



WUNDERFLATS

Profile Tab Test cases

Test Strategy

Revision History

Date	Version	Author	Description
11.11.2021	1	Abdul Wahab	Test Strategy for Profile Page

Table of Contents

1. Scope	3
2. Test Approach	4
3. Test Environment	5
4. Testing Tools	12
5. Risk Analysis and Critical Test case	13
6. Test Summary.....	14

1. Scope

Project Goals

The scope of this testing strategy/documents is to perform full round of testing for the profile tab available in the my account tab of Wunderflats portal (<https://en-hiring.wunderflats.xyz/>)f .The goal is to check the feature, functionality in the profile tab and write manual test suite and perform manual testing. Some of test case are automates using cypress.io framework, Cucmber.js with BBD format. In the case of automation all the test cases and proper test report is generated. All the testing activates perform mention in the documents.

Test Scope:

• Manual Test Cases Document
• Automated Test cases Using cypress
• Test report
• Bug Report

• Project Milestone: Two week Sprint
• Company: WunderFlats
• Department: Product and Engineering
• Team lead : N/A

2. Test Approach

According to given scenario our task is to test profile page and its all-related fields. The first step for testing is to define the test approach and identify how testing would be carried out. There different testing approach but we would use Analytics approach in which we are focusing testing on the most critical functionality (risk based) and understand the high-level. After analyzing the all the feature in the profile page we will define following types of test cases.

- Positive Test Cases/Functionality
- Negative Test Cases
- Performance Test cases
- Security Test Cases
- User interface Test Cases
- Compatibility Test Cases

Positive/Functionality Test Cases
• Verify and navigate to profile page URL
• Verify edit/update full name
• Verify edit/update address
• Verify edit/update email address
• Verify edit/update phone number
• Verify identity verification
• Verify edit/update birth date
• Verify edit/update Nationality

<ul style="list-style-type: none"> • Verify bubble counter in profile tab exactly matches with empty fields
<ul style="list-style-type: none"> • Verify that enter/tab key works
<ul style="list-style-type: none"> • Verify profile page on small screen (Mobile and iPad)
<ul style="list-style-type: none"> • Verify country code matches with country flag in phone number filed
<ul style="list-style-type: none"> • Verify phone number length
<ul style="list-style-type: none"> • Verify address suggestion list from Google
<ul style="list-style-type: none"> • Verify birth date format
<ul style="list-style-type: none"> • Verify edit/update country nationality
<ul style="list-style-type: none"> • Verify year only accepts only four digits
<ul style="list-style-type: none"> • Verify edit/update country nationality

As a customer perspective we should verify all negative test and validation for all the fields in the profile page. Following are the **negative test cases**

Negative Test Cases
<ul style="list-style-type: none"> • Verify empty/null full name
<ul style="list-style-type: none"> • Verify full name validation with special character
<ul style="list-style-type: none"> • Verify empty/null email address
<ul style="list-style-type: none"> • Verify wrong email format validation
<ul style="list-style-type: none"> • Verify already exiting email validation
<ul style="list-style-type: none"> • Verify empty/null phone number
<ul style="list-style-type: none"> • Verify phone number length validation
<ul style="list-style-type: none"> • Verify empty/null address validation

<ul style="list-style-type: none">• Verify empty/null birth date
<ul style="list-style-type: none">• Verify wrong birth date format
<ul style="list-style-type: none">• Verify birth date greater than current date
<ul style="list-style-type: none">• Verify birth date with all zeros
<ul style="list-style-type: none">• Verify empty/null nationality validation
<ul style="list-style-type: none">• Verify spar bar validation in all fields
<ul style="list-style-type: none">• Verify already existing email validation

In our test plan we should also define the some security test according to our profile page. Following are the security test we should considered

Security Test Cases
<ul style="list-style-type: none">• Verify profile page URL is accessible in Incognito window

We should also perform some user interface testing so as a customer perspective we should also validate the design of profile page and check everything works fine according to our requirements

User Interface Test Cases
<ul style="list-style-type: none">• Verify design of profile page with mockup design
<ul style="list-style-type: none">• Verify all buttons, drop down, icon of profile page
<ul style="list-style-type: none">• Verify overall design according to Figma/Zepline

In order to check the profile page on different browser and operating systems we will write some compatibility test cases

Compatibility Test Cases
• Verify profile page on different browser and OS
• Verify profile page on different screen sizes eg Tabs
• Verify profile page design in the mobile view

So Now we have a lot of test cases we need to **prioritize test cases** are on the basis of business requirements. First, we think about **most critical test cases** according to customer perspective. For example

- Verify customer can view profile page with proper design and all fields
- Verify customer edit and update all the fields
- Verify customer is able to view the profile page on small screen

Then we can check all types of positive test and validation (Negative test cases).

