Turhan Kimbrough

Testing Stripe Checkout Using Input Space Partitioning

Software Quality Assurance Plan

Version: 1.0 **Date: (05/04/2021)**

Document History and Distribution

1. Revision History

Revision #	Revision Date	Description of Change	Author
1	5-09-2021	Include information about Firefox	Turhan

TABLE OF CONTENTS

1. INTRODUCTION

Our client, *Rosa Mary Health Foundation* uses a web-based business platform to receive donations for food banks and charities. Using Stripe Checkout, we have built a prototype donation page which will later be integrated into our client's platform.

Our goal is to develop tests which provide visual feedback of the webpage while simulating customer interactions. More specifically, we would like to track which types of payment details trigger successful and unsuccessful transactions.

1.1 Objectives

Our plan is to develop tests for the following criteria:

- Demonstrate the ability to collect payments from around the globe including Africa, Asia, Australia, Europe, North America, & South America.
- Demonstrate the ability to accept a variety of payment processing networks including VISA, Mastercard, American Express, Discover, Diners Club, & Japan Credit Bureau (JCB)
- Discover which combinations of credentials are accepted for the Stripe platform

1.2 Testing Strategy

To deliver the highest quality product, we strive to create tests which simulate a variety of customer interactions. Specifically, we will be providing different combinations of values for the input fields on the Stripe Checkout web form. This approach is known as *input domain testing*. We will use the coverage criteria of *multiple base choice coverage*, where the base set of input values represent successful transactions and deriving test cases represent other potential behaviors.

To maintain our standard of high quality, we will only integrate the Stripe Checkout web form if over 95% of our tests pass. A passing test indicates that the expected results match the actual results.

1.3 Scope

This document shall be included in the GitHub repository with the source code for the Stripe Checkout webpage. T-Visor's Consulting uses agile principles for software development and maintenance. Any changes requested from the client or internal members will be reflected accordingly in both the repository and this document.

1.4 Definitions and Acronyms

- Card Verification Code (CVC)

- Node.js – a back-end JavaScript runtime environment for dynamic web applications
- Unix – a family of operating systems characterized by modular design
- BASH – Bourne Again Shell, a command-line environment for Unix-like Operating Systems

2. TEST ITEMS

The Stripe Checkout web page is the primary software interface which will be tested.

This is an open source project available at https://github.com/T-Visor/stripe-checkout. To install the product, either clone the project using the git version control tool or download the project as a zip folder from the web page. This project requires Python 3, Selenium WebDriver, geckodriver for Firefox, and Node.js to function properly.

Python 3 download: https://www.python.org/downloads/

Selenium WebDriver: 'pip3 install selenium' or 'pip install selenium' using the command-line

geckodriver: https://github.com/mozilla/geckodriver/releases

Node.js: https://nodejs.org/en/download/

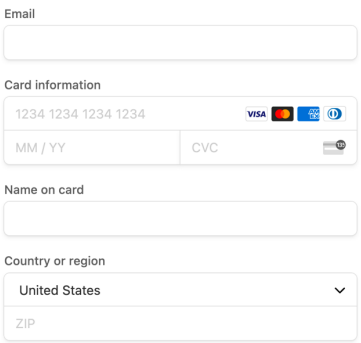
2.1 Program Modules

Solo developer/consultant Turhan Kimbrough will assume all responsibilities with regard to testing the software system.

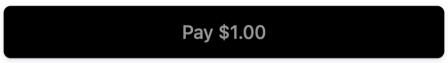
3. FEATURES TO BE TESTED

We test which combination of payment details will allow a customer to submit a donation payment. The payment details are entered via input elements on the web page.

This includes the 'Email' field, 'Card Number' field, 'Card Expiry' field, 'CVC' field, 'Name on card' field, 'Country or region' drop-down menu, and 'ZIP' field.



The primary feature being tested is the 'Pay $1.00' button which will submit payment information to the web server. By default the button will be grayed-out, indicating that payment details are insufficient.



Once the proper amount of information is in the web form, the button will be clickable and execute a payment transaction.

