# Software Test Plan for CSULA No Ticket For Me! (Group 7)

1. Introduction

1.1. Purpose

This document outlines the software testing methodologies and approaches to ensure the functionality, integration, and performance of the “CSULA No Ticket For ME” website, a web-based platform for CalState LA parking management. The objective is to verify that all critical features such as Event Parking Purchase, Visitor Parking Purchase, Non-Affiliated Parking Purchase, High School Permit Purchase, 2-Wheel Vehicle Purchase, U-Pass Purchase, Standart Permit Purchase, Account Fee View, Ticket Payment, Ticket Appeal, and Vehicle Registration all as intended and are implemented securely.

1.2. Scope

The testing will ensure all functionalities of the original CalStateLA Parking & Transportation website along with added functionality in the form of third party websites, including CRUD operations for permit renewals and/or purchases, vehicle management, database reliability.

1.3. References

CalStateLA Parking & Transportation

**2. Test Strategy**

2.1. Objective

To recreate and improve the CalStateLA Parking & Transportation website to create a better and smoother experience for students & faculty alike who wish to purchase a registered vehicle and purchase a permit, or pay off a parking ticket. This will also include the implementation of the manual test to detect functional and nonfunctional defects by identifying defects and discrepancies in the system.

2.2. Approach

For the approach, the tests will fit into two categories, functionality and effectiveness, and user feedback. Test plans will be devised and created during testing utilizing developers' intuition and manual tests will be used when the user inputs incorrect data or the system has hit its limit. Furthermore, test plans will also be devised to see how users interact with the tool in order to dictate front end design.

2.3. Tools

* Python for the core functionality
* SQL for the database
* HTML & CSS for the front end website
* Flask for web framework
* Stripe API for payment processor

2.4. Environments

Testing will be conducted in Firefox, Chrome, and Safari to ensure compatibility for all three platforms.

2.5. Entry and Exit Criteria

Entry:

* Test environment created.
* Necessary testing tools configured.
* Any defects identified.
* Test data made available.

Exit:

* All test cases executed.
* All critical defects are closed.
* Test summary report reviewed and approved.

3. Scope of Testing

3.1 Functional Testing

3.1.1 Unit Testing:

Developers perform unit testing on individual components of an application in isolation to ensure they work as intended. This occurs continuously during each sprint.

3.1.2 Integrated Unit Testing:

Developers collaborate to verify the interactions and interfaces between integrated components to ensure they work together correctly. This will be performed as each component from a story is integrated with other components, towards the mid to end of each sprint after

individual unit tests.

3.1.3 System Integration Testing:

The development team verifies that the entire system, comprising multiple components, functions correctly and meets the specified requirements. This is conducted as completed features are integrated, towards the end of a planning increment.

3.1.4 User Acceptance Testing (UAT):

The product owner identifies a group of novice and expert stamp collectors who will perform UAT. This determines whether the system meets the set requirements and expectations. This is performed after system integration testing before a release.

3.1.5 Regression Testing:

As features are added or modified, testers ensure that recent changes have not negatively affected existing features. The regression test set will run as part of an automated Continuous Integration/Continuous Delivery (CI/CD) pipeline.

3.1.6 Smoke Testing:

Testers conduct a preliminary test to check the main functionalities of an application. This is performed in the test environment at the beginning of each testing phase and in the production environment after each software release.

3.2 Non-Functional Testing

3.2.1 Coding Standard Testing:

Developers run these tests as they code to ensure adherence to company coding guidelines and standards. Use SonarQube as part of the CI/CD pipeline.

3.2.2 Security Testing:

Developers run these tests as they code to identify, fix, and prevent vulnerabilities. Use SonarQube as part of the CI/CD pipeline.

3.2.3 Usability Testing:

UX/UI designers assess the user interface against the company's UX standards. This occurs once features are developed and visually integrated with the mid to late stages of a sprint.

3.2.4 Performance and Load Testing:

Performance engineers will test basic performance during each sprint, while major load tests are scheduled once per release. Verify that the system responds to user action within 2 seconds. Verify that the site maintains performance with up to 2,000 users accessing concurrently

4. Consideration of Infrastructure

4.1. Server Configuration

Tests will be conducted on a test web hosting environment with 48 GB RAM and 14 vCPUs. We'll test if the server can handle increased traffic without slowing down.

4.2 Database

Tests will be conducted on a test standard database with regular backups and rollback capabilities.

5. Risks or Mitigation Plan

5.1. Risks

* Integration discrepancies with the US California DMV.
* Data loss during CRUD operations
* Delays due to server downtime
* Downtime due to influx of traffic
* Delays due to communication

5.2. Mitigation

* Periodic data validation checks with the reference repository
* Regular database backups
* Potential backup server for higher availability
* Server hosting with adequate capacity
* Maintain open communication channels

6. Resourcing

6.1. Team Composition

* 8 Developers

7. Milestones and Deliverables

7.1 Milestones:

Sprint 24.0

* Sign up
* Login

Sprint 24.0 - 24.1

* Account Fees
* Transactions Fees
* Permit Purchasing
* Ticket Payment
* Appeal's
* Event Parking Permit
* Visitor parking
* 2 Wheel parking Permit

Sprint 24.1 - 24.2

* Vehicle Management (JL)
* Shop Cart
* Event Parking Ui
* Stripe
* Vehicle Registration
* Admin functionality

7.2. Deliverables

* Allow users to purchase a variety of parking permits, allowing parking on school premises without issues or delays.
* Allow users to dispute parking permits that may have been distributed by the school on campus grounds without issues or delays.
* Allow users to register any 2 or 4 wheel non-commercial vehicle to park on campus grounds without issues or delays.
* Allow users to purchase U-Pass, gaining access to the school public transit system without issues or delays.
* The system responds to user action within 2 seconds. Verify that the site maintains performance with up to 2,000 users accessing concurrently.
* A detailed report on testing results which will include performances, security, and usability findings