CS691 – Computer Science, Spring 2021

Pace University



SYSTEM TEST PLAN

Carriage

Authors: Vihan Parmar, Mady Dembele, George Lopez, Roneil Boodram, Lang Gong, Tianqi Han, Maisha Masnoon, Jevon Cowell

Project Manager: Vihan Parmar

Date: 4/13/2021

**Table of Contents**

[INTRODUCTION 3](#_30j0zll)

[TESTING SCOPE 3](#_3znysh7)

[TESTING OBJECTIVES 3](#_tyjcwt)

[Features to be Tested 3](#_1t3h5sf)

[Features not to be Tested 4](#_2s8eyo1)

[TEST PROCESS DEFINITION 4](#_17dp8vu)

[Test Process Phases and Tasks 4](#_26in1rg)

[Deliverables 5](#_35nkun2)

[APPROACH TO SYSTEM TESTING 5](#_1ksv4uv)

[Approach to Functional Testing 5](#_2jxsxqh)

[ENTRY/EXIT CRITERIA 6](#_3j2qqm3)

[Entry Criteria 6](#_4i7ojhp)

[Exit Criteria 6](#_2xcytpi)

[ENVIRONMENTAL NEEDS 6](#_3whwml4)

[ROLES AND RESPONSIBILITIES 7](#_2bn6wsx)

[TEST CYCLES AND SCHEDULE 8](#_3as4poj)

[RISKS AND CONTINGENCIES 8](#_49x2ik5)

# INTRODUCTION

This document describes the System Test Plan that provides a common understanding among the “Carriage” application project stakeholders on the scope, objectives, and approach to performing the system testing. Also, the document explains the features to be tested and features not to be tested, testing entry/exit criteria, environmental needs, resources, roles and responsibilities, testing schedule, risks/contingencies and approvals.

# TESTING SCOPE

The testing scope includes two perspectives - the functional scope and technical scope.

The functional scope includes the following modules of the “Carriage” system: the End-User Account and Shopping Cart.

The technical scope includes the following architectural components:

* Android/ios devices
* Firewall
* Application server
* Database server
* File storage

# TESTING OBJECTIVES

The primary focus of this System Test Plan is functional testing with the objective to evaluate the system implementation stability.

The basis for developing functional tests and evaluating the system functionality includes the following sources:

* Business Requirements Document (BRD)
* User Stories (functional requirements)
* Requirements Composition Table (supplementary requirements)

## Features to be Tested

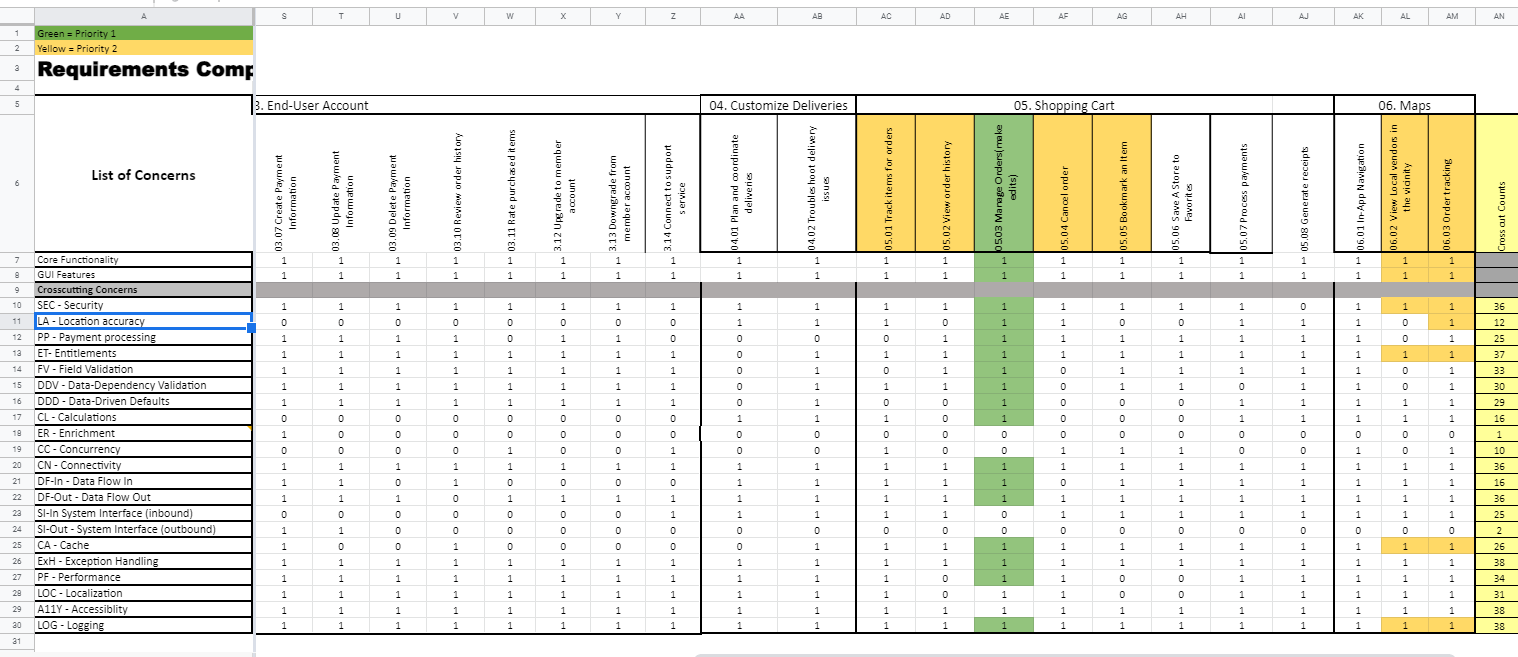
This section lists all core features that will be tested grouped by the application modules below.

