

## User Guides

<b>Test Site</b>
QA: <a href="https://test-site.foodtruck-qa.com">https://test-site.foodtruck-qa.com</a>
UAT: <a href="https://test-site.foodtruck-uat.com">https://test-site.foodtruck-uat.com</a>
<b>Test Report</b>
QA: <a href="https://test-report.foodtruck-qa.com">https://test-report.foodtruck-qa.com</a>
UAT: <a href="https://test-report.foodtruck-uat.com">https://test-report.foodtruck-uat.com</a>

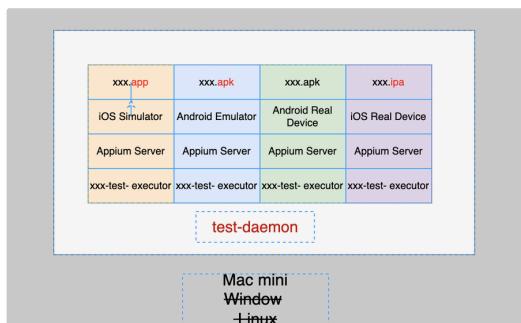
- **Test Automation Platform**
  - [How to deploy test daemon on mac mini](#)
    - Environmental Requirements
    - Build test daemon on development environment
    - Deploy test daemon on mac mini
    - Real device parallel testing
    - Virtual device parallel testing
  - [How to Register Project on Test site?](#)
    - Step 1. Creating executor repo on github
    - Step 2: Creating a personal access token from gitgub
    - Step 3: Register project on test site?
  - [How to make plan?](#)
  - [How to generate UI Difference Report?](#)
  - [How to Generate Allure Report?](#)

## Test Automation Platform

Step	location
Prepare test-executor	<ul style="list-style-type: none"> <li>• <a href="#">https://github.com/food-truck/consumer-app-test-executor</a> Connect your Github account</li> <li>• <a href="#">https://github.com/food-truck/courier-app-test-executor</a> Connect your Github account</li> <li>• <a href="#">https://github.com/food-truck/consumer-web-test-executor</a> Connect your Github account</li> </ul>
Prepare application package	<ul style="list-style-type: none"> <li>• app regression test: <a href="http://192.168.11.38">http://192.168.11.38</a></li> <li>• mobile web test: Chrome Or Safari</li> </ul>
Deploy test-daemon	<ul style="list-style-type: none"> <li>• mac mini m2: 192.168.33.2 (qa2023)</li> <li>• mac mini i7: 192.168.33.5 (qa2018)</li> </ul>
Register project	<ul style="list-style-type: none"> <li>QA: <a href="https://test-site.foodtruck-qa.com">https://test-site.foodtruck-qa.com</a></li> <li>UAT: <a href="https://test-site.foodtruck-uat.com">https://test-site.foodtruck-uat.com</a></li> </ul>
Make plan	<ul style="list-style-type: none"> <li>QA: <a href="https://test-site.foodtruck-qa.com">https://test-site.foodtruck-qa.com</a></li> <li>UAT: <a href="https://test-site.foodtruck-uat.com">https://test-site.foodtruck-uat.com</a></li> </ul>
Monitor device	<ul style="list-style-type: none"> <li>QA: <a href="https://test-site.foodtruck-qa.com">https://test-site.foodtruck-qa.com</a></li> <li>UAT: <a href="https://test-site.foodtruck-uat.com">https://test-site.foodtruck-uat.com</a></li> </ul>
View Report	<ul style="list-style-type: none"> <li>QA: <a href="https://test-report.foodtruck-qa.com">https://test-report.foodtruck-qa.com</a></li> <li>UAT: <a href="https://test-report.foodtruck-uat.com">https://test-report.foodtruck-uat.com</a></li> </ul>

