*Noah Kamphuisen, Twan Asselbergs*

*Happy Herbivore | 2WDV1*

*Performance Test*

03-03-2025

Table of Contents

Introduction 2

Test Environment 3

Test Scenarios 4

Findings 5

Improvement Proposals 6

Introduction

For this document we will be creating a performance test. Performance testing is a critical process to ensure the kiosk and web applications function efficiently under various conditions. This document outlines the approach taken to measure the performance of the applications, identify bottlenecks, and optimize for better user experience.

**Purpose**

The primary goal of the performance test is to evaluate the application's responsiveness, stability, and scalability under different workloads. By conducting these tests, we aim to:

* Measure loading times and transaction speeds under normal and peak conditions.
* Identify performance bottlenecks that could impact usability.
* Ensure the application remains stable under concurrent user activity.
* Provide insights for optimization to improve efficiency.

Test Environment

To ensure accurate and reliable performance testing, a controlled test environment has been set up. This section outlines the hardware, software, and tools used during testing.

**Hardware:**

* **Device:** MacBook Air M3 (15-inch)
* Operating System Version: Sequoia 15.3.1
* **Processor:** Apple M3 chip
* **RAM:** 24GB
* **Storage:** 512GB SSD
* **Network:** Wireless Wi-Fi connection

**Software:**

* **Integrated Development Environment:** Visual Studio Code
* **Application Version:** Latest stable release of the kiosk
* **Database:** Prisma

Test Scenarios

To evaluate the performance of the kiosk and web applications, various test scenarios will be executed. These scenarios cover critical functionalities under different loads and conditions to identify potential bottlenecks.

**1. Kiosk Application Test Scenarios**

**1.1 Menu Loading Time**

Objective: Measure how long it takes for the menu and images to load.

* Test Conditions:
  + Load the menu under different network conditions (Wi-Fi with varying speeds).
  + Test different menu sizes (small, medium, large).
* Expected Outcome:
  + The menu loads within an acceptable time frame without noticeable delays.

**1.2 Order Processing Time**

Objective: Evaluate the time taken to process an order, including customizations and payment.

* Test Conditions:
  + Place orders with different item quantities and customizations.
  + Test order processing under normal and high user loads.
* Expected Outcome:
  + Orders are processed efficiently, and the response time remains consistent.

**1.3 Touchscreen Responsiveness**

Objective: Measure the app's responsiveness to touch inputs.

* Test Conditions:
  + Tap buttons rapidly to simulate heavy usage.
  + Check response time under high CPU/memory usage.
* Expected Outcome:
  + The touchscreen remains responsive without lag.

**1.4 Payment Processing Speed**

Objective: Assess the time required to complete a payment.

* Test Conditions:
  + Process payments
  + Simulate slow network conditions.
* Expected Outcome:
  + Payments are ‘processed’ within an acceptable time, and no transactions fail unexpectedly.

**1.5 Concurrent User Simulation**

Objective: Test how the application performs with multiple users placing orders simultaneously.

* Test Conditions:
  + Simulate concurrent orders using performance testing tools.
  + Monitor system resource utilization (CPU, memory, network).
* Expected Outcome:
  + The application remains stable and responsive under concurrent usage.

Findings

**1. Kiosk Application**

* **Menu Loading:** Fast on good networks
* **Order Processing:** Simple orders process fast
* **Touchscreen Responsiveness:** Slight lag under heavy load
* **Payment Processing:** Varies by provider, slower on bad networks
* **Concurrent Users:** Stable (Only 1 user at a time)

**2. Web Application**

* **Load Time:** Good under normal use, slows under high traffic
* **Load Testing:** Handles **requests**
* **Stress Testing:** Stable up to **1 user**
* **Scalability:** Not needed

Improvement Proposals

**1. Kiosk Application**

* **Menu Loading:** Compress images
* **Order Processing:** Optimize database queries & use caching.
* **Touchscreen:** Prioritize UI, optimize background tasks.
* **Payment Processing:** Optimize payment gateway API calls.
* **Concurrent Users:** Improve memory management & add load balancing.

**2. Web Application**

* **Load Time:** Use CDN & reduce script size.
* **Load Testing:** Index database & use connection pooling.
* **Stress Testing:** Scale resources & improve backend handling.
* **Scalability:** Enhance load balancing & caching strategies.