User Experience

* Create an account
  + To test whether the user can create an account.
  + Applicable crosscut concerns for this core feature are, Security, Entitlements,Field Validation, Data-Driven Defaults.
* Sign Up
  + To test if the user can sign up if they don’t have an account.
  + Applicable crosscut concerns for this core feature are, Security, Location accuracy, Entitlements, Field Validation, Data-Driven Defaults, Data Flow Out, Exception Handling, Logging.
* Sign In
  + To test whether a user can sign in once they have created an account.
  + Applicable crosscut concerns for this core feature are, Security, Field Validation, Data-Driven Defaults, Data Flow Out.
* Sign Out
  + To test whether a user who has an account can sign out after signing in.
  + Applicable crosscut concerns for this core feature are, Accessibility, Logging.
* Manage Orders
  + To test whether the user is successfully able to delete an item from the cart or add an item to the cart that they have. We will also test if the user can successfully delete the order.
  + Applicable crosscut concerns for this core feature are, Security, Location accuracy, Payment processing, Entitlements, Field Validation, Data-Dependency Validation, Data-Driven Defaults, Calculation, Connectivity, Data Flow In, Data Flow Out, Cache, Exception Handling, Performance, Logging.

Besides the core features in the scope of testing, the function testing also will cover crosscutting concerns that are applicable to the context of the individual core features.

# 



## Features not to be Tested

This section lists all core features that will not be tested. These core features will not be implemented at this stage, hence these features are not eligible for testing. The core features not to be tested are,

* Create banners
* Update banners
* Remove banners
* Create Coupons
* Update Coupons
* Remove Coupons
* Promote new vendors
* Troubleshoot technical issues
* Generate open tickets
* Generate close tickets
* Communicate with customers
* Update an account
* Delete an account
* Create Payment Information
* Update Payment Information
* Delete Payment Information
* Review order history
* Rate purchased items
* Upgrade to member account
* Downgrade from member account
* Connect to support service
* Plan and coordinate deliveries
* Troubleshoot delivery issues
* Track items for orders
* View order history
* Cancel order
* Bookmark an Item
* Save A Store to Favorites
* Process payments
* Generate receipts
* In-App Navigation
* View Local vendors in the vicinity
* Order tracking

