Project Test Plan
MotoMoto

The New Panelists
10/27/2021

Blake Del Rey
Isabel Guzman
Jacob Sunia
James Austin Jr.
Naeun Yu

Team Leader: James Austin Jr.

Table of Contents

1. Introduction

1.1 Purpose of this Document?

The purpose of the Test Plan document will provide the procedure and expectations for evaluation, performance, security, and validation for the MotoMoto application developed by the New Panelists. This document will be used to explain the plot of how testing will be done, scope of testing, testing objectives, approach, and how performance would be tested.

1.2 Testing Objectives

The main objective of the testing that will be done for this application is to ensure that our application functions as intended for users, engineers, and any other entity that interacts with MotoMoto in any way. In order to accomplish this objective, testing measures will be put in place to ensure that our features are all completed exactly as promised to the client, the application can support a large number of users at the same time and does not entirely break when that number is surpassed, is fast and responsive enough to not taint the user experience, and secure enough for users to trust that their data will not be extorted by any entity when given to us.

1.3 Scope

The current scope of the test plan will be to ensure that the GUI testing and component testing of the software is completed as per requirements provided by our client. Testing will be done to ensure that all new functions, existing functions, UI/UX, security, and core components are all tested to ensure that the system is operational.

The main goal of ensuring that all functionalities are mainly user input errors with our objective of achieving this goal is through mostly automated testing on our software and manual testing for usability. With the help of automated programs, this will make it easier to detect and resolve application vulnerabilities in a faster amount of time.

1.4 Features to Be Tested

The following list of features will be the subject of testing for the application. Some features are dependent on each other and will be used in separate testing

1.4.1 System UI

- Login and Registration
- User Accounts
- Car Builder

- Community Board and Community Board: Main Feed
- Part Flagging
- Event List
- Part Price Analysis
- Notification System
- Email Client

1.4.2 System Speed

- Because the application is being developed as a single-page application the page speed only needs to be tested for a single page and this page is the application home page.

1.4.3 System Security/Software Vulnerabilities

- The whole application will be tested as user security will be a primary goal of maintaining a report with our users. Since we plan on using a secured server all core components and dependent features will be tested.

1.5 Features Not Tested

All other features that are not included within the features that will be tested will not be included. This is subject to change based on our clients requirements of the application itself.

2. Testing Methodology

MotoMoto will be conducting both functional and non-functional testing of the application. The goal of this section is to explain how our team manages testing and explains in more detail of the features tested. All testing besides security testing will take place on a local host to ensure that released material will not be used. Our main goal will be to ensure usability, availability, and reliability tests are conducted with 99.999% passing marks.

2.1 Unit Testing

Unit testing will be used to test each unit, most likely each function, within the MotoMoto application. This automated testing will ensure that a function in a class satisfies its intention and performs the designed behaviors. This testing helps to find any bugs and flaws within the application in the earlier phase of development, which will ease the process of integration testing. Unit testing will be performed on major features and any necessary features that the end-users interact with.

Sub-Features that will be Tested

- Login/Register page:
    - Validate that all user inputs have valid usernames and passwords that are stored within our RDBMS.
    - We will test password lengths where each password must be at least 8 ASCII characters and will have a maximum of 15 ASCII characters.
    - Testing that usernames entered have no other characters other than english letters or underscores.
        - Spaces and dashes will not be allowed in usernames/passwords.
    - System will validate if a entered
- User Accounts:
    - User profile pictures will be tested so that images will not exceed 1 MB during .jpeg conversion.
    - Profile descriptions will be tested to ensure that the user can have an empty description or can type up to 500 characters.
- Community Board and Community Board: Main Feed
    - Testing will be done on the number of community board post characters before allowing user to post to the board (limit to 1-1500 characters)
    - Picture uploads will be tested to match our conversion criteria (up to 5 images per post and the size of each image doesn't exceed 1 MB during .jpeg conversion)
- Car Builder
    - VIN testing will be conducted as all entries must be 17 characters.
        - We will test to ensure that an error is occurred if a user does enter an invalid VIN
- Part Flagging
    - Validate the full compatibility of a part, if the part is compatible with the car.
        - Since all OEM parts are recognized by a specific vehicle, parts will be tested to evaluate recognizable parts to a vehicle that they were manufactured for.
        - Some aftermarket manufactured parts will also be recognized by a specific vehicle so, these parts will be tested to ensure virtually that a specific part is compatible.
- Event List
    - The zip code field will be tested as each entered number must be 5 digits long and will be tested to assure that no invalid zip codes are entered.
    - Checks the number of event post characters before allowing user to post to the board (limit to 1-2500 characters)

2.1.1 Testing Metrics

- Unit testing will be measured based on a pass or fail criteria on whether an independent feature of MotoMoto can work on its own.
- Since we will be developing the software to run these tests we will ensure that all potential errors will be recorded in external notes to ensure that each test that fails will be a task to work on for the following sprint

2.1.2 Risks of Unit Testing

- The risks of doing unit testing will be refactorization of the software architecture
    - This can become a problem if Software Design is not up to par and if the system needs an overhaul based on lack of proper structure within the application.
- Not writing correct unit tests can be poor testing which we should be testing based on the behavior of the application instead of how the code is written.

2.2 Integration Testing

Integration testing will be used to ensure that all individual components of the application are able to work in unison. Testing all subsystems within the application will be a key to ensuring both functionalities of the application and communication between microservices. Using an incremental testing strategy during integration testing will allow for a more iterative process to detect any errors and dependency defects.

- Incremental Integration Testing
    - Integrating modules one at a time, this method is more beneficial for the development team to detect any errors within the integration process for the immediate fix.
    - It tests as a group of modules to ensure smooth integration and data flow between modules. The testing will be used after the completion of unit testing.
    - Testing will be done on each dependent module in our application
        - Each dependent module will include the software in total as many of the features included in 1.4.1 will be included in integration testing functionalities
        - Each one of these tests will go through a functionality test to determine pass or fail states to help during code-review stages
- The testing will be done using internally generated source code.

2.2.1 Integration Testing Metrics

- Integration testing will be judged based on a pass/fail criteria of if a component/module is having any technical issues or vulnerabilities attached to it.

