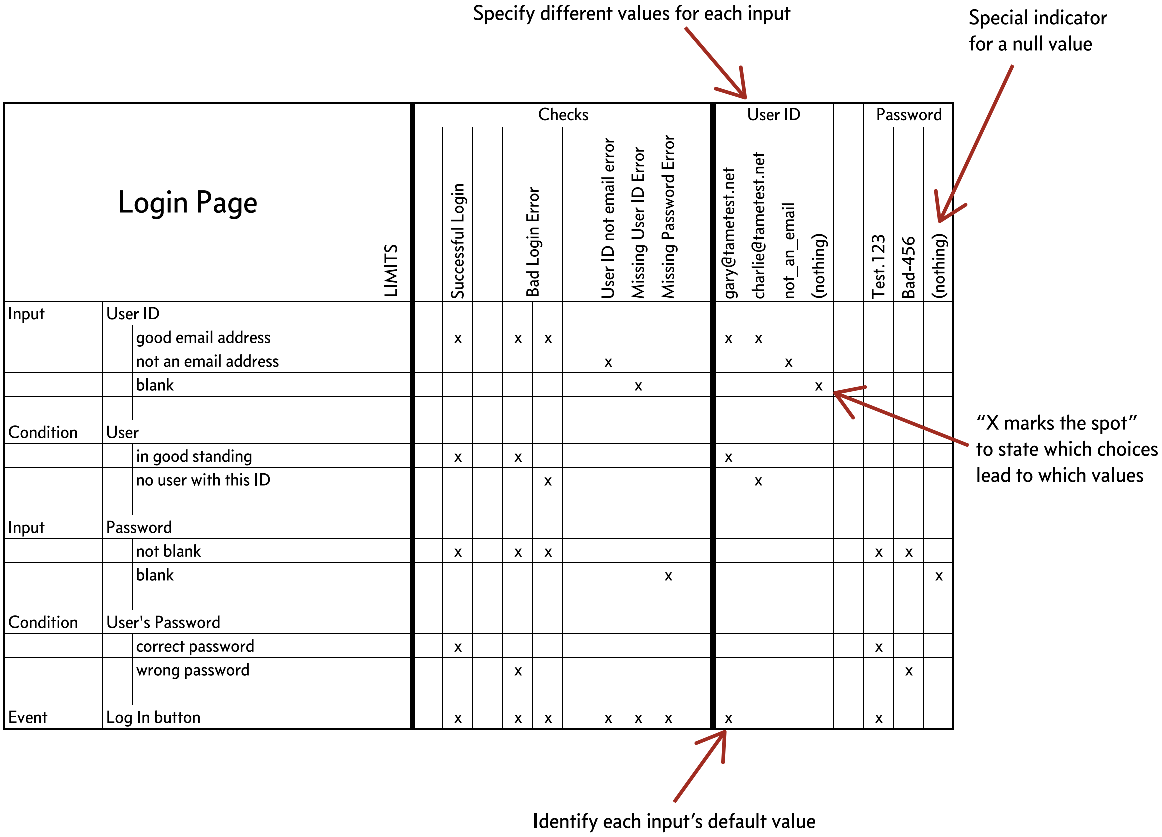
2. Bad Login Error
(User's Password=wrong password)

Test
Start: Login Page
Conditions: • User: in good standing • User's Password: wrong password
Input: good email address for User ID
Input: not blank for Password
Event: Log In button
Check: Bad Login Error

Values

These descriptions are nice, but real tests have real values. To define values, create sets of columns for each input. The column header (on row 1) is the name of the input; values are placed below on row 2.

Values are entered and marked just like results.



Login Page			LIMITS	Checks										User ID					Password		
					Successful Login		Bad Login Error			User ID not email error	Missing User ID Error	Missing Password Error		gary@tametest.net	charlie@tametest.net	not_an_email	(nothing)		Test.123	Bad-456	(nothing)
Input	User ID																				
		good email address			x		x	x						x	x						
		not an email address								x						x					
		blank									x						x				
Condition	User																				
		in good standing			x		x							x							
		no user with this ID						x							x						
Input	Password																				
		not blank			x		x	x											x	x	
		blank										x									x
Condition	User's Password																				
		correct password			x														x		
		wrong password					x													x	
Event	Log In button				x		x	x		x	x	x		x					x		

Specify different values for each input

Special indicator
for a null value

"X marks the spot"
to state which choices
lead to which values

Identify each input's default value

The User ID values are

- a real user with ID gary@tametest.net
- an ID of a user not registered, charlie@tametest.net
- the special keyword "(nothing)" for a blank value. (Note that this keyword is necessary to distinguish a blank value from an empty cell.)

Password values include

- the correct password, Test.123
- a wrong password, Bad.456
- an empty value

Regenerating the tests again (by clicking the Generate Tests button) produces the same six tests, but this time since they contain real values, they should be easier for a tester to follow and replicate consistently.

1. Successful Login

Test
Start: Login Page
Conditions: • User: in good standing • User's Password: correct password
Input: gary@tametest.net into User ID
Input: Test.123 into Password
Event: Log In button
Check: Successful Login

2. Bad Login Error (User's Password=wrong password)

Test
Start: Login Page
Conditions: • User: in good standing • User's Password: wrong password
Input: gary@tametest.net into User ID
Input: Bad-456 into Password
Event: Log In button
Check: Bad Login Error

3. Bad Login Error (User=no user with this ID)

Test
Start: Login Page
Conditions: • User: no user with this ID
Input: charlie@tametest.net into User ID
Input: Test.123 into Password
Event: Log In button
Check: Bad Login Error

4. User ID not email error

Test
Start: Login Page
Input: not_an_email into User ID
Input: Test.123 into Password
Event: Log In button
Check: User ID not email error

5. Missing User ID Error

Test
Start: Login Page
Input: blank for User ID
Input: Test.123 into Password
Event: Log In button
Check: Missing User ID Error

6. Missing Password Error

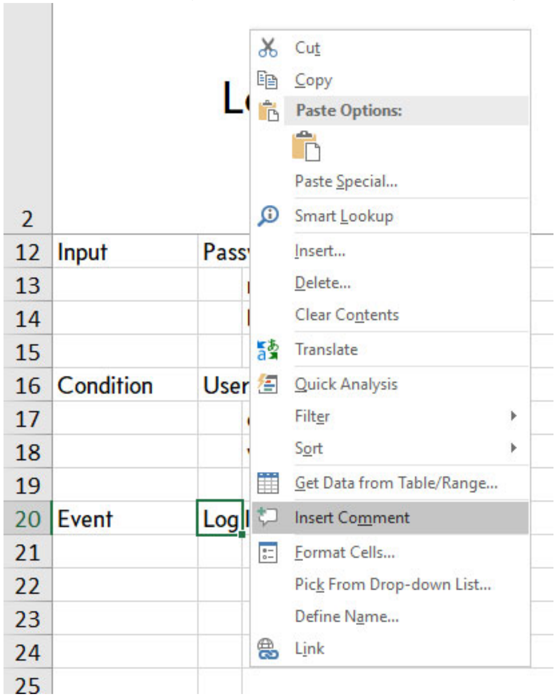
Test
Start: Login Page
Input: gary@tametest.net into User ID
Input: blank for Password
Event: Log In button
Check: Missing Password Error

Note that good tests describe both the kind of value being used (the name of the input choice) and the actual value.

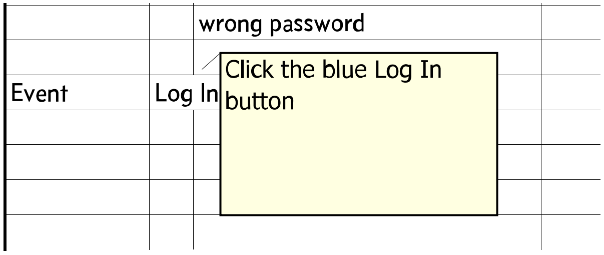
Directions

The scenarios above are better, but they are still a bit stilted. They do not read as smoothly as hand-written tests.