# 

The non-functional features will not be tested due to the lack of testing tools.The non-functional requirements are listed below,

|  |  |
| --- | --- |
| **Category** | **Requirements** |
| **Usability** | The application GUI will provide a user-friendly intuitive design with all the features clearly displayed for the user |
| **Usability** | The application navigation will be self-explanatory by clear and concise descriptions and names of each section, as well as features will be clearly evident by proper location and naming |
| **Usability** | Accessibility will be supported for the disabled users as well |
| **Performance** | The application will be supported on different operating systems and browsers, and should not impact the user’s system capabilities |
| **Performance** | The application will be available for 24/7 without any interruptions, and regular maintenances will be scheduled to support the application |
| **Performance** | The application will support the concurrency where the users will be able to simultaneously browse the app, login & subscribe, make payments, shop the products |
| **Performance** | The application will have short response time to all requests and all the features should be available and not impacted by latency |
| **Security** | The application will be using the automated daily & weekly audits to detect the vulnerabilities. |
| **Security** | The application will use HTTPS protocols for any data exchanges, enforced TLS for all the email communications, and other encryptions that will be maintained on the server |
| **Database** | The application will be using MongoDBB |
| **External System** | The application will be able to interface with the external data feeds from and to payment merchants, ad agencies, social media websites, etc… |

# TEST PROCESS DEFINITION

## Test Process Phases and Tasks

The test process consists of the following five phases:

* Test Planning
* Test Design
* Test Preparation
* Test Execution
* Test Reporting

Test Planning defines **scope** and **objectives** of testing, **roles** and **responsibilities** of those doing the test, and the **approach** **to testing**.

Test Design identifies **test ideas,** defines an **approach to designing test cases**, develops **test case specifications**, measures **test coverage**, and determines **requirements for test data**.

Test Preparation setups up the test **environment**, provision **test data**, and installs the software in the test environment.

Test Execution executes **all test cases**, finds and reports **software defects**, evaluates the **system stability**, and validates all **target features**.

Test Reporting summarizes and reports the **test execution results**, reports **defect metrics**, evaluates the **test exit criteria**, creates a **test completion report**, submits for **stakeholder approval**, and obtains stakeholder signoff on **system testing**.

## 

## Deliverables

On this project, the test process deliverables include:

* System Test Plan Document
* Test Design Specification
* Test Case Specification
* Test Execution Logs
* Test Reports

# APPROACH TO SYSTEM TESTING

The overall approach to System testing will be based on the Black-box method:

* Test execution will be conducted manually, from the user perspective and based on formal test case specifications.
* Testing will examine functionality of an application without knowledge of its internal structure or workings.
* Testing will ignore internal mechanisms of systems or components and focus solely on the outputs generated in response to selected inputs and execution conditions.

The test execution results will be captured and reported in test execution logs.

Approach to Functional Testing

The Functional Testing will be performed based on the black-box techniques. This means that the external functional specifications or business requirements will be used as a primary source to design test conditions. Secondly, testing will be executed from the user perspective, i.e., considering the system as a black box and entering input data and evaluating results via the user interface.

The system features identified above can be classified by the following types of cross cut concerns – GUI features, Core Functionality, Field Validation, Data Flow In, Data Flow Out, Entitlements and many more. Each item can have its own test case that can be reused across the system. Test conditions can be designed using conventional techniques, such as boundary analysis, equivalence partitioning, decision tables, etc. The detailed test logic for each pattern of business rules will be described in the test design specification document.

# ENTRY/EXIT CRITERIA

This section defines both Entry and Exit Criteria for test execution and is intended to establish a common understanding about the conditions when the test execution can start and when it can stop.

## Entry Criteria

The test Entry Criteria include the following items:

* The application build is produced and deployed to the test environment
* The system test plan is produced and approved
* The test environment is ready for testing
* Test Designs and test case specifications are completed

## Exit Criteria

The test Exit Criteria include the following items:

* All test cases have been executed
* No major software defects will be open.
* Open defects of medium/low severity have known work-arounds / solutions
* Test Summary report is produced and published