- To Enter Integration testing, unit testing must be completed first where information about the design of the integration will be noted
- Multiple of tests will be completed before a task is created to gain full report of what issues are present during the testing phase
- If all tests are passed, then the next iterative phase of implementation can take place.

2.2.2 Risks of Integration Testing

- Since the goal of integration testing is to validate components that can work together, there exists no real risks other than not writing complete tests. The lack of tests that are created can become a problem before deployment if this phase is not taken seriously throughout the SDLC which can be the cause of future implementation issues.

2.3 Performance Testing

The main objective of MotoMoto's performance testing is ensuring that no feature within our application holds back the performance of any other feature within our application or the application as a whole. Satisfactory application performance is imperative to ensuring a quality user experience which is why our performance testing will have multiple tests to ensure the application's performance is acceptable. These tests will be performed every Sunday at 1:00AM PST:

- Load Testing
    - Load testing is used to determine how many users MotoMoto can concurrently support on our application.
    - The following scenarios will be load tested. The format for each test will be in the format (duration in minutes, number of users)
        - (2, 100)
        - (2, 500)
        - (2, 1000)
        - (2, 10000)
        - (2, 100000)
    - If there is reason to suspect exceptionally high traffic within the following week we will also test the scenario (2, 1000000) to ensure our application does not crash under extremely high demand.
    - In order for these scenarios to be considered passed there must be 0 interrupted requests.
- Stress Testing
    - After figuring out MotoMoto's maximum user threshold through load testing, we will then use stress testing to determine our application's behavior when more users are concurrently using our application than supported.

    - From the user capacity determined from load testing we will use 5 incremental test cases to stress test with. If our capacity is represented by N then the following test cases will be used:
        - (2, N + 1000)
        - (2, N + 5000)
        - (2, N + 10000)
        - (2, N + 100000)
- General Performance Testing
    - Acting as a speed test, general performance testing is meant to provide an overall picture of how fast the webpage performs. The scores from performance testing is based on the following factors:
        - First Contentful Paint: The amount of time it takes from the beginning of the page loading to when the first piece of the page's content is rendered in the window.
            - Acceptable metric is between [0,1800ms]
        - First Input Delay: The amount of time when a user interacts with an element on the DOM, to when the page actually processes that event.
            - Acceptable metric is between [0,100ms]
        - Largest Contentful Paint: The amount of time it takes to render the biggest image or text block visible to the user on page load.
            - Acceptable metric is between [0, 2500ms]
        - Cumulative Layout Shift: The amount of time the largest set of element shifts take to occur.
            - Acceptable metric is between [0, 100ms]

2.4 Security Testing

Security testing is one of MotoMoto's main priorities as our plans are to ensure that our automated system is able to identify all weaknesses and loopholes that can cause information leaks. With the use of automated security testing, our main goal will be to ensure that threats, vulnerabilities, and risks are all uncovered to prevent attacks from potential malicious intruders.

Our goal will be to assign both penetration testing and vulnerability testing to multiple software engineers within our team and will be tested during new feature implementations which can happen during sprints) and on every Wednesday of each week. Having frequent security testing will ensure that we are taking an iterative approach to understanding what vulnerabilities are within our system.

2.4.1 Vulnerability Scans

During scans, we want to identify all internal and external problems within our software to understand if a vulnerability will be a security liability. We ultimately plan on using an open source automated tool to help verify the applications network and internal system vulnerabilities. These vulnerabilities include binary analysis, cross site scripting, SQL injection attacks, and platform vulnerabilities.

Using vulnerability scanning externally within our software will help determine whether MotoMoto is exposed/prone to attacks on our web server, relational database management system and application. This will be used to target external IP addresses and will be tested to verify the strength of the UI. Internally, our goal will be to ensure all vulnerabilities within our application such as communication within our application is secured. Ensuring that all potential risks are uncovered will help ensure that threats will be deflected and detected.

Vulnerability testing will include the following Automated Tests:

- Binary Analysis: Will sift through files that are composed of binary code and evaluate the applications data flow without the necessary needs of reverse-engineering bits.
- Cross Site Scripting: Evaluates the network of the application to ensure that no injections of malicious scripts or readings of sensitive page content to a victim user.
    - The target will be to identify all the pages in the web application finding all injectable parameters in forms, headers, and URLs while reporting information to the tester.
- SQL injection attacks: Scans will be done to protect each registered user's personal information as queries can become intercepted and modified disrupting the applications content.
    - SQL injection attacks vulnerabilities precautions need to be taken to avoid the compromise of the network server, all back-end infrastructure, or DoS attacks.
- Platform/Feature Vulnerabilities: These scans will be used within our application to ensure that collection, maintenance, and disseminating information are all discoverable within vulnerability testing.
    - This will be used to identify any vulnerabilities within the RDBMS with the use of SQL injection vulnerability protection, we plan on also testing other vulnerabilities that can occur with the database and network server (i.e Initial Deployment Failures, Misconfigured Relational Database, and Inadequate Auditing).

2.4.2 Automated Penetration Testing

Penetration testing will be used to simulate a hacker within MotoMoto to ensure that all vulnerabilities that are scanned as a potential issue are ethically exploited. With the implementation of third party software we will prioritize all flagged issues that can cause a data/network breach and attack the system to fully understand the vulnerability.

Using Vulnerability scans and to find all potential risks in the source code will allow for a thorough test of the applications security architecture. In our case we will be using penetration tests on discovered vulnerabilities so that all internal components of the application are protected (such as user data management, application data, and network security).

Features to be Penetration Tested:

- Web Application Tests: Ensuring that all functionalities and interfaces with sets of data in the web application are thoroughly tested by information from Vulnerability Scans
    - Back-end technologies (RDBMS, Dependencies, High level Programming Language)
    - RESTFUL API's
- Network Service Tests: Evaluating the network system and the services provided for probable security issues and will be knowledgeable in penetration of solving security issues with:
    - Network Technologies (firewalls, switches, routers)
    - Network Protocols (HTTPS)

Types of Internal Penetration (White Box Testing) that will be used

- SQL injection
- Proxy Server Attacks
- Identification and Authentication Failures
- Cross Site Scripting
- File Upload Flaws

External Penetration (Black Box Testing)

External penetration will be more focused on attacks on the whole system itself where the IP is given to our automated system to try to simulate possible attacks around the perimeter of our network. The main goal will be to compromise the network's firewalls where an assessment will take place to see what is exposable within MotoMoto.