In the end we will perform security, Compatibility test and some User acceptance Test .

For the **test Automation approach** we will automates first the type of test case are which has high risk involved, then we will automate test cases which repeat again and again. After that we will automates all fields of profile page, all validation and edit/update functionality.

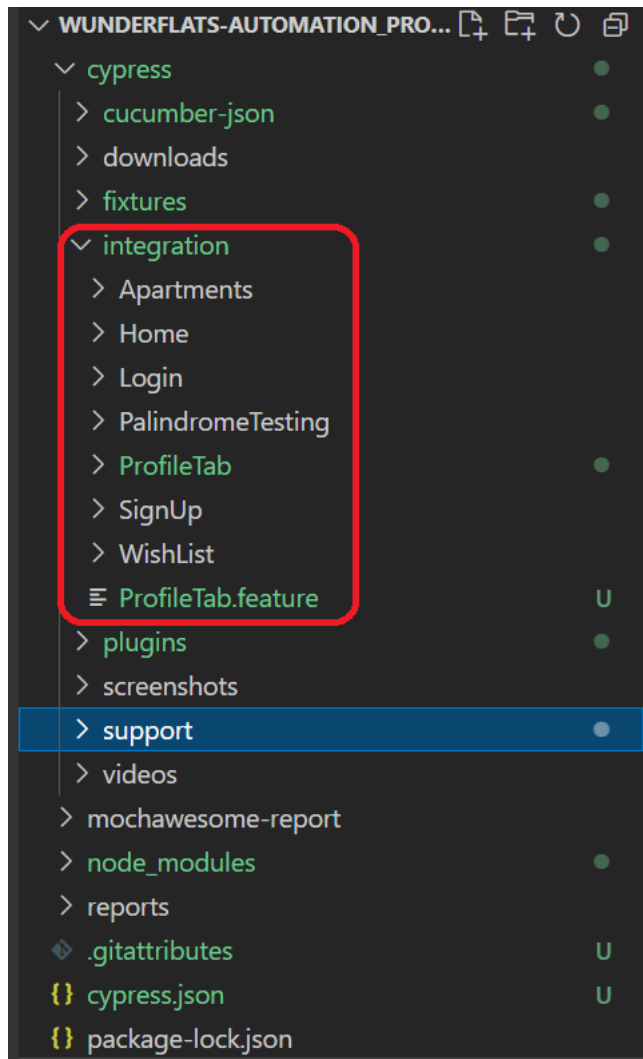
3. Test Environment

To perform the testing on profile page we will define manual and automation testing environment. After deeply analyzing the profile page feature then we will write manual test . In order to write manual test cases, we design test template using Microsoft excel. In the manual testing suite we define test scenario, issue key test description, test steps, actual and expected results .

