

# SOFTWARE QUALITY ASSURANCE

## IBCS-PRIMAX SOFTWARE(BANGLADESH) LTD.

*Presented by - Usrah Saba*

# Presentation Overview

Importance of SQA

Manual Testing – What, why & tested websites

Automation Testing – What, why & tested scenarios

API Testing – What, why & tested APIs

Conclusion – Key takeaways

# SQA (Software Quality Assurance)

SQA is a process that ensures software meets quality standards by preventing defects and improving reliability. It involves testing, process monitoring, and automation to deliver high-quality, error-free software.



ENSURES SOFTWARE  
RELIABILITY AND  
PERFORMANCE



DETECTS AND  
PREVENTS BUGS EARLY



REDUCES  
DEVELOPMENT AND  
MAINTENANCE COSTS



IMPROVES USER  
EXPERIENCE AND  
TRUST



ENSURES COMPLIANCE  
WITH INDUSTRY  
STANDARDS

# Manual Testing



**WHAT?** TESTING DONE MANUALLY  
WITHOUT SCRIPTS OR TOOLS



**WHY?** ENSURES USABILITY, UI/UX, AND  
DETECTS ISSUES BEFORE AUTOMATION



BEST FOR EXPLORATORY, USABILITY,  
AND AD-HOC TESTING

# Manual Testing on Insurance Website

Login/logout  
Functionality

Form  
Validation

Button  
Functionality

Dropdown Test

Input Handling

Log out

Home

Request Quotation

Retrieve Quotation

Profile

Edit Profile

Broker Insurance WebPage

Last modified: November 01 2019 11:00:04.

# Manual Testing Workflow (Profile')

Project Name	Insurance Broker System	Test Summary			Key (Test Status)						
Project Link	<a href="https://demo.guru99.com/insurance/v1/index.php">https://demo.guru99.com/insurance/v1/index.php</a>	No. of Features:	1	Pending	New test cases that has not been started yet						
Environment	Web Application (Tested on Chrome)	No. of Test Cases:	3	On Hold	Project/Module/Feature put on hold						
Assigned To	Usrah Saba	No. of Tested Cases:	3	Tested	Particular test cases has been tested						
Test Item Name	Profile Functionality	No. of Untested Cases:	0	Re-tested	Original test case failed and has been retested						
Test Type	System testing(Black Box)	No. of Re-tested Cases:	0	Key Test (Result)							
Test Objective	To verify if the data is displayed accurately on the profile page	No. of On Hold Test Cases:	0	Pass	Test case achived the expected result						
Total Test Cases	23	No. of Pending Test Cases:	0	Fail	Test cases failed to meet the expectation result						
Start Date	8/3/25	No of Passed Test Cases:	0	N/A							
		No. of Failed Test Cases:	3								

Test Case Description					Test Execution[ This column to be filled up while executing the test cases]							
Test Case id	Test Case Title	Precondition	Test case data	Steps to perform Test Case/Test Script	Expected Results	Actual Results	Pass/Fail	Time	Priority	Test Cycle No.	Test Date	Test Status
Profile_1	Check Profile" Section Visibility	User should be in Profile" Panel	N/A	Navigate to the Profile Header	Profile section should display user details	profile" section is non-functional, and no user details are shown	Fail		High	1	8/3/25	Tested
Profile_2	Profile Data Empty Fields Check	User should be in Profile" Panel	N/A	Verify if any fields are empty.	System should display a default message for missing data	No data displayed for the fields	Fail		High	1	8/3/25	Tested
Profile_3	Profile Data Accuracy	User should be in Profile" Panel	N/A	Verify if the profile data shown matches the expected data accurately	The profile should display the data without any errors	No data is shown in any field	Fail		High	1	8/3/25	Tested