2.4.3 Testing Metrics

- Vulnerability Scans
    - We plan to scan our vulnerabilities every Wednesday of the week to ensure that all potential risks are uncovered

    - Each scan will vary in time as it will depend on the number of IPs that are accessing the application
    - We will measure the vulnerability scans by the
        - Number of Critical Vulnerabilities Detected
            - Critical security scanning possibilities of SQL injection
            - The application can be compromised by attackers
        - The Scanners distance of Coverage over the internet
- Penetration Testing
    - Penetration testing will be measured based on the success of the attack itself and visibility of what needs to be fixed
    - Each automated penetration test can take from 15 minutes to 2-3 days depending on the systems that are being used to test the application
    - Mostly, penetration testing will be used to attack known and unknown vulnerabilities within the network and application to identify specifics of each test.
    - Penetration testing will be conducted during each inclusion of a new feature which can potentially take a week during each sprint to complete after gathering information from the vulnerability scanner.

2.4.4 Risks of Security Testing

- Vulnerability Scans have very little risks compared to penetration testing as there can be intrusions that take place at the same time of testing.
    - The pros of vulnerability testing significantly outweigh the risks of testing as there could be crashes due to programming errors that the team sets.
- Penetration testing can have many different issues that can occur.
    - Since we will be using penetration to exploit vulnerabilities, system outages could take place due to reckless behaviors during the implementation of white box testing and can lose productive hours if the application is simply not available to be modified.
    - We can also come across false negatives (undetected vulnerabilities) which can be the cause for concern as testing will be only limited to so much time.

2.5 Software Testing: Usability Testing

Usability testing measures the application's ease of use and how the users interact and behave with the application. This testing gauges the user-friendliness of the application through the user experience and interactions. It provides feedback from potential users, and any negative feedback can be corrected before the public release of an application. This testing is an iterative process, and it will be performed repeatedly until the application is not confusing to use and the user behaviors are as expected while the

users are interacting with the application. Constant usability testing reduces the risk of creating a confusing application and saves budget and other costly resources.

2.5.1 Testing Metrics

- Usability testing will be used at least once every sprint to evaluate the application's user-friendliness among the clients and other potential users.
- Testing feedback will be validated and measured based on what the feedback of the client has noted on what should be modified for their liking
- All unit testing and integration testing should be completed to avoid all bugs that are viewable by the user.

2.5.2 Usability Testing Risks

- With the conduction of usability testing we can face problems with out of scope implementation issues that will ultimately change the shape of our software architecture.
    - Software architecture changes are going to happen but, our goal will be to keep a consistent shape to avoid overhauls.

2.6 User Acceptance Testing

Acceptance testing will be used during the final stages of the applications release to the client. The main goal here will be to ensure that each feature within MotoMoto will be working based on the clients requirements and is acceptable during deployment. This mostly will represent our Alpha and Beta testing that will most likely be the client and outside users to help the developers use the application for themselves before the applications due date (Scheduled May 2022).

With the integration of end-user testing, this process will ensure that all levels of testing have been completed and the application is ready to be deployed. There should be very little to no vulnerabilities and the only errors that are acceptable is if the application should have cosmetics errors within the style sheets.

All of this testing will be manually contributed by outside users as it will be to ensure that the application will be working properly. This testing is done at the end of the SDLC where the client or any interested users will be able to actually run the application itself. All testing will be monitored by reviewal email or verbal messages to the developers to ensure that everyone is on the same page of what is liked or not liked about a specific view of the application. This testing will only happen once in the project's actual lifecycle and will happen potentially a week or two before release date.

2.6.1 User Acceptance Testing Metrics

- Our testing will mostly be based on what our users experience outside of our clientele. Our goal will be to receive feedback from outside users to ensure the likeness of the application by car enthusiasts. User acceptance testing will only happen once during the SDLC.
- Testing will be conducted through email reviews or in person feedback on what was liked or disliked about the application
- Using this feedback, we will be able to shape the application better for future thought processes on the release of other versions past initial release.

3.0 Test Cases

3.0.1 Testing Criteria

Testing severity will be judged based on the following criteria

LEVEL	Description
CRITICAL	Critical severity will be the group of all core components in the application.
MODERATE	Moderate severity is based on all unique features of the application
LOW	Low severity is based with primarily external components or any unique features interaction with external resources

3.1 Login System

Login System is a *CRITICAL* testing requirement the login feature is a core component to the application itself. Since our application is generated more for our users. Since the login system is going to be required to access some functionalities of the application itself, it will be a main priority to include testing ensuring the usability of the program.

Login - Unit Testing:

Login system input will be tested to ensure that the user enters a valid username or password. Simple validation checks of the characters used will calculate the number of characters entered for a password after a user clicks on the enter button to login.

Testing on the login feature will include test scenarios as follows:

Login Unit Testing Pass/Fail Criteria		
	Pass Criteria (Valid Entries)	**Fail Criteria (Invalid Entries)**
Username	Username does not have any other characters than letters or underscores.	User provides invalid username where other characters are entered other than letters and underscores.
Password	User enter valid username and username length is in the range of 8 characters to 15 characters	User provides invalid password where either the length of the password is greater than the 15 character limit or the password is less than the 8 character minimum.

Login Unit Testing Test Cases		
Scenario	**Username**	**Password**
Valid Test Case	BobNAlice	ABC#$%123
Invalid Test Case	Bob&Alice	ABC@##$@15
Invalid Test Case	BobNAlice	ABC

Login - Integration Testing:

Login system integration testing will be used to work with the backend of the application to ensure that all communication is being handled to log a user onto the application. Validation checks will be handled through the backend business logic to determine if the user is registered with MotoMoto.