### How to deploy test daemon on mac mini

#### How to deploy test daemon on mac mini



## Environmental Requirements

Tool	how to install ?	Env variable configuration	remark
homebrew	<p>URL: <a href="#">Installation</a></p> <pre>1 #check install 2 brew -v</pre>		required install
xcode	<pre>1 xcode-select --install 2 xcrun --version</pre>		Required install to use iPhone devices

libimobiledevice	<pre> 1 # via homebrew 2 brew install libimobiledevice 3 4 # or via port [port install:https://www.macports.org/port/libimobiledevice/] 5 sudo port install libimobiledevice 6 7 # check install 8 iDeviceInfo -v </pre>		Required install to use iPhone real devices
android studio	download from <a href="#">Download Android Studio &amp; App Tools - Android Developers</a>	<pre> 1 # modify ~/.bash_profile, and add the following lines: 2 echo 'export ANDROID_HOME=~/Library/AndroidStudio3.2' 3 echo 'export PATH=\$PATH:\$ANDROID_HOME/tools:\$ANDROID_HOME/platform-tools' 4 source ~/.bash_profile 5 #check install: adb 6 adb version 7 #check install: emulator 8 emulator -version </pre>	Required install to use android devices
android virtual device: google	<a href="#">Create and manage virtual devices   Android Studio   Android Developers</a>	<p>Modify the key-value pair hw.device.manufacturer &amp; hw.device.model in the <code>~/.android/avd/&lt;avd_id&gt;.avd/config.ini</code> file</p> <pre> 1 # Modify the key-value pair hw.device.manufacturer 2 # demo: 3 # avd_id=Samsung_Galaxy_S21 4 # hw.device.manufacturer=Samsung 5 # hw.device.model=Galaxy S21 6 cd \$(cat ~/.android/avd/&lt;avd_id&gt;.ini   grep hw.device.manufacturer   sed -i '' '/hw.device.manufacturer/d') 7 echo "hw.device.manufacturer=&lt;manufacturer&gt;" &gt;&gt; config.ini 8 sed -i '' '/hw.device.model/d' config.ini 9 echo "hw.device.model=&lt;model&gt;" &gt;&gt; config.ini 10 11 12 # how to get avd_id 13 emulator -list-avds </pre>	Required install to use android virtual devices
android virtual device : samsung	<a href="#">how to create android virtual device for samsung?</a>	<pre> 1 # Modify the key-value pair hw.device.manufacturer 2 # demo: 3 # avd_id=Samsung_Galaxy_S21 4 # hw.device.manufacturer=Samsung 5 # hw.device.model=Galaxy S21 6 cd \$(cat ~/.android/avd/&lt;avd_id&gt;.ini   grep hw.device.manufacturer   sed -i '' '/hw.device.manufacturer/d') 7 echo "hw.device.manufacturer=&lt;manufacturer&gt;" &gt;&gt; config.ini 8 sed -i '' '/hw.device.model/d' config.ini 9 echo "hw.device.model=&lt;model&gt;" &gt;&gt; config.ini 10 11 12 # how to get avd_id 13 emulator -list-avds </pre>	Required install to use samsung virtual devices
appium	<ul style="list-style-type: none"> <li>• appium1</li> </ul> <pre> 1 brew install node 2 #check install 3 node --version 4 5 npm install -g appium 6 #check install 7 appium --version </pre> <ul style="list-style-type: none"> <li>• appium2</li> </ul> <pre> 1 brew install node 2 node --version 3 npm install -g appium@next 4 appium --version 5 // Installing drivers 6 appium driver install uiautomator2 7 appium driver install xcuitest 8 // npm install --global appium --drivers=xcuitest,ui </pre>		Required install
iOS screen recording plugin	<pre> 1 # install or sudo port install ffmpeg 2 brew install ffmpeg 3 # check install 4 ffmpeg -version </pre>		Required install to use iPhone devices
JDK17	<a href="#">Java Downloads   Oracle 中国</a>	<pre> 1 export JAVA_HOME=/Library/Java/JavaVirtualMachines/jdk-17.0.1.jdk/Contents/Home 2 export CLASSPATH=\$JAVA_HOME/lib/tools.jar 3 export PATH=\$PATH:\$JAVA_HOME/bin </pre>	Required install to run test-daemon & test-executor

## Build test daemon on development environment

- download test-daemon from Github Repo: <https://github.com/food-truck/foodtruck-test-automation> Connect your Github account
- on rootProject , run `./gradlew installDist -p test-daemon -Penv={env}`

```
1 #dev
2 ./gradlew installDist -p test-daemon -Penv=dev
3 #uat
4 ./gradlew installDist -p test-daemon -Penv=uat
```

- Navigate to a directory: {rootProject}/build/test-daemon/install, compressed into file `test-daemon.zip`