For example, the direction "Event: Log In Button" could be written more descriptively as "Click the blue Log In button."

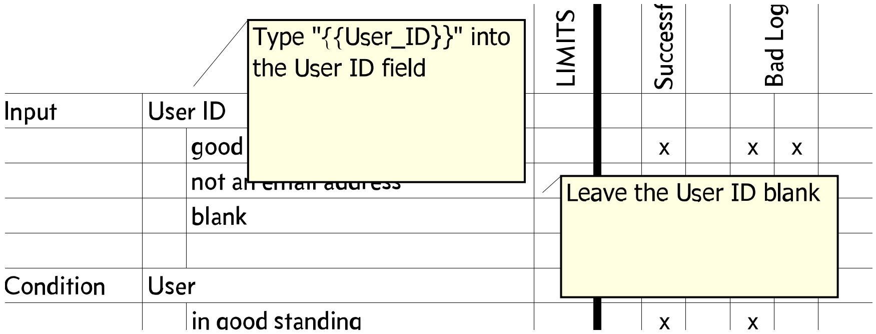


By adding an Excel comment[1] to the cell, this more descriptive direction will be written into the test protocol.



Click the blue Log In button

Cell comments can also include value templates (yes, these are double braces) so that actual values are inserted into the directions.



			LIMITS		Successf		Bad Log		
Input	User ID								
		good			x		x	x	
		not an email address							
		blank							
Condition	User								
		in good standing			x		x		

Type "{{User_ID}}" into the User ID field

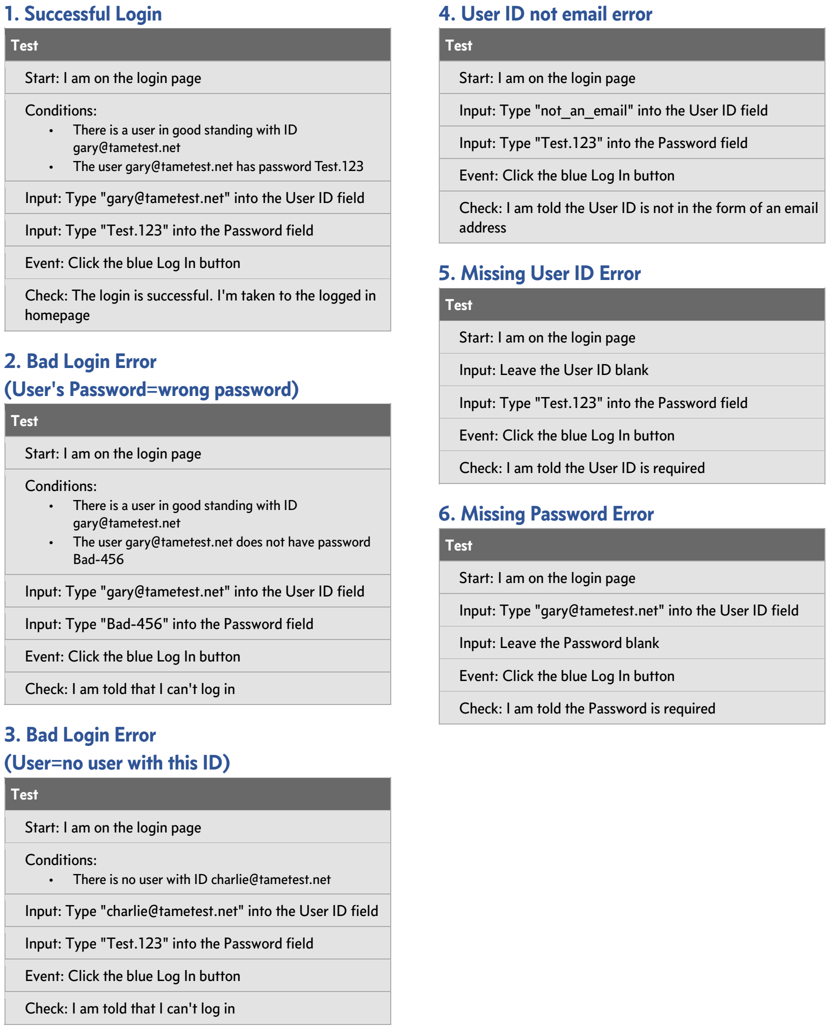
Leave the User ID blank

These templates may also include formatting tokens for adding currency symbols, thousands separators, and decimal points to numbers. Similar to the format specifiers in Java and C#, these can also be used to present date and time values.

Individual input choices may have their own cell comments. For example, instead of "Type '' into the User ID" a better

[1] This is not really the original intent of comment tags in Excel, but we've found this works well for its purpose.

direction would read, “Leave the User ID blank.” Comments can also be used for describing result checks.



1. Successful Login

Test
Start: I am on the login page
Conditions: • There is a user in good standing with ID gary@tametest.net • The user gary@tametest.net has password Test.123
Input: Type "gary@tametest.net" into the User ID field
Input: Type "Test.123" into the Password field
Event: Click the blue Log In button
Check: The login is successful. I'm taken to the logged in homepage

2. Bad Login Error (User's Password=wrong password)

Test
Start: I am on the login page
Conditions: • There is a user in good standing with ID gary@tametest.net • The user gary@tametest.net does not have password Bad-456
Input: Type "gary@tametest.net" into the User ID field
Input: Type "Bad-456" into the Password field
Event: Click the blue Log In button
Check: I am told that I can't log in

3. Bad Login Error (User=no user with this ID)

Test
Start: I am on the login page
Conditions: • There is no user with ID charlie@tametest.net
Input: Type "charlie@tametest.net" into the User ID field
Input: Type "Test.123" into the Password field
Event: Click the blue Log In button
Check: I am told that I can't log in

4. User ID not email error

Test
Start: I am on the login page
Input: Type "not_an_email" into the User ID field
Input: Type "Test.123" into the Password field
Event: Click the blue Log In button
Check: I am told the User ID is not in the form of an email address

5. Missing User ID Error

Test
Start: I am on the login page
Input: Leave the User ID blank
Input: Type "Test.123" into the Password field
Event: Click the blue Log In button
Check: I am told the User ID is required

6. Missing Password Error

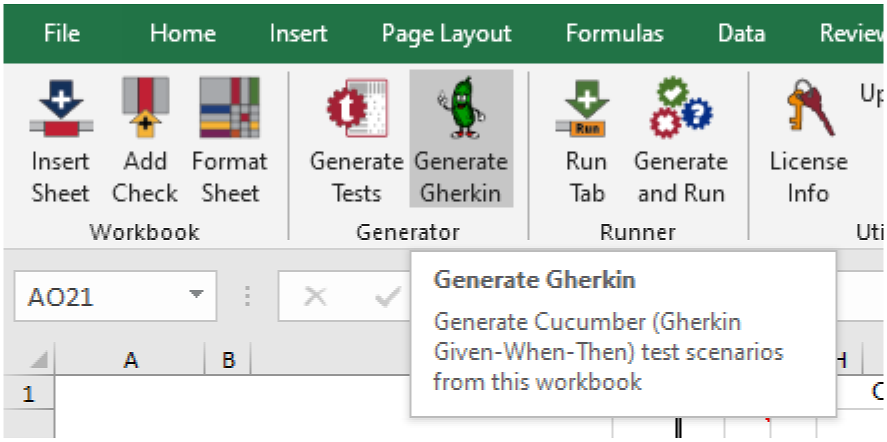
Test
Start: I am on the login page
Input: Type "gary@tametest.net" into the User ID field
Input: Leave the Password blank
Event: Click the blue Log In button
Check: I am told the Password is required

See how the comments make the scenarios much clearer.