Login Integration Testing (Pass/Fail) Criteria		
	Pass Criteria (Valid Entries)	**Fail Criteria (Invalid Entries)**
Stored Information (RDBMS)	Username and Password are jointly stored within the RDBMS and the system validates.	Username or password is not jointly stored within the RDMS and the system rejects the user to re-enter

		username or password
User Input Values	Unit Testing Cases must be passed first before integration testing is valid	Users who fail will be prompted error message displayed if invalid login was entered

Login Integration Testing Cases			
Scenario	**Assumption**	**Username**	**Password**
Pass	User ‘JimBone’ is in the database and ‘AppleSauce11’ is a valid password that matches the username	JimBone	AppleSauce11
No existing account in RDBMS	Username ‘JimmyBone’ is NOT in the database and ‘AppleSauce11’ is an valid password (based on format) that has no ties to a username ‘JimmyBone’	JimmyBone	AppleSauce11
Incorrect Password	Username ‘JimBone’ is in the database and ‘AppleSauce13’ is an valid password that does not match the username	JimBone	AppleSauce13

1. Return Values in Valid Test Case: User is directed back to the home page where they are free to use the all application features
2. Return Values in Invalid Test Case 1: User is prompted with an error message ‘Invalid Username/Password’ please try again
3. Return Values: User is prompted with an error message ‘Invalid Username/Password’ please try again

3.2 Registration

Registration will also be a *CRITICAL* point of testing for the web application as it will ensure that users will be able to register for new accounts. Since registration is a key component to allowing access to all of the features MotoMoto offers.

Registration - Unit Testing:

For registration tests, unit tests will ensure that the user is entering the correct information such as email addresses, usernames, password, re-entered password, and ToS are agreed upon. Most of our unit tests will be to check our business logic that will be used in the front end portion of the application.

Unit Testing on the Registration would include the following criteria:

Register Unit Testing Pass/Fail Criteria		
	Pass Criteria (Valid Entries)	**Fail Criteria (Invalid Entries)**
Username	Username does not include any other characters other than US English Letters and underscores	Username contains characters outside the range of pass criteria characters (Usernames that include other characters besides US English Letters and Underscores)
Password	Password must be in the range of 8-16 ASCII characters	Passwords are shorter than 8-16 Characters and are not in the ASCII chart of characters
Password Re-Entry	Password Re-Entry must be the exact same entered password as the first entered password	Password re-entries are not the same as the first entry
Email	Email must be formatted in correct fashion prefix@domain.com	Email not formatted in correct fashion
Terms of Services (ToS)	Terms of Service must be confirmed and read to move onto registration.	Terms of Services is not selected as agreed after viewal

Registration Unit Testing Cases			
Scenario	**Pass**	**Fail**	**Reason for Failure**

Username: Bob_N_Alice Password: abc123@BC Re-Enter Password: abc123@BC Email : bobNalice@aol.com Terms of Services : Agree	X		
Username: Bob&Alice Password: abc123@BC Re-Enter Password: abc123@BC Email : bobNalice@aol.com Terms of Services : Agree		X	Invalid Username, Bob**&**Alice should not be allowed since ampersand is not a US English Letter or Underscore
Username: Bob_N_Alice Password: ABC (First Entry is Invalid) Re-Enter Password: ABC Email : bobNalice@aol.com Terms of Services : Agree		X	Invalid Password, where ABC is less than the minimum requirements of a password of (all passwords must be within 8-15 ASCII characters)
Username: Bob_N_Alice Password: abcdefghijklmnop Re-Enter Password: abcdefghijklmnop Email : bobNalice@aol.com Terms of Services : Agree		X	Invalid Password, where ‘abcdefghijklmnop’ would be considered greater than the available limit of passwords being only ASCII characters in the range of 8-15
Username: Bob_N_Alice Password: abc123@BC Re-Enter Password: abc123ABC Email : bobNalice@aol.com Terms of Services : Agree		X	Invalid re-entry of a password, this would fail because abc123**A**BC is not the same exact entry as abc123@BC
Username: Bob_N_Alice Password: abc123@BC Re-Enter Password: abc123@BC Email : bobNalice.com Terms of Services : Agree		X	Invalid email address entry would cause a failure because it does not meet the requirements of a prefix@domain.com, prefix@domain.org, etc.
Username: Bob&Alice Password: abc123@BC Re-Enter Password: abc123@BC Email : bobNalice@aol.com		X	Invalid for not accepting the terms of service which will allow for a new user to create an account on the application

Terms of Services : N/A			

Registration - Integration Testing:

Register system integration testing will be used to work with the backend of the application to ensure that all communication is being handled to record a new user onto the application. Validation checks will be handled through the backend business logic to validate an email address and username ensuring that all information entered is not taken/in use.

The criteria that we will be using for integration testing are as follows:

Registration Integration Testing (Pass/Fail) Criteria		
	Pass Criteria (Valid Entries)	**Fail Criteria (Invalid Entries)**
Registering New Users onto the Database(RDBMS)	Username and Email is not stored in the RDBMS by any other registered where a new account will be created	Username and Email is stored in the RDBMS by a registered user
Running Application Servers	Network servers are up and running so users are free to register for accounts	Network servers are down where all users have no access to the registration page.

Registration Integration Testing Cases			
Scenario	**Pass**	**Fail**	**Reason for Failure**
New user creates an account and attempts to create with the username ‘Bob_N_Alice’ (Not Taken) and email ‘BobNAlice@aol.com’ (Not Taken) and servers are running	X		
New user creates an account and attempts to create with the username ‘Bob_N_Alice’		X	Username is taken by another user, therefore the registration process cannot continue without a username

(Taken) and email ‘BobNAlice@aol.com’ (Not Taken) and servers are running			that is not taken.
New user creates an account and attempts to create with the username ‘Bob_N_Alice’ (Not Taken) and email ‘BobNAlice@aol.com’ (Taken) and servers are running		X	Email is currently in use by another user, therefore the registration process cannot continue without a valid email entry
New user creates an account and attempts to create with the username ‘Bob_N_Alice’ (Not Taken) and email ‘BobNAlice@aol.com’ (Not Taken) and servers are NOT running		X	Network servers are not running, therefore creating a new account is impossible at the specific moment.

3.3 Car Builder

Car builder will be a *MODERATE* point of testing for the web application as it will ensure that users will be able to modify vehicles that are listed either by VIN or by vehicle search which is provided by our application. This service will be offered to all users whether they are registered or not, so that excludes any dependencies with the login/registration process itself unless a user wants to save a vehicle build. The functionality of this feature will be accessible from the homepage to a redirected portion of the homepage.