## Deploy test daemon on mac mini

- decompress `test-daemon.zip`
- run command: `nohup sh {} max_number_of_devices={} app_name={} 2>&1 &`

```
1 #dev
2 nohup sh ~/Documents/dev/test-daemon/bin/test-daemonmax_number_of_devices=3 app_name=WONDER > ~/Documents/dev/log/2024.04.01.log 2>&1 &
3
4 tail -f ~/Documents/dev/log/2024.04.01.log
```

- kill test-daemon

```
1 #get pid
2 jps
3 #kill process
4 kill {pid}
```

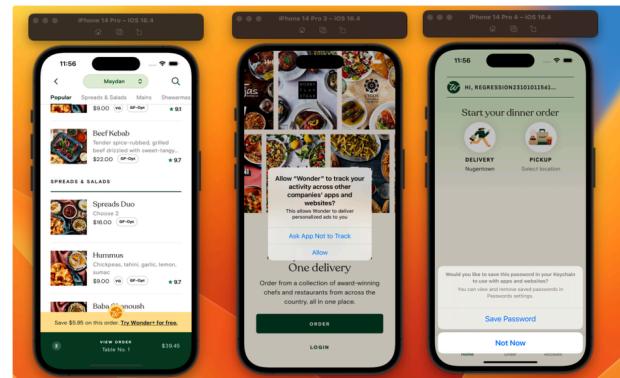
## Real device parallel testing

- iPhone real device
- Android real device



## Virtual device parallel testing

- iPhone simulator

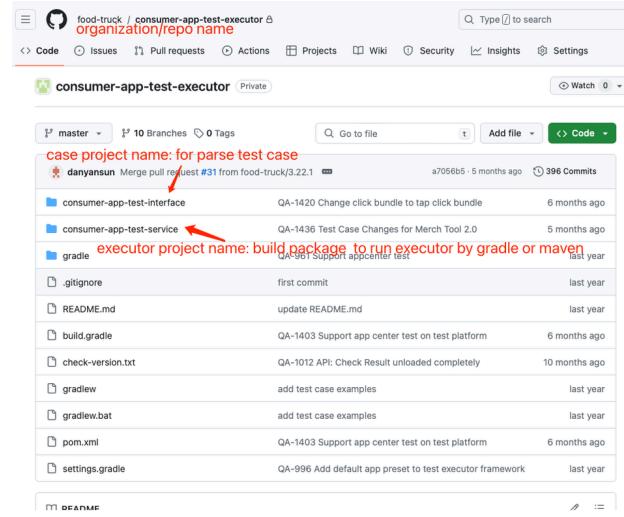


## How to Register Project on Test site?

### Step 1. Creating executor repo on github

- Naming format

	format	example	Remark
Github Organization	food-truck	food-truck	-
Github Repo Name	xxx-test-executor	consumer-app-test-executor . consumer-web-test-executor	-
Executor Project Name	xxx-test-service	consumer-app-test-service	build package to run executor by gradle or maven
Case Project Name	xxx-test-interface	consumer-app-test-interface	parse test cases
Case Package Name	xxx.xxx.xxx	app.consumerappservice.wonder	The package directory where the test case is located
branch	number.number.number	1.0.0 2.0.0	branch:project version=1:1



### Step 2: Creating a personal access token from github

URL : <https://github.com/settings/tokens> Connect your Github account

- In the left sidebar, under **Personal access tokens**, click **Tokens (classic)**.
- Select **Generate new token**, then click **Generate new token (classic)**.
- In the "Note" field, give your token a descriptive name.
- To give your token an expiration, select **Expiration**, then choose a default option or click **Custom** to enter a date.

- Select the scopes you'd like to grant this token. To use your token to access repositories from the command line, select **repo**. A token with no assigned scopes can only access public information. For more information, see "[Scopes for OAuth apps](#)."

- Click **Generate token**.
- Optional, to copy the new token to your clipboard, click

- Next to the token you'd like to authorize, click **Configure SSO**. If you don't see **Configure SSO**, ensure that you have authenticated at least once through your SAML IdP to access resources on [GitHub: Let's build from here](#). For more information, see "[About authentication with SAML single sign-on](#)."

- In the dropdown menu, to the right of the organization you'd like to authorize the token for, click **Authorize**.