Cucumber (Gherkin)

Gherkin is a little language used in the Cucumber system for writing test scenarios. Consisting primarily of sentences written in a Given (precondition) –When (behavior) – Then (postcondition) form, it has become a popular human-readable format for describing scenarios.

Click the Generate Gherkin button to get a Gherkin feature file.



Generate Gherkin

Generate Cucumber (Gherkin Given-When-Then) test scenarios from this workbook

With just this one button click, TAME has generated Cucumber scenarios, ready for review and use by the many tools available for interpreting and generating tests from Cucumber.

```
Feature: Login Demo

    A typical login page used as the initial demonstration of TAME.

  Scenario: Successful Login
      Given I am on the login page
        And there is a user in good standing with ID gary@tametest.net
        And the user gary@tametest.net has password Test.123
       When I type "gary@tametest.net" into the User ID field
        And I type "Test.123" into the Password field
        And I click the blue Log In button
       Then the login is successful. I'm taken to the logged in homepage

  Scenario: Bad Login Error (User's Password=wrong password)
      Given I am on the login page
        And there is a user in good standing with ID gary@tametest.net
        And the user gary@tametest.net does not have password Bad 456
       When I type "gary@tametest.net" into the User ID field
        And I type "Bad 456" into the Password field
        And I click the blue Log In button
       Then I am told that I can't log in

  Scenario: Bad Login Error (User=no user with this ID)
      Given I am on the login page
        And there is no user with ID charlie@tametest.net
       When I type "charlie@tametest.net" into the User ID field
        And I type "Test.123" into the Password field
        And I click the blue Log In button
       Then I am told that I can't log in

  Scenario: User ID not email error
      Given I am on the login page
       When I type "not_an_email" into the User ID field
        And I type "Test.123" into the Password field
        And I click the blue Log In button
       Then I am told the User ID is not in the form of an email address

  Scenario: Missing User ID Error
      Given I am on the login page
       When I leave the User ID blank
        And I type "Test.123" into the Password field
        And I click the blue Log In button
       Then I am told the User ID is required

  Scenario: Missing Password Error
      Given I am on the login page
       When I type "gary@tametest.net" into the User ID field
        And I leave the Password blank
        And I click the blue Log In button
       Then I am told the Password is required
```

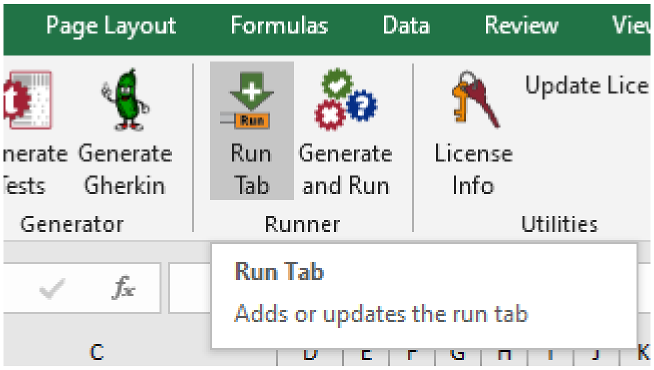
TAME is a great complement to Cucumber. Without TAME, testers need to write each individual scenario. Often this involves a lot of copy-and-paste since the scenarios are mostly identical. Maintaining these scenarios and keeping them consistent in the face of changes can be tough. Because TAME does the boring and repetitive work of forming combinations, there's no copy-and-paste. TAME workbooks follow the "DRY" (Don't Repeat Yourself) principle, thereby making maintenance and updates a snap.

Reviewing Scenarios

Whether presented in TAME format or in Gherkin, the generated scenarios can be reviewed with the whole product team prior to committing to feature development. The team can decide which scenarios are important, which are safely out of scope, and if any additional scenarios are needed. Because TAME generates tests quickly, the workbook can be updated and test plans generated in real-time during these reviews.

Selenium Automation

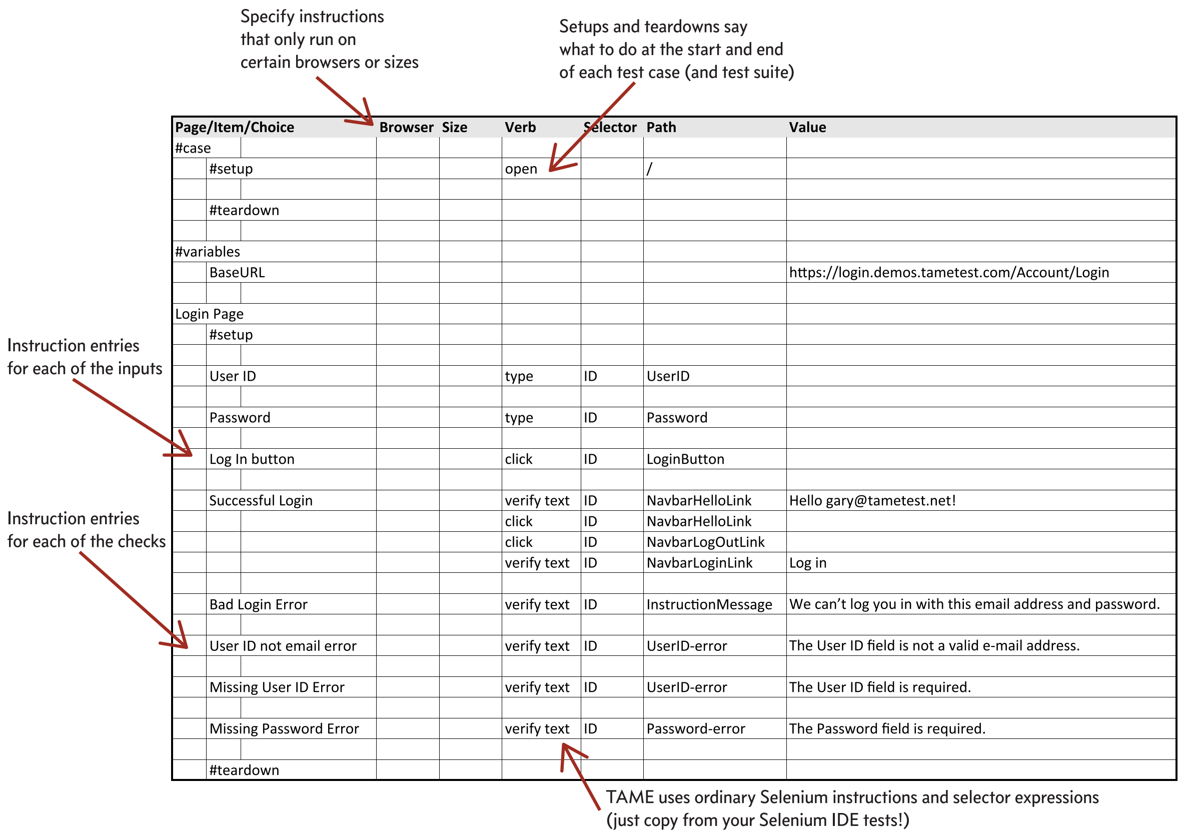
Selenium is a popular technology for automating web browsers. It's easy to use TAME to create automated tests. Just click the button on the Excel ribbon to add a Run tab to the workbook.



Run Tab

Adds or updates the run tab

The Run tab contains places to add the Selenium instructions for each input and result. There are also #setup and #teardown rows for instructions to be run before and after each test case or the whole test suite.