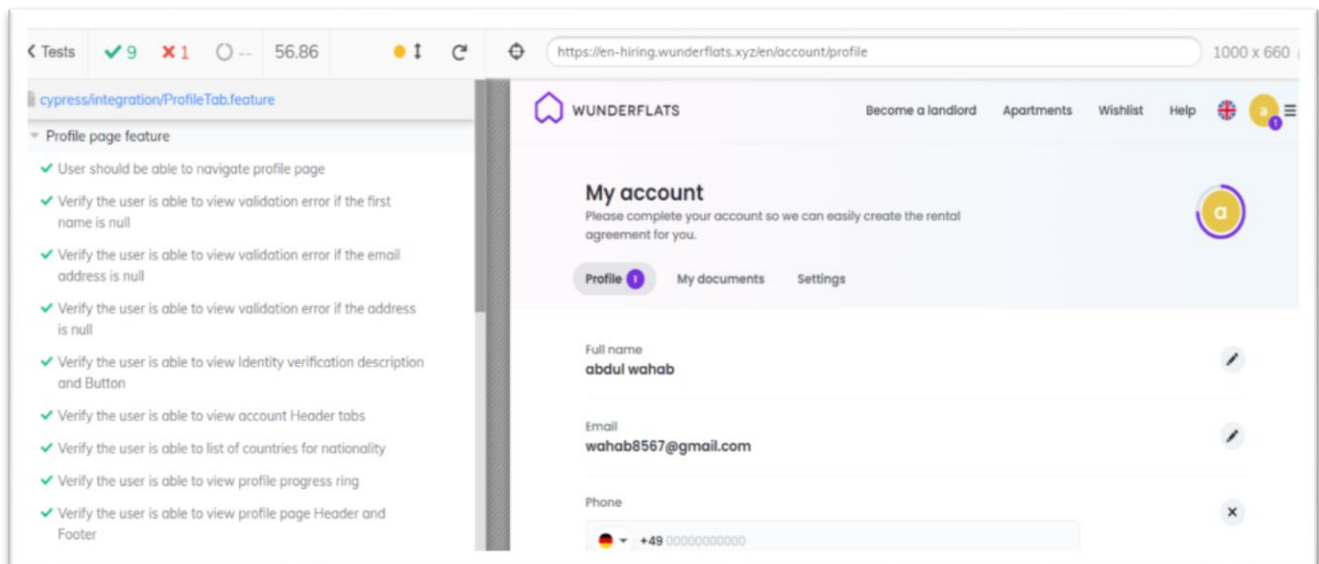
Test cases for User web Service Api											
Features	Test Scenario	Issue key	Affect versions	Fix versions	Type Pos/Neg test Automate/Manual	Priority	Component	OS/browsers	Description	Test Step	
ProfileTAB	General Test Cases										
	Navigate to profile URL								Scenario: Verify the user is able to navigate to profile Uri Given: User should be logged in When: A user click on my account in the drop down menu of menu button Then: User should be able to able to view profile page	1.User to write URL in the browser search box 2. User to press enter 3. Reload or refresh the page of browser	
	Verify profile page Uri is accessible in incognito window(Security Check)								Scenario: Verify the user is able to access profile from private browser Given: User should be logged in When: A copy the profile link in the incognito windows Then: User should not able to view or access the profile page		
	Verify all fileds & Validtaions in the sGerman language								Scenario: Verify the user is able to view all fileds and validation in german Language Given: User should be logged in When: A update the language of web app in German Then: User should be able to view all profile page fileds andvalidations in German language		
	Verify bubble count in the profile tab								Scenario: Verify the user is able to view all fileds and validation in german Language Given: User should be loined in		

Now we will automate our test cases, in order to automate our test cases we will use Cypress.io automation tool. We install Cypress.io in our current directory according to Cypress documentation. In order to automate test cases there are a lot of models in testing but I would prefer to use **Page Object Model**, which is the latest model for automation testing. We will make an object of every page according to application. For example, we are testing the profile page then we will create a profile page object in the support folder of Cypress. In this page object class we will write functions and read all the elements from the profile (e.g., button inputs, pop up text). As we are using

Behavior Driven Development (BDD) approach using cypress and cucumber.js. We will define one js file and one feature file in the cypress integration folder.



In the feature file we will use **Gherkin language** to define the test and in the JavaScript file we will implement the core functionality using cypress frame work .We will run the project and we just run our feature file only and we get the results in cypress test runner



Now we run all these test cases in headless mode using this command `cypress run --spec cypress/integration/*ProfileTab.feature --headless --browser electron`,