### Step 3: Register project on test site?

URL: [Wonder QA Automation Admin](#)

1. Navigate to "APP Management" page
2. click "Add a New APP"
3. on "New Configuration" : Binding project and executor.

### How to make plan?

URL: [Wonder QA Automation Admin](#)

1. Navigate to "Test Plan Management" page
2. click "Make a Test Plan"

3. on "Make a Test Plan" :

- input plan name;
- choose app & app version

Make a Test Plan

Plan name: sprint

Platform: Wonder Auto Test Lab

App: Wonder

App Version: 4.8.0 4.7.0 4.6.0 4.5.0

Choose an App: sprint 04/16/2024 Wonder 4.8.0

Choose Test Cases: 0 Test Cases 0 Test Cases 4.7.0 4.6.0 4.5.0

Choose Test Devices: 0 iOS Phones 0 Android Phones

Cancel Next step

### 3. on "Make a test Plan" modal :

- select cases

Make a Test Plan

Unselected Cases 48 items

Case No.	Case Name	Feature	Story	Epic	Type	Priority
postrelease-011	checkOrderConfirmed		Place order, che...	PostReleaseTestCase	Functional	Critical
postrelease-012	checkOrderCancelled		Cancel order, che...	PostReleaseTestCase	Functional	Critical
postrelease-013	checkDeliveryWhenAtLocation		view some pages ...	PostReleaseTestCase	Functional	Critical

Selected Cases 1 item

Case No.	Case Name	Feature	Story	Epic	Type	Priority
postrelease-014	checkDeliveryWhenAtLocation		view some pages ...	PostReleaseTestCase	Functional	Critical

Choose an App: sprint 04/16/2024 Wonder 4.8.0

Choose Test Cases: 1 Test Cases 4.7.0 4.6.0 4.5.0

Choose Test Devices: 0 iOS Phones 0 Android Phones

C Cancel Previous step Next step

### 3. on "Make a test Plan" modal :

- select devices

Make a Test Plan

Unselected Devices 10 items

Device	Device Type	OS Version	Display	Device Sets
Samsung Galaxy Note20 Ultra	Virtual	Android 12	1440 x 3088	
Samsung Galaxy S23 FE	Virtual	Android 13	1080 x 2340	
Samsung Galaxy S23 Ultra	Virtual	Android 11	1440 x 3088	

Selected Devices 2 items

Device	Device Type	OS Version	Display	Device Sets
Apple iPhone 12	Virtual	iOS 16.0	1170 x 2532	Remove
Samsung Galaxy A55	Virtual	Android 10	720 x 1600	Remove

Choose an App: sprint 04/16/2024 Wonder 4.8.0

Choose Test Cases: 1 Test Cases 4.7.0 4.6.0 4.5.0

Choose Test Devices: 1 iOS Phone 1 Android Phone

C Cancel Previous step Next step

### 3. on test plan page :

- test plan : device task (1:n)

Test Plan

App: Wonder Platform: Wonder Auto Test Lab

Device	Device Type	OS Version	Display	Device Sets
iPhone 12 iOS 16.0	Virtual	iOS 16.0	1170 x 2532	Pending
Galaxy A55 Android 10	Virtual	Android 10	720 x 1600	Pending

Actions: Reports: Message:

Created: 04/16/2024 23:24:22

Duration: 00 mins

Case: checkDeliveryWhenAtLocation Case No: postrelease-014 Type: Functional Priority: Critical Device ID: Pending

Actions: Reports: Message:

Created: 04/16/2024 23:24:22

Duration: 00 mins

Case: checkDeliveryWhenAtLocation Case No: postrelease-014 Type: Functional Priority: Critical Device ID: Pending

Actions: Reports: Message:

Created: 04/16/2024 23:24:22

Duration: 00 mins

Case: checkDeliveryWhenAtLocation Case No: postrelease-014 Type: Functional Priority: Critical Device ID: Pending

Actions: Reports: Message:

Created: 04/16/2024 23:24:22

Duration: 00 mins

Case: checkDeliveryWhenAtLocation Case No: postrelease-014 Type: Functional Priority: Critical Device ID: Pending

## How to generate UI Difference Report?

URL: <https://test-report.foodtruck-qa.com> or <https://test-report.foodtruck-uat.com>