Car Builder - Unit Testing:

For unit testing on the car builder portion of the application, each VIN entry will be tested to ensure that the length of the entry is not greater than what is naturally expected from a legal VIN (All values in between 11 characters and 17 characters). Unit Testing will not be done on any other user entry as all available modifiable vehicles will be listed via dropdown selection tabs.

Carbuilder Unit Testing will be based on the following criteria:

Car Builder Unit Testing (Pass/Fail) Criteria		
	Pass Criteria (Valid	**Fail Criteria (Invalid**

	Entries)	Entries)
User Input VIN	VIN is within the character range from 11-17 characters	VIN has either greater than 17 characters in the search tab or less than 11 characters

Car Builder Unit Testing Cases			
Scenario	**Pass**	**Fail**	**Reason for Failure**
VIN : 4JGBF71E87A169189	X		
VIN : 4JGBF71E87A169189999		X	VIN entry is longer than the requirements (17 characters) and will not complete search
VIN: 4JGBF		X	VIN entry is shorter than the requirements (11 characters) and will not complete the search

Car Builder - Integration Testing:

Car builder integration testing will be used to communicate with other features within the application such as our RDBMS which will allow logged in users to save their respective vehicle builds to their account, Vehicle APIs which allow users to select VIN numbers or access vehicles, and access different parts and compare parts from part price analysis. Overall, the car builder feature will be dependent on other features that use translated information to display to the UI.

Car builder integration testing will be based on the following criteria:

Car Builder Integration Testing (Pass/Fail) Criteria		
	Pass Criteria (Valid Entries)	**Fail Criteria (Invalid Entries)**
Vehicle Search VIN	VIN entry passes unit testing and is available within the APIs	VIN does not pass unit testing and/or fails during search of vehicles within APIs

Saving Vehicles to Registered Account	User is logged into a registered account and network servers are running	Network Servers go down so users cannot save or a user is not logged into a registered account

Car Builder Unit Testing Cases			
Scenario	**Pass**	**Fail**	**Reason for Failure**
Unit Testing on VIN and vehicle is available to be modified via API	X		
User is logged in and attempts to save a vehicle build that has been started	X		
VIN number searched is not valid within the Vehicle API		X	Since VIN fails integration testing because the entered value is not available
User is not logged in and attempts to save a build		X	This will fail but without an error because MotoMoto will prompt the user to create an account.
Unit Testing on VIN and vehicle is available to be modified via API and servers are down		X	Since there is a valid entry for VIN, since the Network Servers are down, there is no access to the application

3.4 Community Board

Community Board will be a *MODERATE* point of testing for the web application as it will ensure that users will be able to view, post, and comment on vehicles or community posts they enjoy. This service will be offered to all registered users and includes dependencies from car builder, login services, and part price analysis. Each post will be verified based on the content in the post and based on the server.

Community Board - Unit Testing:

For unit testing on the community board portion of the application, each post will be tested to ensure the length of a posts title and description, the testing of upvoting once,

and comment length. Unit Testing tested to ensure that each individual component of the community board in the UI/UX is able to work without breaking the application.

Car builder unit testing will be based on the following criteria:

Community Board Unit Testing (Pass/Fail) Criteria		
	Pass Criteria (Valid Entries)	**Fail Criteria (Invalid Entries)**
Post Title	Each post title is between 15-75 characters	Post titles are either less than 15 characters or more than 75 characters
Post Description	Each post description is in between 1 character minimum to 1500 character maximum	Posts have no characters or have more than 1500 characters in the description. 0 character descriptions will not post anything
Upvotes	Each user should be able to upvote one post once	Upvotes are allowed to happen multiple of times per post
Comments	Each comment should be from 1 character to 1000 characters	Posts that are greater than 1000 characters. 0 characters will not allow a user to comment
Image Uploads	Each image must be within 1 MB upload after .jpeg conversion	Images that exceed 1 MB after conversion

Community Board Unit Testing Cases			
Scenario	**Pass**	**Fail**	**Reason for Failure**
Post Title: ‘The Mystery Machine’	X		
Post Description: ‘Hello Scoob’	X		

User will be able to upvote once	X		
Comments: ‘I love scooby snacks’	X		
Image Upload: Compression of image to 900 kilobytes	X		
Post Title: ‘Hi’		X	This will fail because the title ‘Hi’ does not meet the minimum character requirements of 15 characters.
Post Description: Greater than 1500 characters		X	This will fail because of the maximum characters allowed in a description
Multiple upvotes on a single post		X	This will fail because users should only be able to upvote once per post
Comments: Greater than 1000 characters		X	This will fail because of the maximum characters allowed in a comment
Image Upload: Compression of image greater than 1 megabyte		X	This will fail because each post only allows 1 MB per compressed image added to a post.

Community Board - Integration Testing:

Community Board integration testing will be used to ensure communication with other features within the application to upload to a user post. Community Board will only be accessible to logged in users which means that the feature is dependent on other functionalities of the application itself such as car builder, login system, part price analysis, and external APIs. General Feed is accessible without logging in but is dependent on the community board itself.

Car builder integration testing will be based on the following criteria:

Community Board Integration Testing (Pass/Fail) Criteria		
	Pass Criteria (Valid Entries)	**Fail Criteria (Invalid Entries)**

Accessing Community Boards	User is logged into their account where community boards can be accessed	User will just be able to view general feed where no access to specialized content is
Posting to a Community Board	User posts are logged into an account and post meets all unit test passing criterias allowing post to be stored onto the systems servers	Any unit test criteria fails or Network Server is down
Uploading a Car Build	User is logged into a registered account and has vehicle builds saved onto account	No vehicles are existent on the account

Community Board Integration Testing Cases			
Scenario	**Pass**	**Fail**	**Reason for Failure**
Logged in users are able to access all community boards and Network server is up	X		
Posts are saved to RDBMS when all front end logic tests are passed	X		
Former Vehicle builds are able to be added to a post	X		
Logged out users are able to access all community boards		X	Logged out users will only be able to view general feed based on logged out status
Posts are saved to RDBMS when unit tests fail		X	This will cause the database to either be unorganized or the system will simply not save the posts due to internal errors/failures
User wants to add a vehicle to a post but never went through car builder		X	All cars that go through the car builder process can be added to a build. If no car was built then there

			cannot be an upload.

