**PARKING CALCULATOR – TEST PLAN**

1. Introduction

* 1. Test Objectives
  2. Scope of Testing
     1. In-Scope
     2. Out-Of-Scope
  3. System Overview
  4. Definitions/Acronyms
  5. References

1. Approach
   1. Assumptions/Constraints
   2. Coverage
   3. Test Tools
   4. Test Types
   5. Test Data
2. Plan
   1. Test Team
   2. Major Tasks and Deliverables
   3. Environmental Needs
3. Features to be Tested
4. Features Not to be Tested
5. Testing procedures
   1. Test Execution
   2. Pass/Fail Criteria

**1. Introduction**

This Test Plan emphasize the plan, scope, and approach for testing the Parking Calculator application. The goal of this document is to ensure that all functionalities of the Parking Calculator are tested and the final products meet the requirements.

**1.1 Test Objectives**

Main Objective of this testing is to verify that Parking Calculator run as a whole and functions as expected across several parking categories (e.g., Valet Parking, Lot Parking, Garage Parking), including:

* Accurate calculation of parking costs based on user input.
* Validate the system when handling edge cases, such as one-minute durations and multiple-day calculations.
* Documenting any defects or issues encountered during testing.

**1.2 Scope of Testing**

**1.2.1 In-Scope**

* Functional testing of Parking Calculator
* Validation of Correct cost Calculation based on user inputs
* Testing edge cases such as one minute duration and zero-minute parking duration
* Cross-browser testing to make sure consistent performance across different browser
* Regression testing to ensure present changes does not negatively impact previous existing functionality.

**1.2.1 Out-Of-Scope**

* Performance and load testing, as the application is a small-scale project intended for testing practice.
* Integration with external systems such as third-party services.

**1.3 System Overview**

This Parking Calculator is a web-based website designed to calculate parking cost based on selected parking lot and duration of the stay of the customer. Users can select from various parking options, including Valet Parking, Hourly Parking, Garage Parking, Surface Parking, and Economy Lot Parking, which then this web will calculate the total cost based on the selected parking lot and the duration of the parking.

**1.4 Definitions/Acronyms**

* Jira: A project management tool used for tracking issues and managing agile projects.
* Zephyr: A test management tool integrated with Jira for creating, managing, and deleting test cases.

**1.5 References**

* Project Requirements document.
* User stories and test cases documented in Jira and Zephyr.

**2. Approach**

This section contains the testing approach, such as constraint and assumption, coverage, test tool, test types, and test data.

**2.1 Assumption/Constraints**

* Assumptions:
  + Application is stable to undergo functional testing without frequent crashes.
  + Test environment is set up and mirrors the production environment closely.
  + All necessity user stories and requirements are documented and available for references.
* Constraints:
  + Scope of Testing may be limited by time and resource available, since it is focus on functionality and regression testing
  + It is a manual testing as automation testing are not part of the current scope.

**2.2 Coverage**

* Software Components
  + Parking Calculator Web responsible for calculating parking cost with the user input
  + UI allows user to choose parking types and entry exit times.
* Requirements
  + This website must calculate parking cost accurately according to parking category and duration
  + UI must be easy, friendly, and display the cost clearly.
* Business Processes:
  + Validate that Parking Calculator meets business requirements for cost calculation.
  + Ensure application functions fully across all parking categories.

**2.3 Test Tools**

* Jira: used for tracking test cases, defects, and overall management of the project
* Zephyr Scale: App integrated with Jira for test management, used for creating, executing, and managing test cases.
* Web Browsers: Testing will be performed on several web browsers (Chrome and Firefox)
* Selenium: An automation testing tool used to perform functional and regression testing.

**2.4 Test Types**

* Functional Testing:
  + Verifying that the Parking Calculator functions as intended across the web
  + Checking the system that it handles various input scenarios like edge cases.
  + Automation Testing with Selenium to ensure all functionalities work for repetitive tasks and regression testing.
* Regression Testing:
  + Ensuring new code do not negatively impact the previous existing code
* Cross-Browser Testing:
  + Validate Parking calculator displays and function correctly across different web browsers.