Step

1. navigate to "Standard Screenshot Management" page

2. click "add" to new standard screenshot

Standard Screenshot Management

Add Case

Case Name: checkDeliveryWhenAtLocation

Standard Screenshot Name: delivery\_order\_status\_orderReceived

Screenshot: delivery\_order\_status\_orderReceived

Screenshot: delivery\_order\_status\_preparing

Screenshot: delivery\_order\_status\_onTheWay

Screenshot: delivery\_order\_status\_delivered

Actions: Reports: Message:

Cancel Save

3. navigate to "UI Diff Case" Page

entry1: navigate to "APP Test Report Dashboard" page after test completed (test report site)

entry2: navigate to "Test Plan" page after test completed (test site)

demo:

- device\_task\_id = test\_session\_id = e1bffe2835ef40c7bcc2b6d25fd765ba
- plan\_id = 3c9064cb04a54d19842f2c652a1989dc

The screenshot shows the 'APP Test Report Dashboard' interface. At the top, there are search filters for 'Plan ID', 'Test Session ID', 'Test Start Time', 'Test End Time', and 'Device'. Below the filters, a table displays a single test session row. The session details include the session ID, plan ID, app name ('WONDER'), old app version ('4.7.0'), device ('iPhone SE (2020) iOS 13.7'), and duration ('26 min'). At the bottom right of the table, there is a red box around the 'entry' button.

The screenshot shows the 'Test Plan' page with a completed test session. The session details are identical to the dashboard. At the bottom right of the table, there is a red box around the 'Load as Standard Screenshot' button.

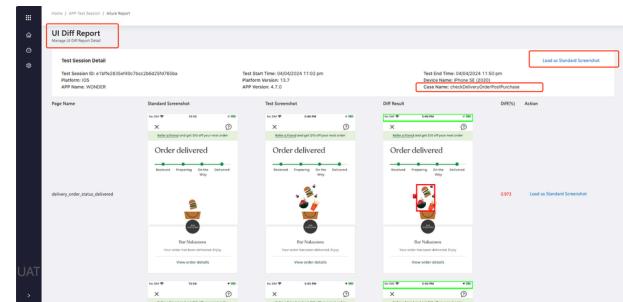
4. click "load as standard screenshot" to Updated standard screenshots for all test cases.

The screenshot shows the 'UI Diff Case' page for a completed test session. The session details are identical to the previous screenshots. At the bottom right of the table, there is a red box around the 'Load as Standard Screenshot' button.

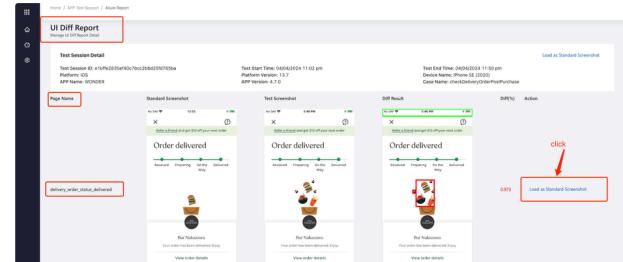
5. navigate to "UI Diff Report" Page

The screenshot shows the 'UI Diff Report' page for a completed test session. The session details are identical to the previous screenshots. At the bottom right of the table, there is a red box around the 'Load as Standard Screenshot' button.

6. click "load as standard screenshot" to update all standard screenshots for a test case



7. click "load as standard screenshot" to update a standard screenshot of a test case



8. navigate to "Device Screenshot Management" page, click "Device Standard Screenshot"

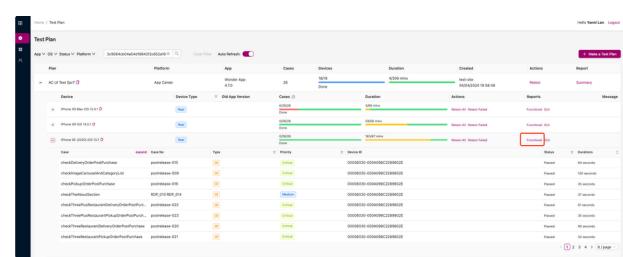
9. on "page, check device standard screenshot , demo  
a. test\_session\_id = e1bffe2835ef40c7bcc2b6d25fd765ba  
b. devcie= iPhone SE (2020)



## How to Generate Allure Report?

### step

1. entry: navigate to "test plan" page, on device task level: click "Functional".



2. on "Allure Report" Page, click "Generate Allure Report".