3.5 Part Flagging

Part Flagging will be a *MODERATE* point of testing for the web application as it will ensure that users will be able to evaluate whether a brand specific car part is compatible or not recommended with a brand specific vehicle. This service will be offered to all users who use car builder as flagging is dependent on the car builder feature.

Part Flagging - Integration Testing:

Part flagging is a feature that is dependent on other features hence the reason why we will only use integration testing with part flagging. Since, part flagging will be used to help ensure assistance to users who are trying to modify their vehicle of their choice which part flagging is used to notify if a part is compatible or not recommended. Part flagging will be both dependent on the information received from users and the information provided by manufacturing details via our vehicle APIs.

The following cases and criteria will be used to satisfy integration testing:

Part Flagging Integration Testing (Pass/Fail) Criteria		
	Pass Criteria (Valid Entries)	**Fail Criteria (Invalid Entries)**
Selecting a compatible brand specific car part	User is able to select a brand specific compatible part and is able to add the part to the build	User is not able to add a brand specific part due to UI/UX errors or website is down due to Network Servers
Selecting an non-recommended brand specific car part	User is prompted ‘Are you sure you want to add this part’ which can cause compatibility errors amongst other vehicle parts	No prompt message and part is just added to the vehicle build

Part Flagging Integration Testing Cases			
Scenario	**Pass**	**Fail**	**Reason for Failure**
User vehicle is a Honda Civic 2021 and wants to add a OEM Spoiler made for a Honda Civic 2021	X		
User vehicle is a Honda Civic 2021 and wants to add an Aftermarket Spoiler made for a Nissan Altima 2020 and is prompted for reassurance of adding that part.	X		
User vehicle is a Honda Civic 2021 and wants to add a OEM Spoiler made for a Honda Civic 2021 but, user is not able to add part		X	Fails because there is a UI/UX error that does not allow the user to add the part itself to car builder
User vehicle is a Honda Civic 2021 and wants to add a OEM Spoiler made for a Honda Civic 2021 but, user is not able to add any new parts because of network issues		X	Network servers need to be running in order for users to continue using car building to determine if a Specific car part is compatible or not.
User vehicle is a Honda Civic 2021 and wants to add an Aftermarket Spoiler made for a Nissan Altima 2020 and is NOT prompted for reassurance of adding that part.		X	Reassurance messages must pop up for user acceptance, this case fails because the part is simply added to the vehicle without a user knowing the risks involved of adding such part.

3.6 Event List

Event List will be a *MODERATE* point of testing for the web application as it will ensure that event account users will be able to post new events or for regular users to view event posts. This service will be offered to all users and are only able to register based on their logged in status.

Event List - Unit Testing:

For unit testing on the event list feature of MotoMoto, each event post will be tested to ensure the length of a posts title, description, subscribing to a post, and upvoting is functional. Unit Testing tested to ensure that each individual component of the event list in the UI/UX is able to work without breaking the application.

Event List unit testing will be based on the following criteria:

Event List Unit Testing (Pass/Fail) Criteria		
	Pass Criteria (Valid Entries)	**Fail Criteria (Invalid Entries)**
Event Post Title	Event titles are in the range between a minimum of 5 characters to a maximum of 100 characters. Users must also be a registered event account.	Event post titles are less than the minimum requirement of 5 characters or title is greater than the maximum characters of 100 characters. Users who are not event accounts should not be able to post events.
Event Post Description	Post descriptions must remain within the range of 1 character to 1500 characters. Users must also be a registered event account.	Post descriptions without a description or if the maximum limit of characters are used. Users who are not event accounts should not be able to post events.
Event Post Address	Post Address must be in the range of 25 characters to 300 characters. Zip code must also be entered for filtering locations	Posts that do not include zip codes or exceed the character limit of address.
Event Post Date	Date must be greater than the current date in the US calendar. Users must also be a registered event account.	Date is attempted to be posted for a day in the past. Also, users who are not event accounts should not be able to post events.
User Upvote	Registered users should	Users are able to upvote

	only be able to upvote an individual event post once. Users must also be a registered account.	an individual event post more than once or users who are not registered account users should not be able to post events.
Event Post Subscription	Registered users will only be able to subscribe to get notifications once per event post and must have a registered account	If a user is able to subscribe to a specific post more than once, this should fail and all registered account users should not be able to post events.
Event Post Poster Upload	Poster uploads will be converted to .jpeg and do not exceed 1 megabyte in size. All event accounts must be registered as an event.	Poster upload exceeds 1 megabyte in .jpeg file. Registered accounts that are not event accounts should not be able to post events.

Part Flagging Integration Testing Cases			
Scenario	**Pass**	**Fail**	**Reason for Failure**
User is logged into an event account and creates a new post titled ‘Fast and Loud Rides’	X		
User is logged into an event account and creates a new post titled ‘Fast’		X	This would fail as the title does not meet the requirements of 5 characters for the event title
User is logged into an event account and creates a new post titled with more than 100 characters		X	This would fail as the title exceeds the character limit of the title.
User is logged into an event account and adds a post Address that says: ‘Event Located at 1234 East Beverly Hills’ and also adds Zip Code : 90999	X		
User is logged into an event			This fails because an address is

account and adds a post Address that is empty		X	required for each event post that attempts to take place. Cannot have a real event without an address
User is logged into an event account and adds a post description that says: ‘Bring your own food and water’	X		
User is logged into an event account and adds a post description that is empty		X	Does not pass as there must be a description for users who cannot understand a poster if one is inserted in the post.
User is logged into an event account and adds a post description that is greater than 1500 characters		X	Does not pass as the description goes over the character limit and the user should not be able to post to the event page.
User is able to upvote a individual post once	X		
User is able to upvote a individual post twice		X	Regular registered users should only be able to upvote once
User is able to subscribe to a specific event once	X		
User is able to subscribe to a specific event more than once		X	Users should only be able to subscribe to a specific event once to get notification and updates on an event.
Event post poster upload does not exceed 1 Megabyte after .jpeg conversion	X		
Event post poster upload Exceeds 1 Megabyte after .jpeg conversion		X	Fails because the maximum limit for an event poster after upload is 1 megabyte, therefore violating the upload policy.

Event List Integration Testing:

Event list will be integrated with other features in the application such as the login system for user subscription and event account login for event account users. Testing will be done to ensure that the dependencies required for the event list are covered.