Specify instructions that only run on certain browsers or sizes

Setups and teardowns say what to do at the start and end of each test case (and test suite)

Instruction entries for each of the inputs

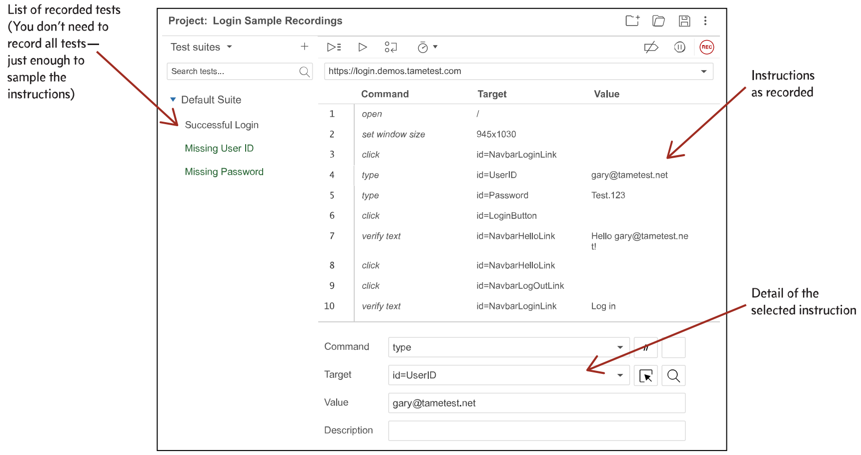
Instruction entries for each of the checks

TAME uses ordinary Selenium instructions and selector expressions (just copy from your Selenium IDE tests!)

Page/Item/Choice		Browser	Size	Verb	Selector	Path	Value
#case							
	#setup			open		/	
	#teardown						
#variables							
	BaseURL						https://login.demos.tametest.com/Account/Login
Login Page							
	#setup						
	User ID			type	ID	UserID	
	Password			type	ID	Password	
	Log In button			click	ID	LoginButton	
	Successful Login			verify text	ID	NavbarHelloLink	Hello gary@tametest.net!
				click	ID	NavbarHelloLink	
				click	ID	NavbarLogOutLink	
				verify text	ID	NavbarLoginLink	Log in
	Bad Login Error			verify text	ID	InstructionMessage	We can’t log you in with this email address and password.
	User ID not email error			verify text	ID	UserID-error	The User ID field is not a valid e-mail address.
	Missing User ID Error			verify text	ID	UserID-error	The User ID field is required.
	Missing Password Error			verify text	ID	Password-error	The Password field is required.
	#teardown						

An easy way to get the instructions to put into the Run tab is to record some actions using Selenium IDE, a free open source add-in for Chrome and Firefox.

Selenium IDE (a Chrome and Firefox plug-in)



List of recorded tests (You don't need to record all tests—just enough to sample the instructions)

Instructions as recorded

Detail of the selected instruction

Project: Login Sample Recordings

https://login.demos.tametest.com

	Command	Target	Value
1	*open*	/	
2	*set window size*	945x1030	
3	*click*	id=NavbarLoginLink	
4	*type*	id=UserID	gary@tametest.net
5	*type*	id=Password	Test.123
6	*click*	id=LoginButton	
7	*verify text*	id=NavbarHelloLink	Hello gary@tametest.net!
8	*click*	id=NavbarHelloLink	
9	*click*	id=NavbarLogOutLink	
10	*verify text*	id=NavbarLoginLink	Log in

Command: type

Target: id=UserID

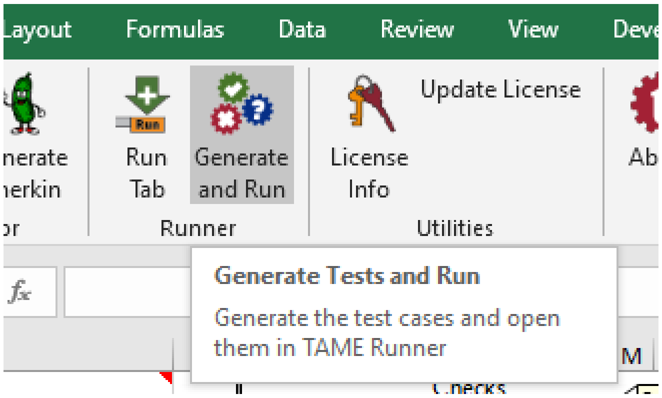
Value: gary@tametest.net

Description:

It's not necessary to record the whole test. Just record the actions necessary to determine the verbs and selector expressions for each action. Then copy the instructions to the Run tab.

One caveat: the TAME login demo has been built to be easy to test. Each of the input fields, buttons, and error messages has a unique HTML ID. Not all web applications will be so easy—but TAME should still be able to automate them.

Once the instructions are in place, click the Generate and Run button. This opens the TAME Runner.



Generate Tests and Run

Generate the test cases and open them in TAME Runner

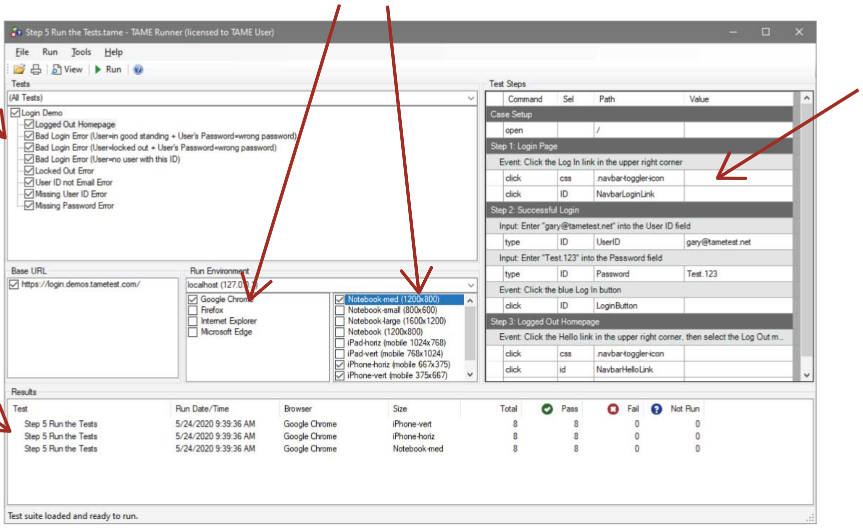
All of the test cases are listed on the left. Select a test case and see its instructions in the right panel.

Select browsers and sources

All generated test cases
Select to run all
or only some tests

Instructions
as written in the
Run tab

Each run is listed
Double-click
to view its log



(All Tests)

- ☑ Login Demo
  - ☑ Logged Out Homepage
  - ☑ Bad Login Error (User=in good standing + User's Password=wrong password)
  - ☑ Bad Login Error (User=locked out + User's Password=wrong password)
  - ☑ Bad Login Error (User=no user with this ID)
  - ☑ Locked Out Error
  - ☑ User ID not Email Error
  - ☑ Missing User ID Error
  - ☑ Missing Password Error

Base URL: ☑ https://login.demos.tametest.com/

Run Environment: localhost (127.0.0.1)

☑ Google Chrome
☐ Firefox
☐ Internet Explorer
☐ Microsoft Edge

☑ Notebook-med (1200x800)
☐ Notebook-small (800x600)
☐ Notebook-large (1600x1200)
☐ Notebook (1200x800)
☐ iPad-horiz (mobile 1024x768)
☐ iPad-vert (mobile 768x1024)
☑ iPhone-horiz (mobile 667x375)
☑ iPhone-vert (mobile 375x667)

