




# Test Plan

C++ Code Streaming Project

Kendall Vargas

Programmer | Tester



November 18th, 2024.

Contents

INTRODUCTION: ..... 2

Objectives:..... 2

In Scope: ..... 2

Out of Scope: ..... 3

Test Strategy: ..... 3

Exit Criteria:..... 4

Priority List..... 5

Testing Tools:..... 5

## INTRODUCTION:

This document outlines the strategy for testing the C++ program that assists in calculating the monthly streaming consumption fee. The goal is to validate the correctness and robustness of the program to ensure it meets the intended functionality and all the calculations are performed as expected, as this is crucial for the system's operation, given that this is its primary purpose.

## Objectives:

The **objective of testing** this project is to thoroughly evaluate the entire system and identify any critical issues that may significantly negatively impact on the user experience in calculating the price based on their monthly streaming consumption.

- Validate all the functions are working as expected, and no dead code/unused variables are presented: Static Testing.
- Confirm input validation for all the fields: name, numbers, menus, characters and so on. This will include Boundary Value Analysis and negative values.
- Validate that the system does not close unexpectedly when some issue is triggered, ensuring robustness and making sure the system can handle issues without crashing.
- Validate the FE part (menus, receipt) is displayed in an organized/correct way to the user.
- Validate end-to-end scenarios for the whole program flow, considering the different categories of the price calculations.

## In Scope:

### Data Entry:

- All the data entry information requested in the menu 1.

### Receipt Generation:

- Populating the information requested in the menu 1, and all the calculations of the streaming consumption and the necessary information based on the requirements.

## Out of Scope:

### Multiple receipt generation:

- Only 1 receipt can be generated, if a new one is added in the same flow, it won't be supported.

### Database:

- This project does not include a database, due to this, there's no way to store an ID to a specific object when entering the name and the ID or matching the information with an invoice ID.

### Hours:

- Partial hours or minutes were not included in the scope.

## Test Strategy:

### 1. Timeline:

#### 1. Preparing Test Cases:

The creation of test cases for the different features and calculations will be covered by Black Box and Experienced-based techniques.

#### 2. Executing Test Cases:

- Executing the created test cases and logging the test result in the respective documentation.
- Report the defects by using the assigned testing tool for the respective fix.

#### 3. Bug Fix:

- Once all Test Cases have been executed and all issues have been reported, I will proceed with the bug fixes and push them to the **master** branch.

#### 4. Retest and Regression

- Retesting for fixed bugs will be done after the fixes are submitted. Along with a regression test depending on the section that was adjusted and some exploratory testing across the system.

## 2. Approach

### Black box:

- **Boundary Value Analysis:** Apply this technique across the inputs where numeric outputs are needed in a certain length, ensuring the boundary values are checked and confirmed.
- Calculate manually if the amount given is correct for the receipt generation.

### Static Analysis/Testing:

- Check that there's no errors on the IDE or variables not used.
- Dead code, or paths that are not reachable as a normal user.

### Experienced based techniques:

- Perform exploratory testing across the system to detect additional issues or missing information/functions/values and so on.
- Apply positive and negative scenarios across menus and all different inputs.
- Validate the correct structure of menus, receipt, and grammar issues.

### Exit Criteria:

- The system should handle unexpected inputs and errors without closing the system.
- No high-priority (blocker) issue remains in the system.
- The price calculation has the correct parameters and the respective %.

## Priority List

Priority	Priority Level	Priority Description
1	High	This bug must be fixed immediately, blocker or critical issue.
2	Medium	Incorrect functionality. There's a simple workaround for the bug.
3	Low	Grammar errors, UI issues, consideration that bug doesn't have medium/high priority.
4	Nice-to-add	Improvements that can be added to the system.

## Testing Tools:

Process	Tools
Test Case Creation	Word
Bug Reporting	Notion
Test Case Execution	VSCoDe / Manual
Codification	VSCoDe