Event List integration testing will be based on the following criteria:

Event List Unit Testing (Pass/Fail) Criteria		
	Pass Criteria (Valid Entries)	**Fail Criteria (Invalid Entries)**
Event Postings	User must be logged into an event account to access postings	User is logged into a regular account
Event Post Uploads	Network servers must be up and running and the user must be a registered event account user.	Network servers are either down for event account users or user is logged into a regular account

Event List Integration Testing Cases			
Scenario	**Pass**	**Fail**	**Reason for Failure**
User is logged into an event account and attempts to post a new event	X		
User is logged into a regular user account and attempts to post a new event		X	Fails because regular user accounts will never have the option to post events. If so then we have a error

3.7 Part Price Analysis

Part Price Analysis will be a *LOW* point of testing for the web application as it will ensure that users will be able to evaluate prices based on the information provided via information APIs and affiliate programs. This service will be offered to all users who use the car builder feature or just want to view vehicle parts for a more detailed pricing. Part Price Analysis integration testing will be based on the following criteria:

Part Price Analysis - Integration Testing:

Part price analysis will be based on what the affiliate programs offer and the information that our APIs will be able to provide. Integration testing will ensure that the information from the programs and APIs are accessible to the users from the front end of the application.

Part Price Analysis integration testing will be based on the following criteria:

Part Price Analysis Integration Testing (Pass/Fail) Criteria		
	Pass Criteria (Valid Entries)	**Fail Criteria (Invalid Entries)**
Brand Specific Part Chart	Each price must match the price of the external link to shop from	Price is not the same as the ones listed on the linked website
Part Graph Over Time	Each vendors price for a brand specific part over the past month at a maximum of 6 months	Prices are not updated for the current month
User interaction with part price analysis	Users should be able to get details on all listed parts on the application	Application services are in maintenance or are not accessible due to application issues

Part Price Analysis Integration Testing Cases			
Scenario	**Pass**	**Fail**	**Reason for Failure**
Same part prices for a vendor on both the application and external link	X		
Different part prices for a specific vendor on the application and external link		X	Price checks are done for a brand specific part and are ensured that prices are different on the external link

3.8 Notification System

Notification System will be a *MODERATE* point of testing for the web application as it will ensure that users will be able to receive information regarding followed accounts or monthly price drops on followed part price analysis. This service will be offered to all registered users for in app notifications.

Notification System - Integration Testing:

Notification system testing will be based on the system's ability to fetch information from other users, or other details such as event updates. Integration testing will ensure that the information from the subscribed events, accounts, and community boards are sent to the users in-app inbox.

Notification system integration testing will be based on the following criteria:

Notification System Integration Testing (Pass/Fail) Criteria		
	Pass Criteria (Valid Entries)	**Fail Criteria (Invalid Entries)**
Subscribed Notifications	Users are able to see notification from account posts and events they have subscribed to.	Nothing available in the inbox after a subscribed account has recently posted or event information is updated

Notification System Integration Testing Cases			
Scenario	**Pass**	**Fail**	**Reason for Failure**
Registered user subscribes to User('BobNAlice'). BobNAlice post and registered user receives in-app notification of new post	X		
Registered user subscribes to User('BobNAlice'). BobNAlice post and registered user does receives in-app notification of new post		X	Fails because no message was sent to the user's inbox even after subscribing to the account.

Registered user subscribes to the event(‘Summer LowRiders’) and the registered user receives in-app notification of the updated date.	X		
Registered user subscribes to event(‘Summer LowRiders’) and registered user does not receive in-app notification of updated date of event		X	Fails because no message was sent to the user's inbox even after subscribing to the event.

3.9 Email Client

Email Client will be a *LOW* point of testing for the web application as it will ensure that users will be able to receive emails on two factor authentication during account creation. This service will be used to ensure that users are notified for password changes, two factor authentication, and events subscribed to by a user.

Email Client - Integration Testing:

Email integration testing will be based on if two factor authentication messages are sent to users, changes/forgot passwords, or if event updates are sent to users (i.e. an event was cancelled or changes that were made to the original post).

Email client integration testing will be based on the following criteria:

Email Client Integration Testing (Pass/Fail) Criteria		
	Pass Criteria (Valid Entries)	**Fail Criteria (Invalid Entries)**
Password Change Request	Email is sent to the user requesting a password change	Email server is down or email configuration is incorrect which a user cannot receive password change to user

Updated Subscribed Events	Email is sent to the user notifying changes to a subscribed event	Email server is down or email configuration is incorrect which a user cannot receive message of changed event
Two Factor Authentication	Email is sent to the user to authenticate a new user	Email server is down or email configuration is incorrect which a user cannot receive message for two factor authentication to create an account

Email Client Integration Testing Cases			
Scenario	**Pass**	**Fail**	**Reason for Failure**
A registered user forgets their password and requests for password change. Email is sent to user	X		
A registered user forgets their password and requests for password change. Email is not sent to user		X	Email server is either down or email configuration is incorrect which a user cannot receive password change to user
An event has changed their date and email is sent to all subscribed users notifying changes	X		
An event has changed their date and email is not sent to all subscribed user notifying changes		X	Email server is down or email configuration is incorrect which means a user cannot receive a message of a changed event.
New user requests two factor authentication to verify their email address and receives code.	X		
New users request two factor authentication to verify their email address and do not		X	Email server is down or email configuration is incorrect which means a user cannot receive

receive a verification code.			message for two factor authentication to create an account

3.10 User Accounts

User accounts will be a *CRITICAL* point of testing as user accounts are a core component of the application. Users will be able to have a profile image uploaded to their account as well as a description of their account. User accounts are dependent on the RDBMS, car builder, feeds, and comments that are posted.

User Accounts - Unit Testing:

Unit testing with user accounts will be used to ensure that profile descriptions and profile image uploads are all able to be accounted for, and subscribing once to an individual user's account. There will be different types of tests to ensure that each profile image upload meets MotoMoto's requirements.