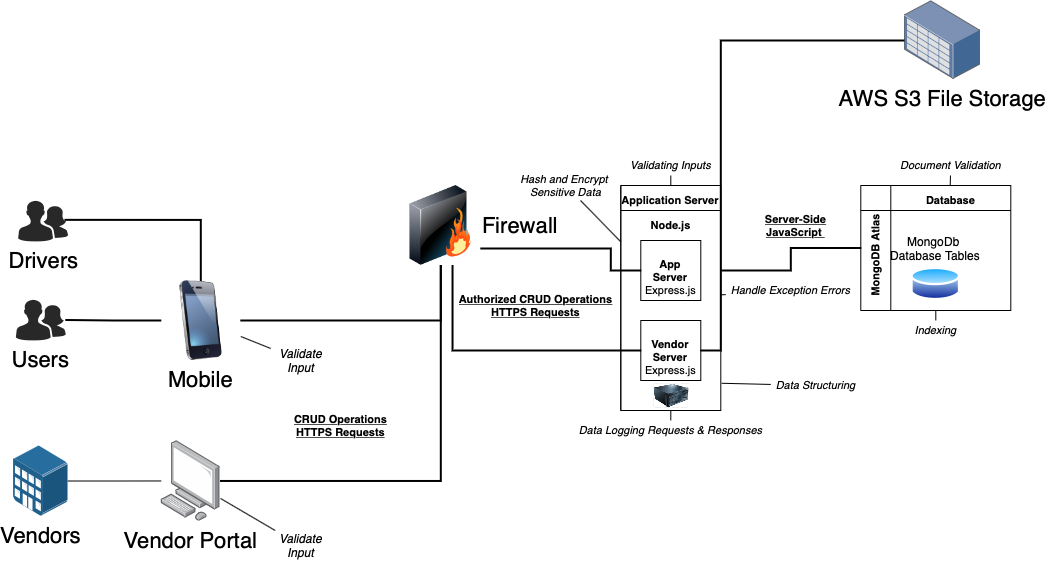
# 

# 

# ENVIRONMENTAL NEEDS

The Test Environment should be available to start test execution. node js, android/ios device, react native The architecture of the test environment is shown below.

**Carriage Application  
Architecture Type: Application Architecture  
View: Process View  
Style: Client-Server N-Tier Architecture Pattern**



# ROLES AND RESPONSIBILITIES

The project team has seven members that are assigned various project roles including Project Manager, Product Owner, Lead Business Analyst, Lead Developer, DBA, Lead QA Analyst. Their responsibilities are defined in the table below.

|  |  |
| --- | --- |
| **Project Role** | **Role Responsibilities** |
| Project Manager | Reviewing and approving the System Test Plan, test design specifications.  Managing the test environment preparation.  Tracking the testing schedule and results. |
| Lead QA Analyst | Designing a test plan, establishing a test repository, developing test case specifications, executing testing and reporting defects. |
| Product Owner | Contributing to the test plan and test case specifications. Reviewing test results. |
| Lead Business Analyst | Contributing to the test plan and test case specifications. Reviewing test results. |
| Lead Developer | Establishing and maintaining the test environment, assisting a Lead QA Analyst throughout the testing process. |
| DBA | Assisting the Lead Developer in establishing and maintaining the test environment. |
| Tester | Testing the application scope keeping various core features and their related cross-cuts in mind. |

# TEST CYCLES AND SCHEDULE

The system test execution will be conducted as three test cycles that are aligned with three application modules as follows:

Cycle 1. End-User Account Module1

* This cycle concentrates on testing the first part (create user account and sign up) of the End-User Account Module

Cycle 2. End-User Account Module2

* This cycle concentrates on testing the second part (sign in and sign out) of the End-User Account Module.

Cycle 3. Shopping Cart Module

* This cycle concentrates on testing the third part (add / delete items to / from Shopping Cart) of the Shopping Cart Module.

See the schedule of the test execution cycles in the project plan.

RISKS AND CONTINGENCIES

The following is a list of potential risk and contingencies that may be encountered during the testing of our application.

* Coding defects within the application may result in a delay in testing.
* Miscommunication between team members can lead to lack of collaboration.
* Needing to change previous diagrams/documents as testing issues are identified could lead to delays.
* Not having appropriate software and hardware resources could be a significant risk to testing.
* Excessive hours needed in the testing procedure could lead to poor team morale.
* Any changes in the overall vision and goals of the application during testing could cause further delays.
* Inaccurate estimations of how application will function as a completed product will cause additional time and effort to correct.
* Inaccurate documentation of testing can lead to lost data that would be challenging to recover.

APPROVALS

The following is a list of who can approve the process as complete and allow the project to proceed to the next level (depending on the level of the plan).

* The master test plan will need to receive approval from all team members.
* The Manage Orders test unit will need to receive approval from Lead Business Analysts, as it plays a significant role in the overall workflow/business of the product.
* Test units for Create Account, Sign Up, Sign In and Sign Out will need approval from DBA, as information entered in the respective data fields will be stored in the database.
* Any significant changes made to GUI will need approval from Lead Developer, as they will need to implement the changes via respect programming language.