Command	Sel	Path	Value
Case Setup			
open		/	
Step 1: Login Page			
Event: Click the Log In link in the upper right corner			
click	css	.navbar-toggler-icon	
click	ID	NavbarLoginLink	
Step 2: Successful Login			
Input: Enter "gary@tametest.net" into the User ID field			
type	ID	UserID	gary@tametest.net
Input: Enter "Test.123" into the Password field			
type	ID	Password	Test.123
Event: Click the blue Log In button			
click	ID	LoginButton	
Step 3: Logged Out Homepage			
Event: Click the Hello link in the upper right corner, then select the Log Out m...			
click	css	.navbar-toggler-icon	
click	id	NavbarHelloLink	

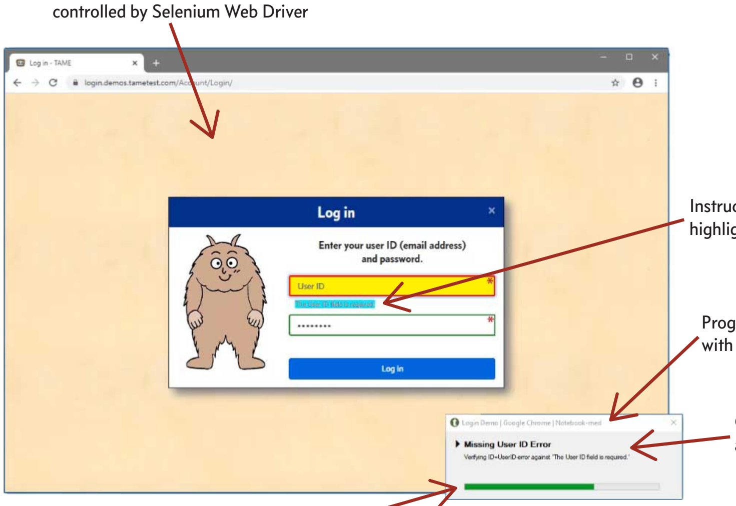
Test	Run Date/Time	Browser	Size	Total	✓ Pass	✗ Fail	? Not Run
Step 5 Run the Tests	5/24/2020 9:39:36 AM	Google Chrome	iPhone-vert	8	8	0	0
Step 5 Run the Tests	5/24/2020 9:39:36 AM	Google Chrome	iPhone-horiz	8	8	0	0
Step 5 Run the Tests	5/24/2020 9:39:36 AM	Google Chrome	Notebook-med	8	8	0	0

Test suite loaded and ready to run.

Choose browsers and sizes for running the tests. TAME can run tests on any installed browser for which a Selenium driver has been installed.

Click Run to watch the tests run. The main Runner window minimizes and is replaced by a small progress dialog in the lower right corner. As the tests run, the target of each instruction is highlighted in blue. These are used in the screen snapshots placed into the test log.

Application running in a browser window
controlled by Selenium Web Driver



Log in - TAME
login.demos.tametest.com/Account/Login/
Log in
User ID
Log in
Login Demo | Google Chrome | Notebook-med

Enter your user ID (email address) and password.

The User ID field is required.

Missing User ID Error

Verifying ID=UserID error against "The User ID field is required."

Instruction target
highlighted in blue

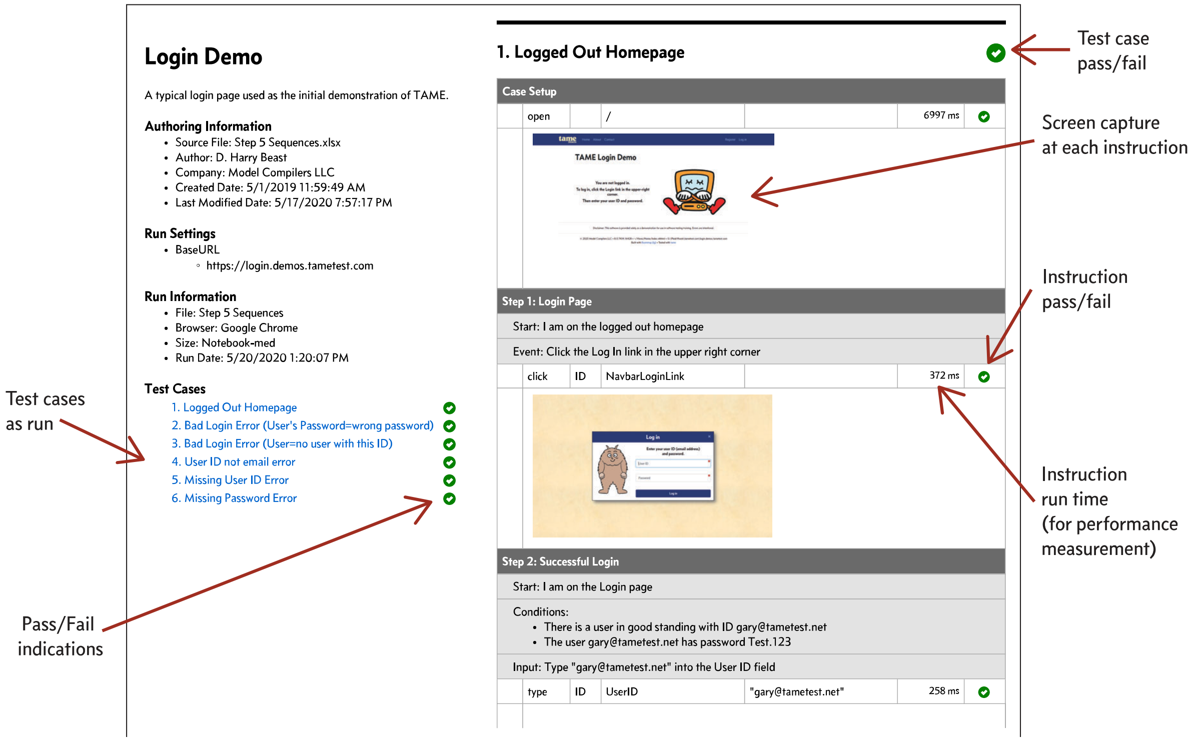
Progress Dialog
with browser and size

Current test
and instruction

Progress bar
Green = success, Red = something failed

When the tests complete, the Runner displays the count of passed, failed, and not-run tests for each different run.

Double-click the result to view the log. Note the summary and pie chart at the top of the log. Each instruction is listed for each test along with a screen snapshot, the time required to run the instruction, and whether the instruction succeeded or failed.



Login Demo

A typical login page used as the initial demonstration of TAME.

Authoring Information

- Source File: Step 5 Sequences.xlsx
- Author: D. Harry Beast
- Company: Model Compilers LLC
- Created Date: 5/1/2019 11:59:49 AM
- Last Modified Date: 5/17/2020 7:57:17 PM

Run Settings

- BaseURL
  - https://login.demos.tametest.com

Run Information

- File: Step 5 Sequences
- Browser: Google Chrome
- Size: Notebook-med
- Run Date: 5/20/2020 1:20:07 PM

Test Cases

1. Logged Out Homepage ✓
2. Bad Login Error (User's Password=wrong password) ✓
3. Bad Login Error (User=no user with this ID) ✓
4. User ID not email error ✓
5. Missing User ID Error ✓
6. Missing Password Error ✓

1. Logged Out Homepage ✓

Case Setup						
	open		/		6997 ms	✓

TAME Login Demo
You are not logged in.
To log in, click the Login link in the upper-right corner.
Then enter your user ID and password.