# Manual Testing Workflow- (Edit Profile)

Test Case id	Test Case Title	Precondition	Test case data	Steps to perform Test Case/Test Script	Expected Results	Actual Results	Pass/Fail	Comments	Priority	Test Cycle No.	Test Date	Test Status
EditProfile_01	Verify Profile Update" Functionality	User should logged into the account and navigates to the edit profile panel	N/A	1. Open profile edit page. 2. Enter valid details. 3. Click "Update User".	Profile should update successfully with a success message.	The profile update button is not responding, and no success message is displayed.	Fail		High	1	8/3/25	Tested
EditProfile_02	Verify required fields validation	User should navigate to the edit profile panel	N/A	1. Leave one or more required fields 2. Try to submit the form.	The system should show an error message indicating the missing required fields. All required fields should be validated before submission	The form is submitted without showing any validation errors.	Fail		Medium	1	8/3/25	Pending
EditProfile_03	Validate phone number	User should navigate to the edit profile panel	Phone: 01234	1. Open profile edit page. 2. Enter an invalid phone number. 3. Click "Update User".	System should show an error message for invalid phone format.	The system accepts the invalid phone number	Fail		Medium	1	8/3/25	Pending
EditProfile_04	Verify DOB Format	User should navigate to the edit profile panel	N/A	1. Open profile edit page. 2. Enter a future date as DOB. 3. Click "Update User".	System should prevent saving an invalid DOB.	The system does not allow entering a birth year later than 1995.	Fail		Medium	1	8/3/25	Pending
EditProfile_05	Dropdown Functionality	User should navigate to the edit profile panel	N/A	1. Open profile edit page. 2. Select different options 3. Click Update User"	System should save the selected values correctly.	As Expected	Pass		Medium	1	8/3/25	Tested
EditProfile_06	Post Code Format	User should navigate to the edit profile panel	Post Code: A	1. Open profile edit page. 2. Enter an invalid post code 3. Click "Update User".	System should show an error message for invalid post code.	The system accepts the invalid postcode	Fail		Medium	1	8/3/25	Pending
						The "Update" button						

# Bug Report – Profile Section Not Displaying User Details

ation | Profile | Edit Profile

Title:

Firstname:

Surname:

Phone:

Dataofbirth:

License type:

License period:

Occupation:

Driver History:

ADDRESS:

Last modified: November 01 2019 11:00:04.

Bug ID: IBS-BUG-001

Module: Profile Section

Priority: High

**Issue:**

Profile section is empty after login.  
Users cannot see their saved details.

**Steps to Follow:**

Log in / Profile panel

**Expected Result:**

Profile section should show user details.

**Actual Result:**

Profile section is blank and not working.





# Automation Testing



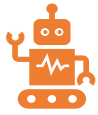
**WHAT?** USING SCRIPTS/TOOLS TO  
AUTOMATE REPETITIVE TESTING TASKS



**WHY?** SAVES TIME, INCREASES  
ACCURACY, AND SUPPORTS LARGE-  
SCALE TESTING



USED PLAYWRIGHT FOR WEBSITE  
AUTOMATION



# Automation Testing Workflow

→ ↺ demoqa.com/upload-download

≡ Elements ↑

Text Box

Check Box

Radio Button

Web Tables

Buttons

Links

Broken Links - Images

Upload and Download

Dynamic Properties

## Energy Storage Systems Webinar

Get insights on the application of Commercial & Utility Battery Energy Storage Systems.