4. FEATURES NOT TO BE TESTED

We will not be testing post-transaction conditions such as declining a transaction, 3D authentication, or fraudulent transactions. Testing for these would require additional interactions depending on the context of the post-transaction condition.

In addition, we will not be testing mobile payment systems such as Apple Pay or Google Pay, since they require an Apple ID and Google account respectively along with a certain browser combination for each.

5. APPROACH

Input domain testing using multiple base block coverage. Since the web form is being tested to determine if the customer accepts the system, this will fall under *acceptance testing*.

Acceptance Testing

Input variables:

- Email field
- Card number field
- Card expiry field
- CVC field
- 'Name on card' field
- 'Country or region' dropdown field
- Zip field

Testable component (listener button): 'Pay $1.00' button

Characteristics:

- C1: Email field contains a valid email
- C2: Card brand used in Card number field
- C3: Relationship of card expiry field to current date
- C4: Number of digits in CVC field
- C5: Names used in Name on card field
- C6: Country selected for Country or Region field
- C7: Zip field is empty

Partitioning into blocks:

Characteristic	b1	b2	b3	b4	b5	b6
C1: Valid email	True	False				
C2: Card brand	Visa	Mastercard	American Express	Discover	Diners Club	JCB
C3: Relationship of card expiry to current date	Before	After				
C4: digits in CVC	3	4				
C5: Names on card	First	First & last name	First, middle, & last name			
C6: Country	United States	Japan	South Africa	United Kingdom	Australia	Brazil
C7: zip field empty	True	False				

Identify values:

Characteristic	b1	b2	b3	b4	b5	b6
C1	akira@persona.com	akira_kurusu				
C2	4242 4242 4242 4242	5555 5555 5555 4444	3782 822463 10005	6011 1111 1111 1117	3056 9300 0902 0004	3566 0020 2036 0505
C3	12/20	12/30				
C4	223	8784				
C5	Akira	Akira Kurusu	Akira Ren Kurusu			
C6	United States	Japan	South Africa	United Kingdom	Australia	Brazil
C7	21252	<empty>				

First Base block characteristics
(valid email, Visa, future expiration date, 3-digit CVC, First/Last name, United States, zip field not empty)

Second Base block characteristics
(valid email, American Express, future expiration date, 4-digit CVC, First/Middle/Last name, Japan, zip field empty)

Please refer to the external 'test-results' document for all 30 test cases. Each test case specifies the input values being used, the expected outcome, actual outcome, and whether the test passed or failed.

6. PASS / FAIL CRITERIA

6.1 Suspension Criteria

A situation which may impede the ability to continue testing is if Stripe servers experience latency issues and/or downtime.

6.2 Resumption Criteria

When Stripe Checkout page loads normally.

6.3 Approval Criteria

A test item 'passes' when the expected outcome description matches the screenshot (actual result) produced from the test. A test item fails otherwise.

7. TESTING PROCESS

7.1 Test Deliverables

A total of 30 tests will be developed using input space partitioning with a coverage criteria of multiple base block coverage. In addition, a BASH script will be provided to run all 30 tests at once.

A screenshot will be taken during each test and the results will be included in a separate document.

7.2 Testing Tasks

Install the appropriate Selenium driver software for your preferred web browser over at https://github.com/mozilla/geckodriver/releases

The Node.js server must be running before any tests take place. To run the Node.js server, use the command-line and enter the root directory of the project.

Change into the ‘server’ directory and run the command ‘npm start’.

```
~/r/stripe-checkout/server  master  ls                                  ✔  12:01:05 AM
README.md         node_modules      package-lock.json package.json      server.js

~/r/stripe-checkout/server  master  npm start                           ✔  12:01:19 AM

> stripe-sample-demo@1.0.0 start
> node server.js

Node server listening on port 4242!
```

