HSC SDD Major Project Documentation

CS:GO Case Opening Simulator

Oscar Reinitz

Testing Report with Test Data

Overview

As the application is set to run locally on a user's browser, performance testing had to be done on a range of hardware which the system may encounter in its usage, which includes varying processors, network speeds and screen dimensions. Individual components were also tested on a functional level with set test data, such as the search function.

Scope of testing

- Hardware level Performance testing on Desktop, Laptop and Mobile platform
  - Varying processors, network speed and screen dimensions
- Testing of search function with set test data
- Functional testing of the case opening component

Performance testing

Desktop

Performance of the desktop hardware was sufficient when fetching and loading the item database, no issues were visible, and the process was within the expected 5 second loading time. This is to be expected as the desktop is running a high-performance processor and is connected to high-speed wired network connection.

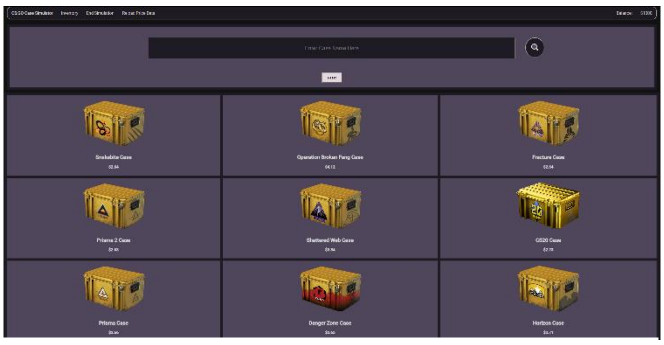

Figure 0.1

The desktop has the largest screen dimensions out of the three tested devices, properly displaying the site in both portrait and landscape (See fig 0.1 and 0.2). However, when shrinking the width of the window a point is reached where the case grid begins to stretch off the window and exist outside the confines of the CSS grid (See fig 0.3).

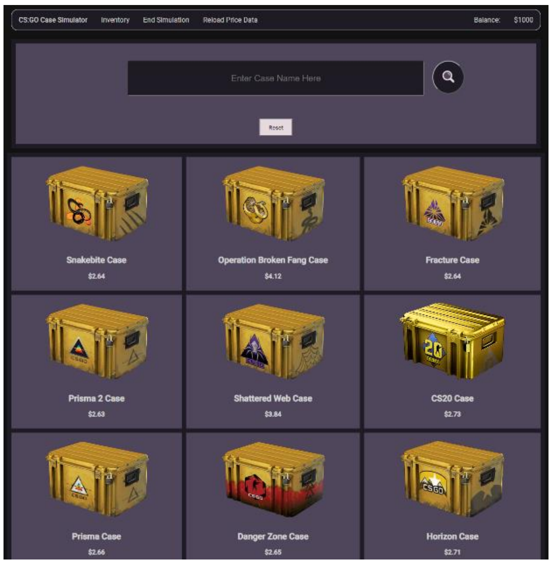

Figure 0.2

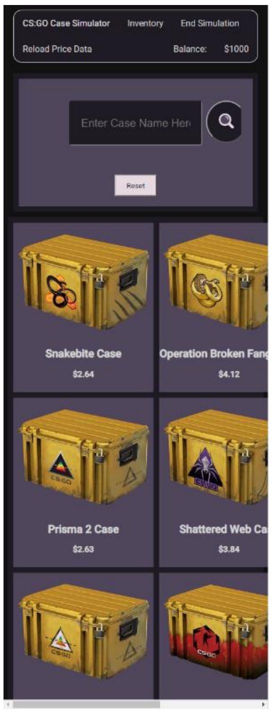

Figure 0.3

Laptop

Similarly to the desktop, the laptop performed sufficiently when fetching and loading the item database, no issues were visible, and the process was within the expected 5 second loading time. The laptop has marginally lower specs than the desktop however exists in a very similar performance class and therefore this result is to be expected. The laptop was connected to a high-speed wireless network connection yet exhibited no difference in performance to the desktop.

As the desktop and the laptop run identical operating systems and were using the same browsers, there was no difference in the screen dimension testing between the two devices, both exhibiting the same issue when the window was excessively shrunk.

Mobile

The mobile device performed adequately in comparison to the other devices, it exhibited a "micro freeze" during the loading animation, presumably when the database was loaded, the loading animation paused for a split second as the processor could not maintain both processes simultaneously. However it was still within ~1 second of the expected 5 second loading time, making the issue of little importance, and most likely attributed to the hardware itself and not a software issue, as the program was developed for a predominantly desktop based userbase with much higher specifications than a smartphone. The smartphone was connected to a low speed (2.5ghz) wireless network, however this did not appear to impede the loading time significantly enough that it was noticed in the testing.