⋮ ▶

### Upload and Download

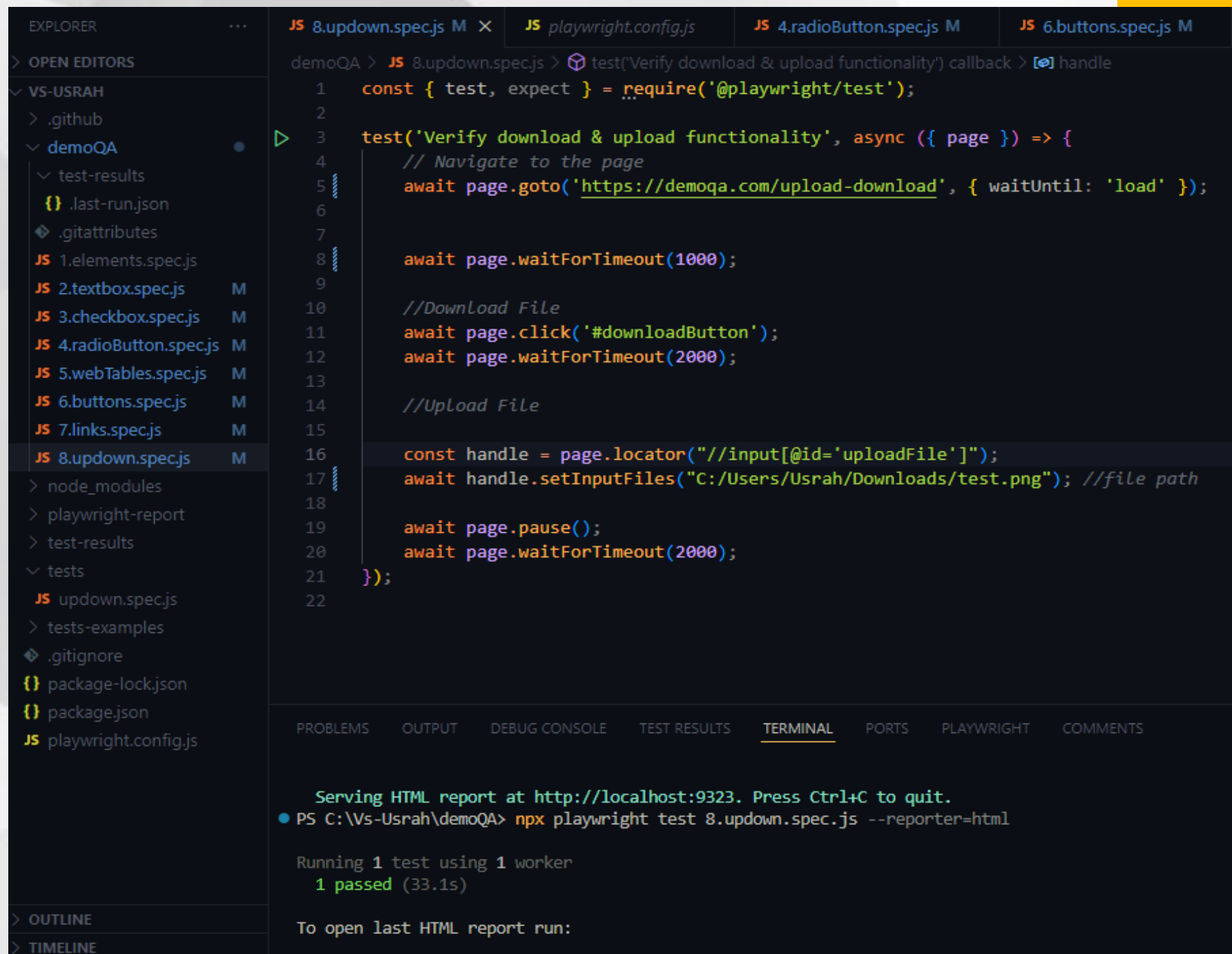
Download

Select a file

Choose File No file chosen

© 2013-2020 TOOLSQA.COM | ALL RIGHTS RESERVED.

# Automation Testing Workflow



The screenshot displays the Visual Studio Code interface with a Playwright test file open in the editor. The Explorer sidebar on the left shows the project structure, including a 'demoQA' folder with various test files. The editor window shows the content of '8.updown.spec.js', which contains a test function 'Verify download & upload functionality'. The terminal at the bottom shows the command to run the test with an HTML reporter, and the output indicates that the test passed successfully.

```
demoQA > JS 8.updown.spec.js > test("Verify download & upload functionality") callback > handle
1  const { test, expect } = require('@playwright/test');
2
3  test('Verify download & upload functionality', async ({ page }) => {
4    // Navigate to the page
5    await page.goto('https://demoqa.com/upload-download', { waitUntil: 'load' });
6
7
8    await page.waitForTimeout(1000);
9
10   //Download File
11   await page.click('#downloadButton');
12   await page.waitForTimeout(2000);
13
14   //Upload File
15
16   const handle = page.locator("//input[@id='uploadFile']");
17   await handle.setInputFiles("C:/Users/Usrah/Downloads/test.png"); //file path
18
19   await page.pause();
20   await page.waitForTimeout(2000);
21 });
22
```

PROBLEMS OUTPUT DEBUG CONSOLE TEST RESULTS TERMINAL PORTS PLAYWRIGHT COMMENTS

```
Serving HTML report at http://localhost:9323. Press Ctrl+C to quit.
● PS C:\Vs-Usrah\demoQA> npx playwright test 8.updown.spec.js --reporter=html

Running 1 test using 1 worker
  1 passed (33.1s)

To open last HTML report run:
```