User Accounts Unit testing will be based on the following criteria:

User Accounts Unit Testing (Pass/Fail) Criteria		
	Pass Criteria (Valid Entries)	**Fail Criteria (Invalid Entries)**
Uploading Profile Picture	Profile picture must be within 1 Megabyte after image conversion to .jpeg	Profile image is greater than 1 Megabyte after image conversion to .jpeg
Profile Description	Profile description is within the range of 0-1000 characters. Profile descriptions can be empty and is not a profile necessity	Profile description is greater than the limit of 1000 characters
Profile Subscription	Registered user subscribes to another account once	Registered user subscribes more than once to another users account

User Accounts Integration Testing Cases

Scenario	Pass	Fail	Reason for Failure
A registered user uploads a profile image that converts to less than 1 Megabyte after .jpeg conversion	X		
A registered user uploads a profile image that converts to greater than 1 Megabyte after .jpeg conversion		X	Failure because the image must meet the requirements of 1 MB conversion to .jpeg if not a error message is thrown to the user
A register user does not post anything in their description	X		
A registered user adds to their description : ‘I am speed’	X		
A registered copies and pastes the great gatsby into their description		X	Failure due to the number of characters exceeding the limit of 1000 characters
A registered user subscribes to another user's account once	X		
A registered user subscribes on another user’s account 10 times		X	Failure due to corrupted nodes or invlaid implementation. Users should only be able to subscribe to a specific account one time.

User Accounts - Integration Testing:

User account integration testing will be used to ensure that all posts and vehicle builds are accessible through an account. The whole purpose of integration testing will be to ensure that all user account dependencies are able to communicate with each other.

User Accounts Integration testing will be based on the following criteria:

User Accounts Integration Testing (Pass/Fail) Criteria		
	Pass Criteria (Valid Entries)	**Fail Criteria (Invalid Entries)**

All Community Board Posts are located on an account	All user posts must be available on the users account	All posts are not on the user account. This could be due to bad implementation or the application is down
Saved Vehicle Builds are located on account	All registered user saved vehicle builds are located on the users account	All vehicle builds are not located on the users account.

User Accounts Integration Testing Cases			
Scenario	**Pass**	**Fail**	**Reason for Failure**
When a user saves a vehicle build, it is also located on the registered users profile	X		
When a user saves a vehicle build, it is also NOT located on the registered users profile		X	This would fail because all vehicle builds must be placed on a users account
When a user creates a new post, the new post is added onto the users profile page	X		
When a user creates a new post, the new post is NOT added onto the users profile page		X	This would fail because all new community board posts are highlighted onto the users account

3.11 Event Accounts

Event accounts will be a *CRITICAL* point of testing as user accounts are a core component of the application. Event accounts will be able to have a profile image like user accounts as well as a description of the account. Event accounts are dependent on the RDBMS and are more focused on posting events although an event account has the functionalities of a user account.

Event Accounts - Unit Testing:

Unit testing with event accounts will be used to ensure that profile descriptions and profile image uploads are all able to be accounted for, and subscribing once to an individual user's account. There will be different types of tests to ensure that each event upload meets MotoMoto's requirements.

Event Accounts Unit testing will be based on the following criteria:

Event Accounts Unit Testing (Pass/Fail) Criteria		
	Pass Criteria (Valid Entries)	**Fail Criteria (Invalid Entries)**
Uploading Profile Picture	Profile picture must be within 1 Megabyte after image conversion to .jpeg	Profile image is greater than 1 Megabyte after image conversion to .jpeg
Profile Description	Profile description is within the range of 0-1000 characters. Profile descriptions can be empty and is not a profile necessity	Profile description is greater than the limit of 1000 characters
Profile Subscription	Registered user subscribes to another account once	Registered user subscribes more than once to another users account

Event Accounts Unit Testing Cases			
Scenario	**Pass**	**Fail**	**Reason for Failure**
A event account uploads a profile image that converts to less than 1 Megabyte after .jpeg conversion	X		
A event account uploads a profile image that converts to greater than 1 Megabyte after .jpeg conversion		X	Failure because the image must meet the requirements of 1 MB conversion to .jpeg if not a error message is thrown to the user

A event account does not post anything in their description	X		
A event account adds to their description : ‘I am speed’	X		
A event account copies and pastes the great gatsby into their description		X	Failure due to the number of characters exceeding the limit of 1000 characters
A event account subscribes to another user's account once	X		
A event account subscribes on another user’s account 10 times		X	Failure due to corrupted nodes or invlaid implementation. Users should only be able to subscribe to a specific account one time.

Event Accounts - Integration Testing:

User account integration testing will be used to ensure that all posts, comments, and vehicle builds are accessible through an account. The whole purpose of integration testing will be to ensure that all user account dependencies are able to communicate with each other.

Event Accounts Integration Testing (Pass/Fail) Criteria		
	Pass Criteria (Valid Entries)	**Fail Criteria (Invalid Entries)**
All Community Board Posts are located on an account	All user posts must be available on the users account	All posts are not on the user account. This could be due to bad implementation or the application is down

Saved Vehicle Builds are located on account	All registered user saved vehicle builds are located on the users account	All vehicle builds are not located on the users account.

Event Accounts Integration Testing Cases			
Scenario	**Pass**	**Fail**	**Reason for Failure**
When a user creates a new post, the new event post is added onto the event profile page	X		
When a user creates a new event post, the new event post is NOT added onto the events profile page		X	This would fail because all new event posts are highlighted onto the users account

3.12 System Network

System Network will be a *CRITICAL* point of testing as stored information will be a priority before the deployment of our application to safeguard users from hackers. Since we plan on storing user information such as emails, we want to ensure that our users are safe whenever they are using our application. Providing network security is one of our main goals of creating software where users will not have to worry if their information is safe.

System Network - Network Security:

Network security tests will be used to ensure that user information is not forcefully taken from our RDBMS. The goal of network security tests is to attempt to break into any vulnerabilities uncovered by our software. By using blackbox and whitebox testing, our plan as a development team is to ensure user safety.

Network Security testing will be based on the following criteria:

System Network Network Security Testing (Pass/Fail) Criteria		
	Pass Criteria (Valid Entries)	**Fail Criteria (Invalid Entries)**

Vulnerability Scanning	No network vulnerabilities are found and network is safeguarded	Vulnerabilities found in the network which will need to be penetration tested to identify the severity.
Penetration Testing	Vulnerability is impenetrable and firewall security does not allow for breach to take place	Penetration testing breaks through network firewalls and is able to access internal logic.