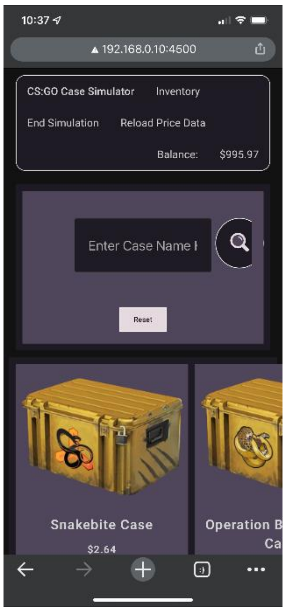

Figure 0.4

The smartphone has significantly smaller screen dimensions compared to the other two devices, as well as being a predominantly portrait focused device, whereas laptops and desktops are predominantly landscape focused. When viewing the website in portrait the text and images appear quite large, inconsistent with the standard text size on mobile phones, additionally the case grid stretches off the screen and extends over the bounds of the CSS grid (See Fig 0.4), making the website fairly unusable in portrait. In landscape the CSS grid displays correctly however each grid item is still inconsistently proportioned compared to the standard website sizing for a smartphone, appearing extremely large (See fig 0.5).

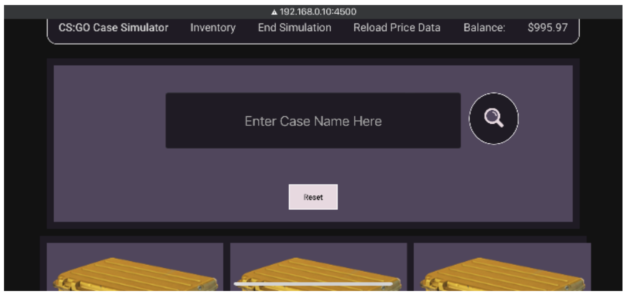

Figure 0.5

Search function testing with Test Data

Test Data Used + Results

Inputs	Explanation of Input	Expected Output	Real Output
Fracture Case	Case Name written just as it appears on the database	Fracture Case appears on site	Fracture Case appears on site
Operation Broken Fang Case	Case Name written just as it appears on the database	Operation Broken Fang Case appears on site	Operation Broken Fang Case appears on site
SNAKEBitE CaSE	Case name with a range of capitalisation on the letters	Snakebite Case appears on the site regardless of the capitalisation	Snakebite Case appears on the site
Chrome Case	Case Name spelt incorrectly	Indicate to user input is invalid via alert box	Indicates to user input is invalid via alert box
2	Input a number to determine if integers are detected as invalid	Indicate to user input is invalid via alert box	Indicates to user input is invalid via alert box
Chroma 2 Case	Case Name written just as it appears on the database	Chroma 2 Case appears on site	Indicates to user input is invalid via alert box
$%@#$	Input a symbol to determine if	Indicate to user input is invalid via alert box	Indicates to user input is invalid via alert box

	symbols are detected as invalid		
Chroma	Input word similar to name of a case to test integrity of binary search function	Indicate to user input is invalid via alert box	Indicates to user input is invalid via alert box

Explanation and analysis of results

The test data determined that the majority of cases in the item array successfully produce the expected output, oddly with the exception of the "Chroma 2 Case". Random capitalisation of case names was tested, with the binary search still able to identify when the name of the case. Strings or integers which do not occur in the array were also input and it was found the search function can identify this and indicate an invalid term is used.

In the case of the "Chroma 2 Case" tedious black box and white box testing found that when using this input, the binary search is unable to locate the string in the item array, despite its existence in the array. This fault seems to occur due to the fact a binary search is used, as existence of a "Chroma Case" and "Chroma 3 Case" breaks the component used to determine which half of the array the desired input exists in. This can be rectified through use of another search algorithm such as a basic linear or another more advanced search algorithm, however due to the requirements of the project a binary search was a necessary component to be included. This dilemma will be considered in the discussion on potential changes to be made.

Functional Testing of the case opening component

The entire program itself was functionally tested in both alpha stages by myself and in beta by one of my siblings on his local machine. The process of opening a case, receiving an item, and selling the item was thoroughly tested for each case a number of times. Each case was tested for animations, validity of the rewarded item and ability for the item to be sold.

During the alpha testing stage it was found the three "CS:GO Weapon" series of cases produced error when being opened. It was discovered that unlike every other case, these three had items which did not exist in all wear states, whereas in other cases every item can be in every wear state, this caused the algorithm to attempt to insert HTML data of an item which had no price or image as it did not exist. The solution to this error was implementing a system where the case opening function tests the won item by requesting its name in the item array, if it returns "null" the case is immediately re-rolled until a valid item is generated. This solution reduces the efficiency of the case opening function on these three cases, however in practice the impact is miniscule to a level in which a human could not realistically process, allowing this solution's implementation.

