*Noah Kamphuisen, Twan Asselbergs*

*Happy Herbivore | 2WDV1*

*Compatibility Test*

03-03-2025

Table of Contents

Introduction 2

Test Environment 3

Test Scenarios 4

Findings 5

Improvement Proposals 6

Introduction

This compatibility test aims to ensure the usage of our kiosk across different platforms. We will verify that the web application functions correctly in different browsers, operating systems, and devices.

We’ll be testing on different platforms, including:

* Browsers: Google Chrome, Brave Browser, Safari, Firefox, Zen
* Operating Systems: macOS, Linux, Windows
* Devices: Laptops, Desktops

Test Environment

We have tested many platforms to the best of our ability, here are our findings:

Browsers

* **Google Chrome** works perfectly, which is also the most used browser in the world, making this option very important.
* Using the same browser engine as Google Chrome, **Brave Browser** works great too. It functions the same as Google Chrome.
* In **Safari**, the kiosk functions well too, just like in a Chrome browser.
* In **Firefox**, the kiosk works great as well.
* Building upon Firefox, we’ve tested **Zen**. Zen is a Firefox based browser, so we expected it to work just like Firefox, and it does.

Operating Systems

* As we are both using **macOS**, we have tested that first. On macOS, we have been able to get it to work almost completely. The kiosk is able to open in fullscreen mode, but making it so the kiosk opens on startup has been difficult so far. Currently, we are still working on this feature.
* My team partner was using **Linux** before recently switching to macOS, so because of that we know that the kiosk functions well on a laptop with the Linux operating system.
* Our final build will be made for a **Windows** machine, since most laptops/ desktops will be using Windows. We expect the Happy Herbivore restaurant to also use Windows as its operating system. We are still working on this feature.

Devices

* We are both using **laptops**, so testing the kiosk on our laptops was easily done. To test it, we used responsiveness functions inside a feature like Chrome DevTools. Setting the responsive display to 1080x1920, we had successfully imitated a full-on kiosk screen.
* For **desktop** screens, we have used the portrait mode to imitate a real life-size kiosk screen. This also functions as desired.

For screen resolutions we tested with 1080x1920 for laptops, and portrait mode on desktop screens.

Test Scenarios

**Display of page elements in different browsers**

In every browser that we have tested today, the kiosk functions practically the same. Of course, it is best to run the kiosk in a Chrome based browser, because that is what is widely used and will have the most up-to-date browser functionalities. Everything loads in the same way, at the same speed. The speed is mostly dependent on the speed of the server.

**Functionality of forms on different devices**

We do not really have a traditional form, but we are using features like an order summary page where you can add and delete certain items in your basket before completing the payment. This functions the same on the devices we tested today.

**Responsiveness of the interface on different screen resolutions**

Since the kiosk will not be used on dynamic screens, we built the kiosk around one static screen resolution. At the start of the project we were told the kiosk does not have to be responsive, as long as you are building the kiosk on the correct screen resolution from the start.

Findings

Summarizing the compability test results, we have found that our kiosk system works in every browser and operating system we have tested. It functions well in every browser and the load times of the server are fast. Besides that, the UI/UX is modern and clean, and uses icons with its buttons for a universal experience. It is not dynamic/responsive, but that also was not a necessary feature, since the kiosk is used for one static screen resolution.

Below is a list of any display or functionality issues we have found:

* Console language selection bug: When opening the kiosk on startup for the first time, the menu will appear as a blank page. This happens because there is a bug in our code that makes it so you have to choose either the English or Dutch language before it shows the menu page. We know the cause of this bug and are currently working on fixing it. Once you have chosen a language, it’s added to your localstorage, it works every time after doing this.
* On macOS, the kiosk functions great. There is but one problem: Making it so the kiosk starts at startup has been difficult so far, macOS can be quite difficult to work with on this matter because of its tight security. It does open in fullscreen instantly once you start it yourself.

Improvement Proposals

Any suggestions for adjustments to resolve our compatibility issues are listed on the ‘Findings’ page. As for any CSS changes to normalize the look and feel across all platforms.