Step 1: Login Page						
Start: I am on the logged out homepage						
Event: Click the Log In link in the upper right corner						
	click	ID	NavbarLoginLink		372 ms	✓

Step 2: Successful Login						
Start: I am on the Login page						
Conditions: • There is a user in good standing with ID gary@tametest.net • The user gary@tametest.net has password Test.123						
Input: Type "gary@tametest.net" into the User ID field						
	type	ID	UserID	"gary@tametest.net"	258 ms	✓

Test case pass/fail

Screen capture at each instruction

Instruction pass/fail

Instruction run time (for performance measurement)

Test cases as run

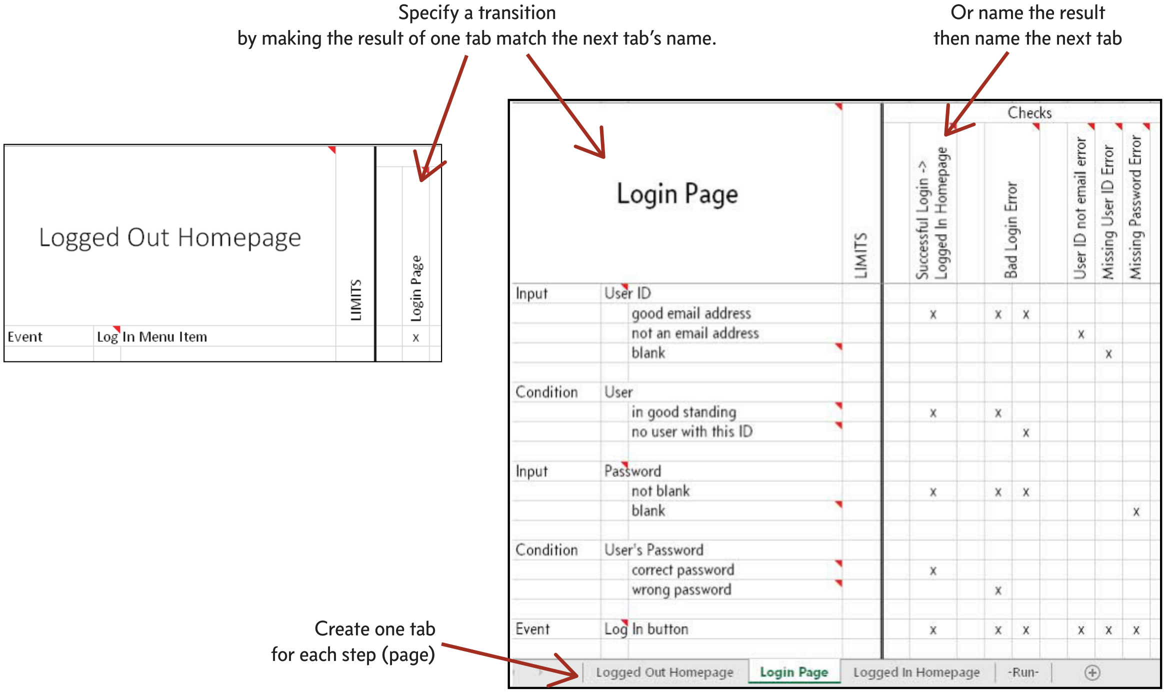
Pass/Fail indications

Combine Steps into Sequences

The individual login page is just one step in the entire login activity. While analysis and design generally proceed top-down, coding and testing are often done bottom-up. This means that we start with individual units, such as functions or single pages, test those units, and then assemble the units into more complex sequences.

Testing with TAME follows this pattern. Now that we have a worksheet for the login page, we can add worksheets for the other steps in the sequence.

Add tabs for the logged out homepage and the logged in homepage. Note that each tab in TAME corresponds to a step in a test. Even if a test goes back to the same UI page more than once, if each step represents a different state in the test (such as “I’m not logged in” vs. “I’m logged in”), representing those different states as separate tabs makes test development much easier.



Specify a transition
by making the result of one tab match the next tab’s name.

Or name the result
then name the next tab

Logged Out Homepage		LIMITS	Login Page
Event	Log In Menu Item		x

				Checks					
Login Page			LIMITS	Successful Login -> Logged In Homepage	Bad Login Error		User ID not email error	Missing User ID Error	Missing Password Error
Input	User ID								
		good email address		x	x	x			
		not an email address					x		
		blank						x	
Condition	User								
		in good standing		x	x				
		no user with this ID				x			
Input	Password								
		not blank		x	x	x			
		blank							x
Condition	User's Password								
		correct password		x					
		wrong password			x				
Event	Log In button			x	x	x	x	x	x

Create one tab
for each step (page)

Connect tabs by making result names match the names of other tabs. For example, the result of clicking the Log In navbar (menu) item on the Logged Out homepage is to go to the Login page.

On the Login page, a successful login goes to the Logged In Homepage. Instead of having to replace the very useful “Successful Login” result name with the destination page, a little arrow in the result cell lets the tester keep the result name and define the next page.

The generated tests now begin at the Logged Out Homepage and proceed to the Login page. Each test is divided into steps.

1. Logged Out Homepage

Case Setup				
	open		/	
Step 1: Login Page				
	Start: I am on the logged out homepage			
	Event: Click the Log In link in the upper right corner			
	click	ID	NavbarLoginLink	
Step 2: Successful Login				
	Start: I am on the Login page			
	Conditions: • There is a user in good standing with ID gary@tametest.net • The user gary@tametest.net has password Test.123			
	Input: Type "gary@tametest.net" into the User ID field			
	type	ID	UserID	"gary@tametest.net"
	Input: Type "Test.123" into the Password field			
	type	ID	Password	"Test.123"
	Event: Click the blue Log In button			
	click	ID	LoginButton	
Step 3: Logged Out Homepage				
	Start: I am on the logged in homepage			
	Event: Click the Hello link in the upper right corner, then select the Log Out menu item			
	click	ID	NavbarHelloLink	
	click	ID	NavbarLogOutLink	

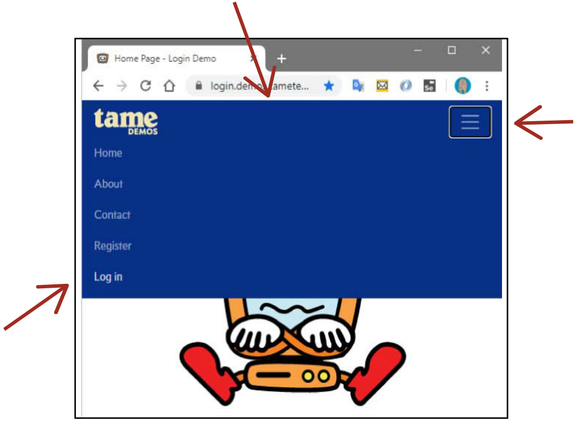
Updating Automation

To add the additional pages and their elements to the Run tab, simply click the Run Tab button. The current elements are left in place, but new rows are added for the inputs and checks in the other two pages. Add the Selenium instructions, and you're ready to run complete test sequences.

Platform-Specific Instructions

If you run the Login demo on a small web browser such as the ones on a smartphone, you'll notice that the navbar is replaced with the iconic three-line "hamburger menu." To get to the Login button, a user has to first click the hamburger menu then click the Login button.