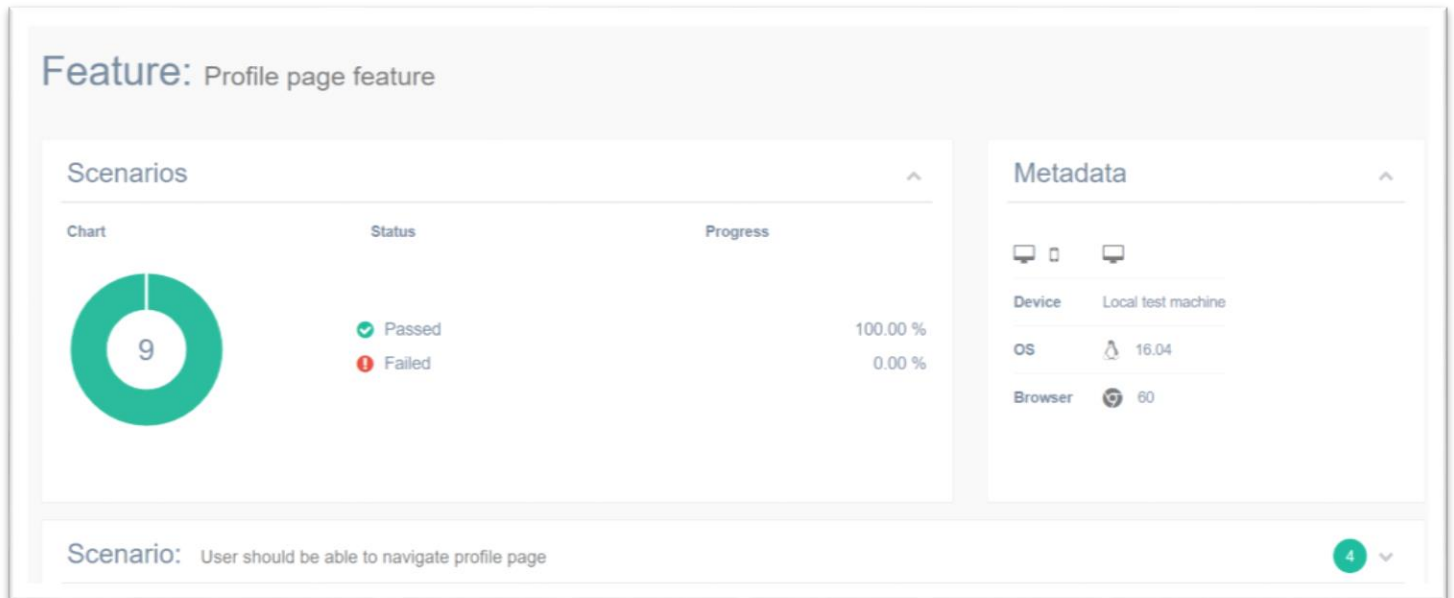
Spec	Tests	Passing	Failing	Pending	Skipped
✗ ProfileTab.feature	01:04	10	9	1	-
✗ 1 of 1 failed (100%)	01:04	10	9	1	-