# Automation Testing Workflow

All 1 Passed 1 Failed 0 Flaky 0 Skipped 0

## Verify download & upload functionality

8.updown.spec.js:3 27.8s

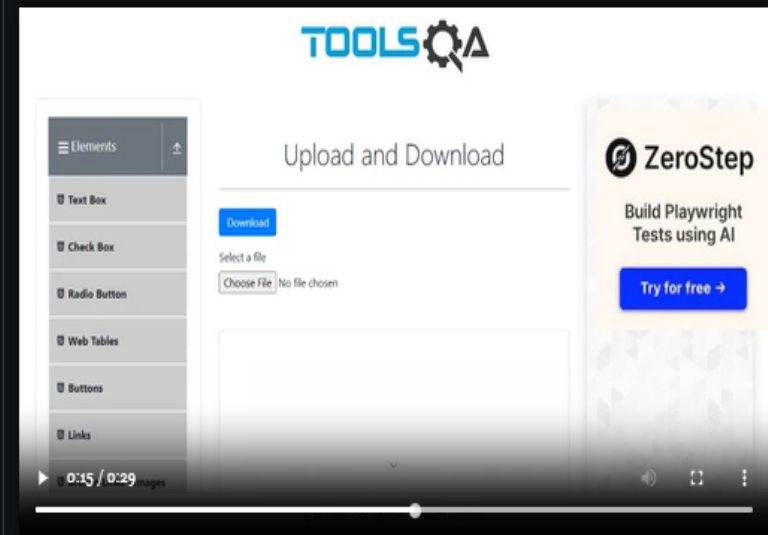
✓ Run

Test Steps

> ✓ Before Hooks	4.0s
> ✓ page.goto(https://demoqa.com/upload-download) — 8.updown.spec.js:5	21.3s
> ✓ page.waitForTimeout — 8.updown.spec.js:8	1.0s
> ✓ page.click(#downloadButton) — 8.updown.spec.js:11	280ms
> ✓ page.waitForTimeout — 8.updown.spec.js:12	2.0s
> ✓ locator.setInputFiles(//input[@id='uploadFile']) — 8.updown.spec.js:17	58ms
> ✓ page.pause — 8.updown.spec.js:19	2ms
> ✓ page.waitForTimeout — 8.updown.spec.js:20	2.0s
> ✓ After Hooks	60ms

Traces

Videos



The screenshot shows a web application interface for file uploads and downloads. On the left is a sidebar menu with options: Text Box, Check Box, Radio Button, Web Tables, Buttons, and Links. The main content area has a 'Download' button and a 'Select a file' section with a 'Choose File' button and the text 'No file chosen'. A ZeroStep AI assistant overlay is on the right, with the text 'Build Playwright Tests using AI' and a 'Try for free' button. The video player at the bottom shows a progress bar at 0:15 / 0:29.

video

# Application Programming Interface(API) Testing



**What?** API testing ensures that APIs work correctly, securely, and efficiently. It is a key part of software quality assurance.



**Why is it Important?** Ensures backend communication is smooth and data is exchanged correctly & securely



Uses tools like **Postman**, **JMeter** for API validation





# API Testing – Websites Tested



Tested Website: [Restful Booker API](#)



**Responses:** Checked if the API gives the correct status (like 200 OK) and data.



**Authentication:** Tested login and security to make sure only authorized users can access.



**Request/Response Validation:** Verified that the data sent and received is correct and complete.



- **Postman Procedure:**
- Create a Workspace and Collection.
- Add Requests:
  - **POST:** Create resource
  - **GET:** Retrieve data
  - **PUT:** Update resource
  - **PATCH:** Partially update resource
  - **DELETE:** Delete resource

# API Testing Workflow

The screenshot displays the Restful-booker API testing interface. On the left sidebar, the 'API Collection' is expanded, showing a list of endpoints: GET Auth - CreateToken, GET GetBookingIds, POST CreateBooking (selected), PUT UpdateBooking, DEL DeleteBooking, and GET Ping - HealthCheck. Below this, there are sections for 'Contract Testing', 'Integration Testing', 'Intro to Writing Tests', 'Mock Data Generation', 'Performance Testing', and 'Regression Testing'.

The main panel shows the details of the selected POST request to `https://restful-booker.herokuapp.com/booking`. The request is configured with the following body (JSON):

```
1 {
2   "firstname": "Jim",
3   "lastname": "Brown",
4   "totalprice": 111,
5   "depositpaid": true,
6   "bookingdates": {
7     "checkin": "2018-01-01",
8     "checkout": "2019-01-01"
9   }
10 }
```

The response is displayed below the request, showing a 200 OK status with a response time of 1.61 s and a body size of 947 B. The response body (JSON) is:

```
1 {
2   "bookingid": 2806,
3   "booking": {
4     "firstname": "Jim",
5     "lastname": "Brown",
6     "totalprice": 111,
7     "depositpaid": true,
8     "bookingdates": {
9       "checkin": "2018-01-01",
10      "checkout": "2019-01-01"
11    }
12   }
13 }
```

POST CreateBooking

DEL DeleteBooking

+

SQA project

API Collection / DeleteBooking

Save

Share

DELETE

https://restful-booker.herokuapp.com/booking/1 \

Send

Params

Authorization

Headers (10)

Body

Scripts

Settings

Cookies

Headers

8 hidden

	Key	Value	Description		Bulk Edit	Presets
<input checked="" type="checkbox"/>	Content-Type	application/json				
<input checked="" type="checkbox"/>	Cookie	token=abc123				
	Key	Value	Description			

Body

Cookies

Headers (11)

Test Results

201 Created

681 ms

753 B

Save Response

Raw

Preview

Visualize

1

Created

# API Testing Workflow

# Conclusion



Manual, API, and Automation testing are essential for software quality



Automation saves time but manual testing is still important



API testing ensures proper backend communication

The background features a complex, low-poly geometric pattern in various shades of gray. Overlaid on the right side is a faint, light-gray silhouette of a person's head and shoulders, facing right. The person appears to be wearing a hat or has a distinctive hairstyle.

# Thank You!

Looking forward to your feedback!