On a narrow browser (e.g. a smartphone)
the navbar is replaced by a "hamburger menu" and dropdown



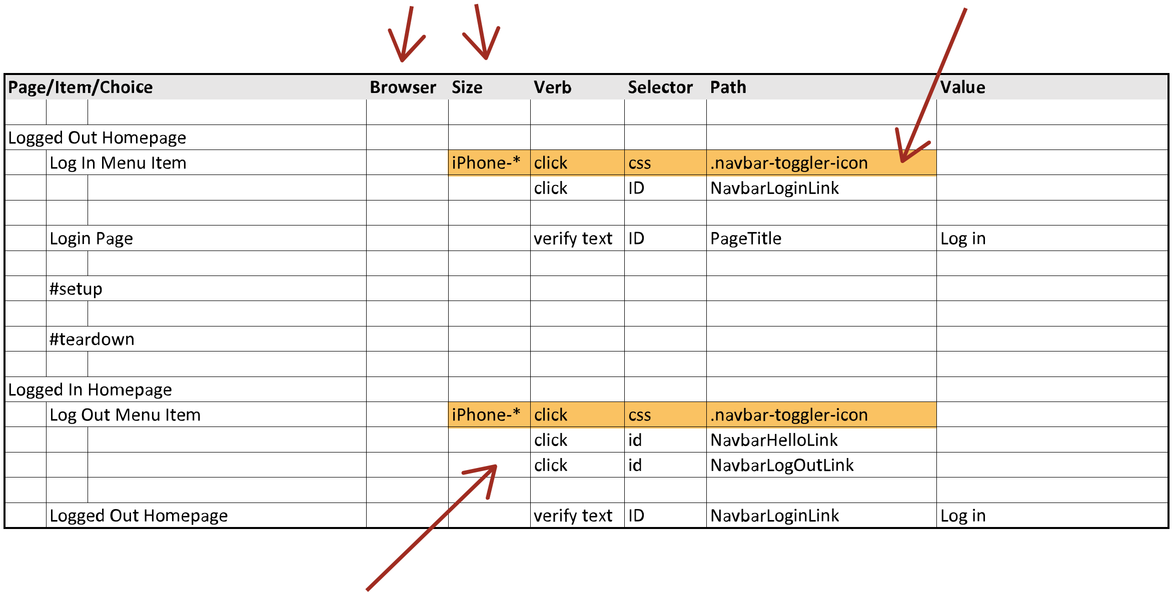
First click the
hamburger menu

Then click Log In

Adding this kind of conditional logic in most automation tools has required either creating two separate suites of test cases (which then have to be separately maintained) or adding conditional logic in code.

Columns for denoting platform-specific instructions

Open the hamburger menu on small devices



Page/Item/Choice			Browser	Size	Verb	Selector	Path	Value
Logged Out Homepage								
	Log In Menu Item			iPhone-*	click	css	.navbar-toggler-icon	
					click	ID	NavbarLoginLink	
	Login Page				verify text	ID	PageTitle	Log in
	#setup							
	#teardown							
Logged In Homepage								
	Log Out Menu Item			iPhone-*	click	css	.navbar-toggler-icon	
					click	id	NavbarHelloLink	
					click	id	NavbarLogOutLink	
	Logged Out Homepage				verify text	ID	NavbarLoginLink	Log in

Unmarked instructions will run on on all devices

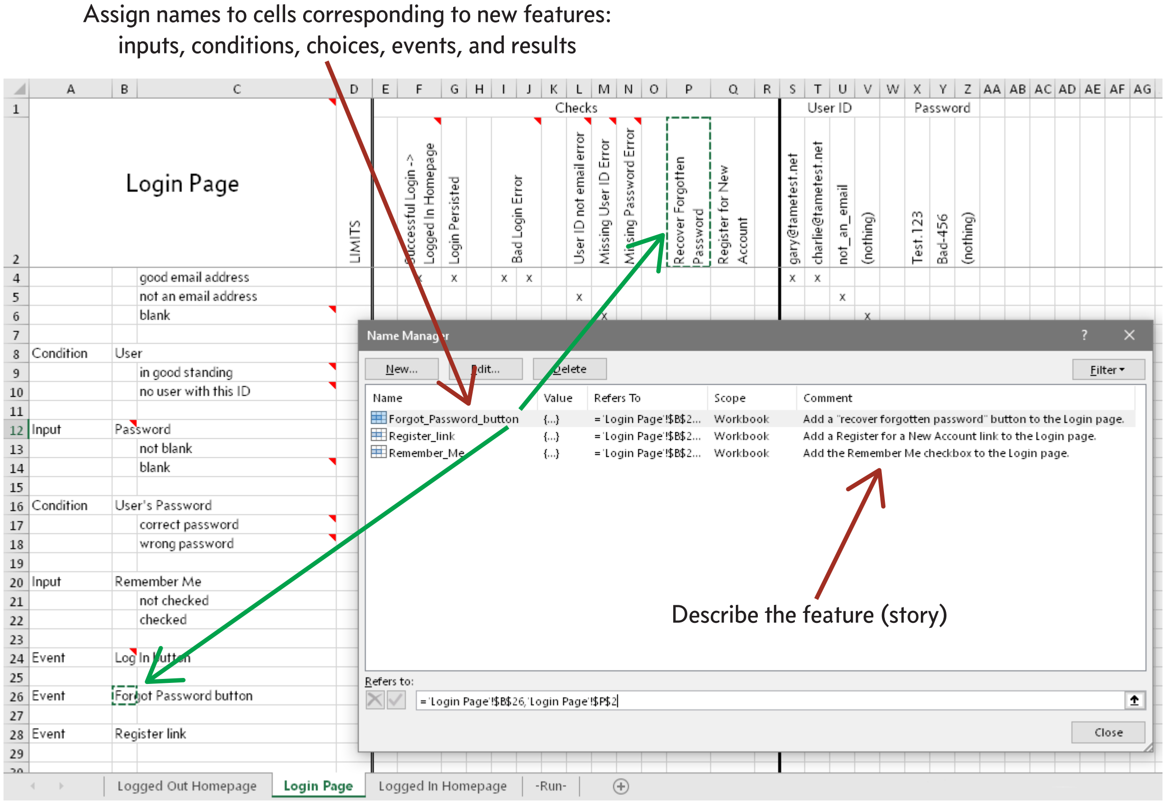
TAME makes this common use case easy. Just add the instructions for clicking the hamburger menu to the Run tab and mark the instructions with the size(s) that require this instruction.

Now when the tests run on a regular large browser, the click is not run. In the log file, the instruction is marked with a strikethrough to indicate that it was skipped. But on a narrow smartphone browser, the instruction is run. Similar methods exist for including instructions to be run on specific browsers or platforms.

Grow with New Features

TAME has been built with agile incremental development in mind. Not only is it easy to review scenarios before committing to stories during iteration planning, it is also easy to add elements and to tag them as related to particular user stories.

For example, to add new controls to the Login page—a Register link, a Remember Me checkbox, and a Forgot Password button—tag the new elements with Excel names.



Assign names to cells corresponding to new features: inputs, conditions, choices, events, and results

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	Login Page				Checks														User ID					Password		
2				LIMITS		Successful Login -> Logged In Homepage	Login Persisted			Bad Login Error		User ID not email error	Missing User ID Error	Missing Password Error		Recover Forgotten Password	Register for New Account		gary@tametest.net	charlie@tametest.net	not_an_email	(nothing)		Test.123	Bad-456	(nothing)
4			good email address			x	x		x	x									x	x						
5			not an email address									x									x					
6			blank										x									x				

	A	B	C
7			
8	Condition	User	
9			in good standing
10			no user with this ID
11			
12	Input	Password	
13			not blank
14			blank
15			
16	Condition	User's Password	
17			correct password
18			wrong password
19			
20	Input	Remember Me	
21			not checked
22			checked
23			
24	Event	Log In button	
25			
26	Event	Forgot Password button	
27			
28	Event	Register link	
29			

Name	Value	Refers To	Scope	Comment
Forgot_Password_button	{...}	='Login Page'!$B$2...	Workbook	Add a "recover forgotten password" button to the Login page.
Register_link	{...}	='Login Page'!$B$2...	Workbook	Add a Register for a New Account link to the Login page.
Remember_Me	{...}	='Login Page'!$B$2...	Workbook	Add the Remember Me checkbox to the Login page.

Refers to: ='Login Page'!$B$26,'Login Page'!$P$2

Describe the feature (story)

The test protocol now identifies the test cases affected by each of these stories.