**2.5 Test Data**

* Standard Data:
  + Combinations of entry and leave time across different parking categories.
* Edge Case Data:
  + Unlikely values such as one-minute durations, same entry and exit time, and maximum durations.

**3. Plan**

This section contains the test plan, such as details on the test team, major tasks, and environmental needs

**3.1 Test Team**

The test team in Parking Calculator web consists of the following roles:

* Test Engineers/ QA Analysts: Execute test cases, defects, and perform regression testing which is responsible for detailed testing for each category and all test cases.
* Developers: Resolve defects quickly and clarify any issues related to the code or functionality.

**3.2 Major Task**

* Test Planning:
  + Develop Test Plan, focusing on the scope, approach, and resource needed for testing.
  + Document test cases in Zephyr Scale.
* Create Test Case:
  + Create Detailed test cases covering all functional tests, including edge cases and negative scenarios.
  + Develop Selenium Scripts: Create Selenium scripts for automating the test cases, making sure they can be run repeatedly using automation testing.
  + Ensure test cases are traceable.
* Test Execution:
  + Execute test cases manually and results recorded in Zephyr Scale.
  + Automate Test Execution: Run automate test scripts using Selenium especially for regression testing.
  + Perform cross-browser testing to ensure compatibility in different web browsers.
* Defect Management:
  + Track all defects during testing in Jira.
  + Collaborate with development team to resolve the bugs.
  + Retest the resolved defects to confirms the issues have been fixed.
* Test Summary Report:
  + Compile test summary report at the end of testing phase, highlighting the testing effort, including the number of test cases executed, passed, failed, and any outstanding defects.
  + Present the report to stakeholders.

**3.3 Environmental Needs**

* Test Environment:
  + A stable test environment mirroring the production environment will be set up for testing with access to all features and parking categories.
* Tools and Software:
  + Jira: Tool For defect tracking and project management.
  + Zephyr Scale: For test management, including creating, executing, tracking, and deleting test cases.
* Hardware Requirements:
  + Desktop or Laptop with enough processing power and memory to run several browsers.

**4. Features to be tested**

* Parking Cost Calculation:
  + Verify that the systems calculates parking costs correctly for each categories (Valet Parking, Lot Parking, Garage Parking, etc. ) based on user inputs for entry and leaving time.
  + Test edge cases, such as one-minute duration, same entry and leaving time, and zero-miute duration.
* User Interface:
  + Ensure UI displays the calculated parking costs clearly and dynamically.
  + Test all the dropdowns and input fields
* Cross-Browser Compatibility
  + Test the web on different web browsers (Chrome, Firefox, etc.) to ensure consistent behavior and appearance.
* Error Handling:
  + Verify that the system correctly handles invalid inputs, such as entry time is after the leaving time.

**5. Features not to be tested**

* Payment Processing:
  + The Parking Calculator does not include payment processing features.
* Integration with External Systems:
  + Any potential future integration with third-party parking services not included in this testing.
* Performance and Load Testing:
  + This project is intended for testing practice; hence, performance and load testing will not be conducted.

**6. Testing Procedures**

**6.1 Test Execution**

* Test Cases:
  + Each test case will be executed according to the defined steps documented in Jira/Zephyrs, which later the expected result will be compared with the actual result.
  + Automated test execution for repetitive and regression test cases using Selenium.
* Order of Testing:
  + Testing will begin with functional testing for each category and followed by browser testing. Moreover, Edge case will be tested later to ensure the efficiency of the Parking Calculator.
  + Automated testing will be executed periodically, especially after code changes when bugs identified.
* Bug Reporting:
  + Any defects identified during testing will be log by Jira, including with detail descriptions.

**6.2 Pass/ Fail Criteria**

* Pass Criteria:
  + A test case will be marked as passed if actual results is the same as expected result.
  + All functional requirements for parking cost calculations, UI behavior, and error handling must be met.
* Fail Criteria:
  + Test case will be marked as failed if actual results is not the same as expected result.
  + Any inconsistencies in parking cost calculations, UI behavior, and error handling will result in fail.
* Retesting:
  + Failed test cases will be retested after defects have been resolved by the development teams and the same Pass/Fail Criteria will apply.