Once in beta testing it was discovered when selling some items the balance would gain a large number of decimals points as it increased. This was easily patched by implementing rounding to 2 decimal places on all values which were related to the balance or price of an item or case.

As of the alpha and beta testing phase conclusion the case opening function operates efficiently without error.

Comparison with Original Design Specification

Specification List

- Price Fetch Component
- Case Opening Component
- Statistic Logging Component
- Inventory Component
- Range of cases included on the site

Price Fetch Component

The price fetch component was altered significantly compared to the original specification after it was discovered the database could not be directly fetched via chrome due to security issues on the "CS:GO Backpack" end. Instead the database was collected into a text file stored on my GitHub and is updated periodically. The prices are not stored in browser local storage, instead they are stored in the JSON file of the database on the project GitHub page. The current price of the items is displayed based off this database, using the 7-day average price. There is no function indicating the price is out of date due to the aforementioned issues, the price simply updates inline with the database file on the project GitHub page

Case Opening Component

The case opening component was developed to the specification of the proposal. The odds for the cases match exactly to the listed in game odds, the interface resembles the original game via use of images directly off their servers for the case image and item images. When opening a case a CSS animation is played and sounds from the game are used to imitate the experience of opening a case as close as can be recreated in a browser. The selection screen for the cases is dynamic, updated as the images and prices update in game, it has a search bar for finding a desired case and the grid cells for each case react with CSS animation to a hovering cursor.

Statistic Logging Component

The statistic logging component is accurate to the specification, tracking the balance, amount spent, and amount gained. This is totalled and analysed on the page displayed when the user ends the simulation. In line with the specification the user can stop the simulation at any time to view their statistic report which is generated automatically based on these variables, displaying their loss, gain and profit.

Inventory Component

The inventory component was implemented exactly as specified in the proposal, with the ability to sell your items, as well as displaying their image and price and storing the data in the user's browser. The price that was chosen was the average over the last 7 days, as it was not feasible to update such a large database on a daily basis to have daily prices.

Range of cases included on the site

In line with the original design specifications the site includes all CS:GO cases, however due to issues with the image database the cases only go up to the "Snakebite Case" excluding two of the newer cases, this issue is attributed to the maintainers of the external database I am utilising, who have failed to update their data to match the current state of the game. As this is the only way of retrieving the image hash data they had to be excluded, in the event the database is update the code can be patched to allow the new cases to be read by the case populating function, and the site would be able to immediately integrate the new cases automatically.

Discussion on Potential Changes to be Made

CSS Styling for range of displays

To accommodate for mobile users or users on a small desktop/laptop display the CSS could potentially be modified to have differing styling depending on the users display dimensions. This is a feature within CSS however requires the entire CSS styling to be rewritten for each set of dimensions, as the site will not be deployed for a mobile userbase this was not implemented in original development. However in future there is potential to modify the CSS to implement a mobile layout for the site if demand were to exist.

Search Algorithm

As the search algorithm used (binary) is required for the project, there are two routes that could be taken in changing the program to fix the error.

One route is to completely remove the binary search as the algorithm used for searching the cases and implement a search algorithm which does not encounter the same error that currently occurs. The binary search would then have to be implemented elsewhere in the project to keep the requirements met, this would most likely require adding a binary search routine to another aspect of the project such as the inventory, requiring significant changes to be made to the site.

The second route would be to add a simple switch or if statement prior to the binary search to test for the cases which produce error, and in the case, they are input, produce the correct output without use of a binary search. This solution would be inefficient in the long term as it would detract from the "dynamic" aspects of the project where you could theoretically add or remove cases form the database and the website would be able accurately display these changes, however in a situation of urgency this would be the quickest, yet temporary, way of resolving the error.

Updating of item database

Currently the item database is not directly requested from the "CS:GO Backpack" API due to security limitation on their end. As a result the database is placed in a JSON file on the project GitHub manually and updated on a periodic basis. Potentially if the website was to be implemented on a server, utilisation of a GitHub bot could be used, this bot would be able to retrieve the database from the "CS:GO Backpack" API manually and replace the JSON file on the GitHub on a set time period. Currently as the project runs locally on user machines, this change is unable to be made, however if implementation moves to a webserver the issue of database validity could be easily modified through this potential change.

References for sites used in development

- Coolors - The super fast color palettes generator! (2022). Available at: https://coolors.co/ (Accessed: 1 August 2022).
- CS:GO Backpack API (2022) Available at: https://csgobackpack.net/api/ (Accessed: 1 August 2022).
- GitHub - jonese1234/Csgo-Case-Data (2022). Available at: https://github.com/jonese1234/Csgo-Case-Data (Accessed: 1 August 2022).
- Loading IO (2022). Available at: https://loading.io/ (Accessed 1 August 2022)
- Vectr - Free Online Vector Graphics Editor (2022). Available at: https://vectr.com/ (Accessed: 1 August 2022).