Profile page feature

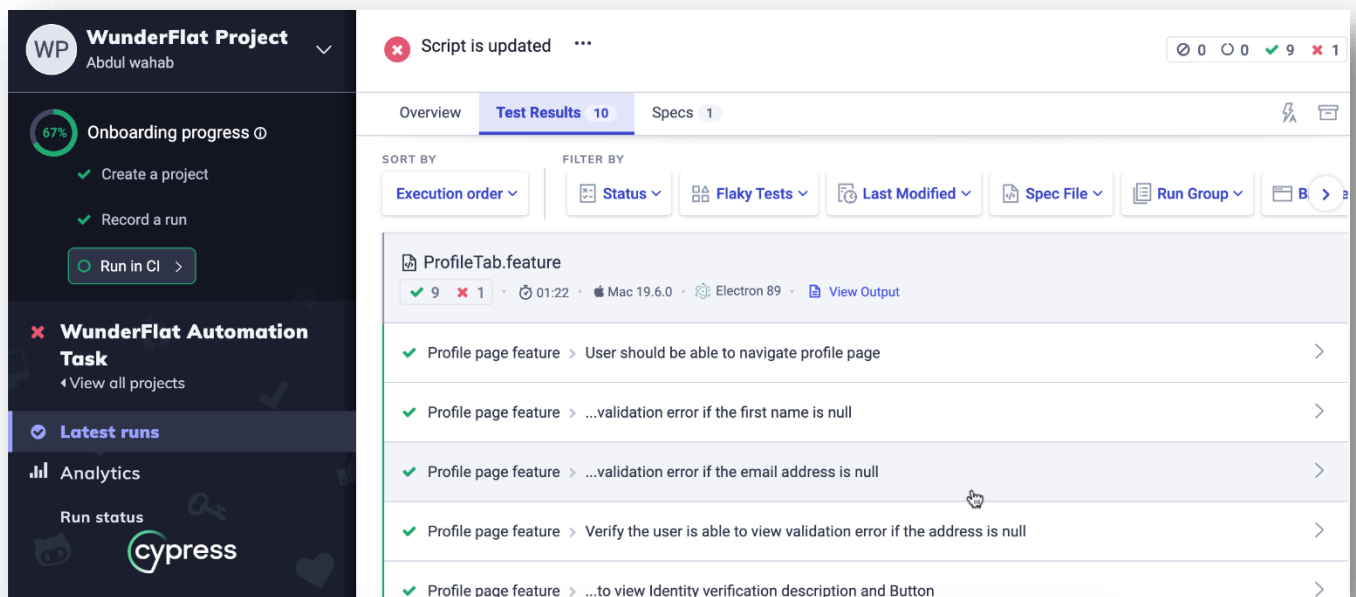
- ✓ User should be able to navigate profile page (52351ms)
- ✓ Verify the user is able to view validation error if the first name is null (2532ms)
- ✓ Verify the user is able to view validation error if the email address is null (1357ms)
- ✓ Verify the user is able to view validation error if the address is null (2106ms)
- ✓ Verify the user is able to view Identity verification description and Button (697ms)
- ✓ Verify the user is able to view account Header tabs (1806ms)
- ✓ Verify the user is able to list of countries for nationality (1219ms)
- ✓ Verify the user is able to view profile progress ring (327ms)
- ✓ Verify the user is able to view profile page Header and Footer (92ms)
- 1) Verify the user is able to view validation error if the phone number is null

9 passing (1m)
1 failing

To generate test report from our automation test cases we are using cucumber html report and here are the results

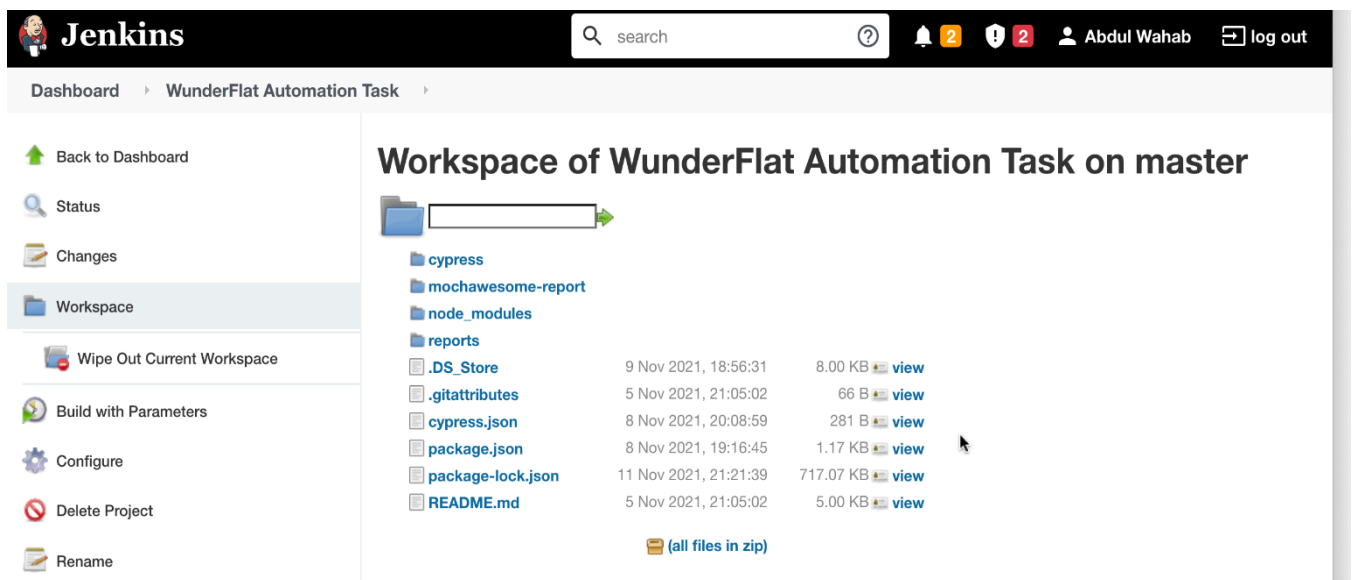


To generate test report from Cypress Dashboard we will connect our project with cypress dashboard by adding project id in cypress.json file, here are the results