Test Cases

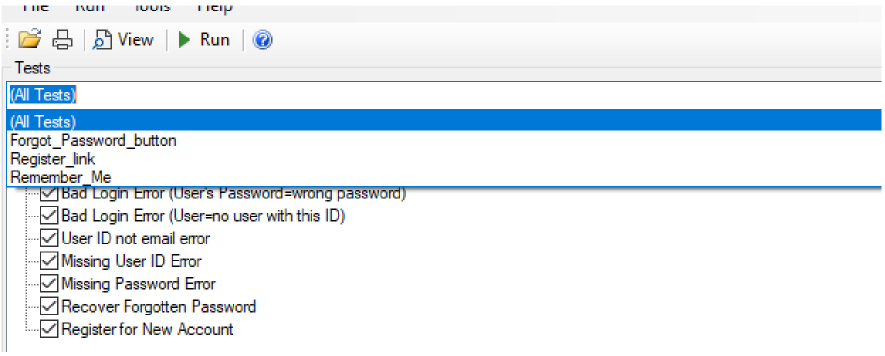
1. Remember Me=not checked
2. Remember Me=checked
3. Bad Login Error (User's Password=wrong password)
4. Bad Login Error (User=no user with this ID)
5. User ID not email error
6. Missing User ID Error
7. Missing Password Error
8. Recover Forgotten Password
9. Register for New Account

Features

- Add a "recover forgotten password" button to the Login page.
  - 8. Recover Forgotten Password
- Add a Register for a New Account link to the Login page.
  - 9. Register for New Account
- Add the Remember Me checkbox to the Login page.
  - 1. Remember Me=not checked
  - 2. Remember Me=checked

These tests were added or changed by the named stories

Moreover, the Runner lets you choose to run all tests or just the tests for selected stories.



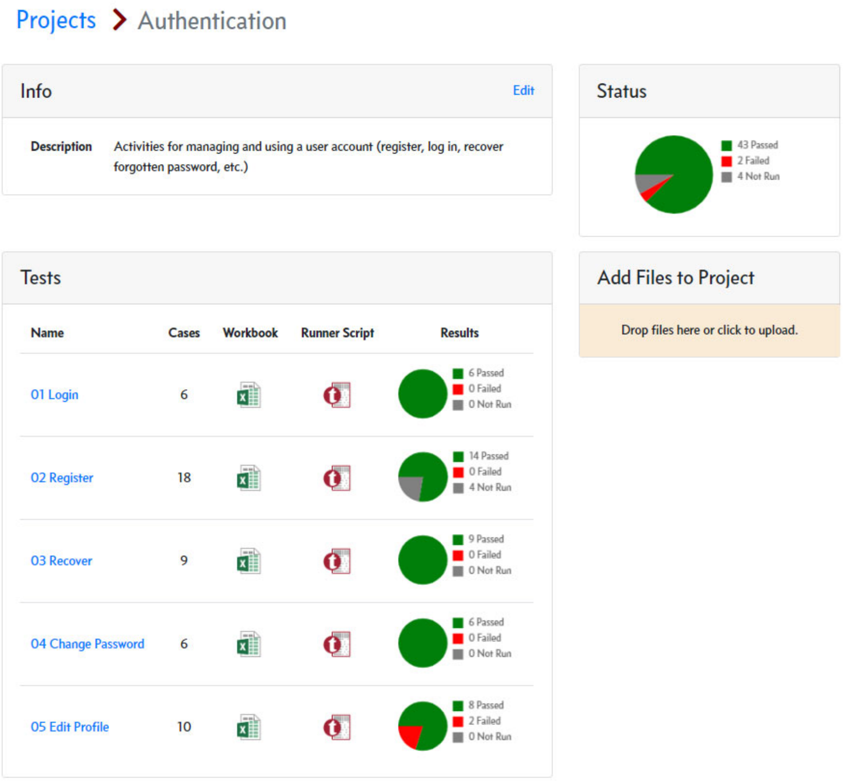
(All Tests)

(All Tests)
Forgot_Password_button
Register_link
Remember_Me

☑ Bad Login Error (User's Password=wrong password)
☑ Bad Login Error (User=no user with this ID)
☑ User ID not email error
☑ Missing User ID Error
☑ Missing Password Error
☑ Recover Forgotten Password
☑ Register for New Account

Share Results with your Team

Every active TAME user can create online workspaces. Upload workbooks and results to share those results with others in your team.



Projects > Authentication

Info Edit

Description Activities for managing and using a user account (register, log in, recover forgotten password, etc.)

Status

🟩 43 Passed
🟥 2 Failed
▨ 4 Not Run

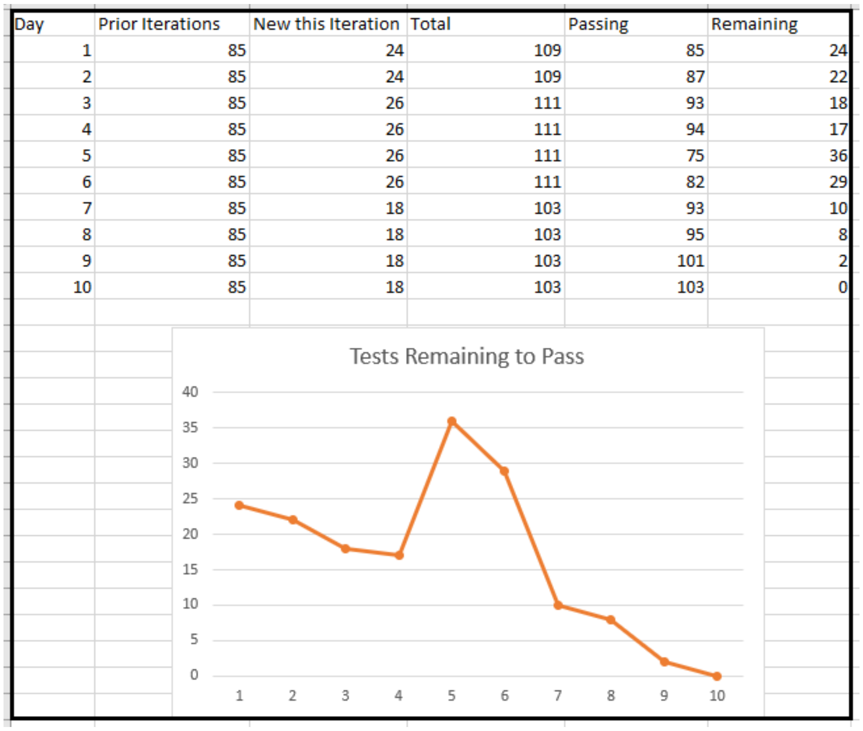
Tests

Name	Cases	Workbook	Runner Script	Results
01 Login	6			🟩 6 Passed 🟥 0 Failed ▨ 0 Not Run
02 Register	18			🟩 14 Passed 🟥 0 Failed ▨ 4 Not Run
03 Recover	9			🟩 9 Passed 🟥 0 Failed ▨ 0 Not Run
04 Change Password	6			🟩 6 Passed 🟥 0 Failed ▨ 0 Not Run
05 Edit Profile	10			🟩 8 Passed 🟥 2 Failed ▨ 0 Not Run

Add Files to Project

Drop files here or click to upload.

Over time as you upload results you can track progress toward completion as a number of tests remaining to pass.



Day	Prior Iterations	New this Iteration	Total	Passing	Remaining
1	85	24	109	85	24
2	85	24	109	87	22
3	85	26	111	93	18
4	85	26	111	94	17
5	85	26	111	75	36
6	85	26	111	82	29
7	85	18	103	93	10
8	85	18	103	95	8
9	85	18	103	101	2
10	85	18	103	103	0

"But wait! There's more!"

This quick overview only covered some of TAME's many features. The following chapters provide more detail on how to build test suites and to automate those tests.

- Inputs and Results
- Conditions
- Values
- Properties
- Sequences and State Models
- Automation Instructions
- Running Automated Tests