To run the scripts, switch into the root directory of the project and navigate into the ‘input-space-partitioning’ folder. Execute the script named ‘run-all-tests.sh’ using the command ‘./run-all-tests.sh’ or ‘bash run-all-tests.sh’.

```
~/r/stripe-checkout/input-space-partitioning  master !2  ls                 ✔  12:09:15 AM
geckodriver.log  test-12.py  test-18.py  test-23.py  test-29.py  test-7.py
results          test-13.py  test-19.py  test-24.py  test-3.py   test-8.py
run-all-tests.sh test-14.py  test-2.py   test-25.py  test-30.py  test-9.py
test-1.py        test-15.py  test-20.py  test-26.py  test-4.py
test-10.py       test-16.py  test-21.py  test-27.py  test-5.py
test-11.py       test-17.py  test-22.py  test-28.py  test-6.py

~/r/stripe-checkout/input-space-partitioning  master !2  ./run-all-tests.sh  ✔  12:09:16 AM
```

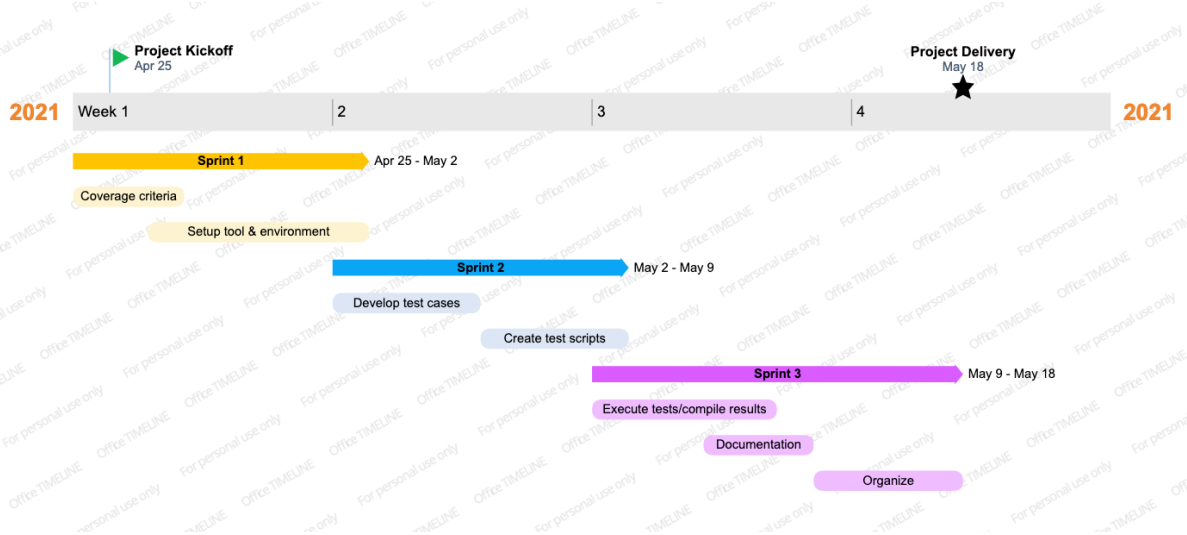
7.3 Responsibilities

Solo developer/consultant Turhan Kimbrough assumes all responsibilities.

7.4 Resources

Selenium WebDriver is the automated tool which will be used for testing with its source code available at https://github.com/SeleniumHQ/selenium. For this project, Selenium is required to be installed as a module for Python 3. Selenium is open-source and free to use.

7.5 Schedule



8. ENVIRONMENTAL REQUIREMENTS

8.1 Hardware

- At least 1GB of RAM
- Active internet connection

8.2 Software

- Unix-like Operating System
- Mozilla Firefox
- Node.js for the web server configuration
- Open computer port (port 4242 used in Stripe Checkout page)
- Python3
- Selenium WebDriver
- geckodriver for Firefox

8.3 Security

Ensure there is an encrypted connection on the Stripe Checkout page since payment details are submitted to Stripe's servers.

8.4 Tools

Basic knowledge of Python 3 syntax and Selenium module bindings will be required, since Selenium WebDriver acts as an API for different programming languages. No special qualifications will be necessary as Turhan Kimbrough will be responsible for individual and/or group training of new members to this project.