Integrate Cypress project with Jenkins

Now we will integrate automation cypress project with Jenkins ,we install and configure Jenkins according to their documentation. Now we will create project in the Jenkins and connect Jenkins to your GitHub project (work place), Now we will run build with parameter and get our test results test report from Cypress Dashboard we will connect our project with cypress dashboard by adding project id in cypress.json file, here are the results



Jenkins | Dashboard | WunderFlat Automation Task

Workspace of WunderFlat Automation Task on master

File/Folder	Last Modified	Size	Action
cypress			
mochawesome-report			
node_modules			
reports			
.DS_Store	9 Nov 2021, 18:56:31	8.00 KB	view
.gitattributes	5 Nov 2021, 21:05:02	66 B	view
cypress.json	8 Nov 2021, 20:08:59	281 B	view
package.json	8 Nov 2021, 19:16:45	1.17 KB	view
package-lock.json	11 Nov 2021, 21:21:39	717.07 KB	view
README.md	5 Nov 2021, 21:05:02	5.00 KB	view

[\(all files in zip\)](#)

List of test cases which are Automated using Cypress

<ul style="list-style-type: none">• Verify Profile Page URL
<ul style="list-style-type: none">• Verify all the fields in profile page
<ul style="list-style-type: none">• Verify Full Name validation
<ul style="list-style-type: none">• Verify Email validation
<ul style="list-style-type: none">• Verify Address validation
<ul style="list-style-type: none">• Verify identification validation
<ul style="list-style-type: none">• Verify Country List in address section
<ul style="list-style-type: none">• Verify Google Address Api suggestion address

<ul style="list-style-type: none">• Verify Profile page header and footer
<ul style="list-style-type: none">• Verify profile progress ring

4. Testing Tools

Basically, our main focus is to validate and check all the feature for the profile page of wunderflats web app. According to features and web app requirements We use following tools, technologies, and programming languages.

<ul style="list-style-type: none">• Cypress.io Framework (For Automates test cases)
<ul style="list-style-type: none">• Mocha.js, chai.js and Cucumber.js with BDD (Behavioural DrivenDevelopment) approach
<ul style="list-style-type: none">• JavaScript
<ul style="list-style-type: none">• Jira
<ul style="list-style-type: none">• Mocha Awesome Reports
<ul style="list-style-type: none">• VS code
<ul style="list-style-type: none">• Microsoft Excel (For writing manual test cases)

5. Risk Analysis and Critical Test case

After running all test cases we just notice that some test are critical and its create some risk for profile page

- Special Character should not be allowed in First and Last Name
- Design is breaking (Text over flow) for user having long name (Character limit should be define)
- Blank Space in first and last name shows proper validation text
- German and English Language "Birth date" format is totally changed
- Future dates are accepted in the "Birth date" field. Eg: (Greater than current date)
- Birth date format is changed when user update language to German
- Date of birth greater then Leap year should not be allowed

6. Test Summary

We perform manual testing and automated testing for our profile page of wunder flats here is the overall three categories for test summary

- 1. In Scope**
- 2. Out of scope**
- 3. Area/Items not tested**

1. In Scope

We perform functional testing, Security testing, compatibility testing, Interface testing and user acceptance testing (positive and negative test)

2. Out of Scope

Some of key area of profile page is not tested. For example **load or Performance testing** is not implemented mean that when a lot of user update the some features of profile at same time then our profile page features/API works fine and give proper results in less time and web app stuck to hang state when no of user increased to E.g. 500

4. Area/Items not tested

Some tests are partially test because they are connected with third party system for example in profile page, we have identity verification using external software name as 'Veriff'.

In a nutshell above then 80 % test cases are passed there 20 % test cases which are failed. We will automate maximum test cases using cypress. Some to test cases we try to automate we are facing some authentication error which is already explained in the bug report