

---

# Assignment Report

SFWRENG 3S03 (2026W)

## Assignment 1

---

Author(s)

Salma Baig   baigs19@mcmaster.ca   baigs19

GitHub URL

<https://github.com/hydrowoxy/3S03-A1>

February 20, 2026

---

# 1 Task 1 - Testing with generative AI

## 1.1 a)

I asked ChatGPT 5.2 to specify three types of tests for the game described above.

The specific prompt I used was:

"Suppose I am developing an action-adventure game such that: a) players explore a post-apocalyptic version of McMaster University; b) up to eight players may be accosted by zombies, mutated professors, angry Deans, killer squirrels, and more; c) gameplay can be both competitive and cooperative; and d) the game will be released on the PS5, Xbox Series X, Nintendo Switch and Sega Genesis (which is 16-bit and hasn't been manufactured since 1997). Specify three types of tests for such a game."

The response I recieved was as follows (it originally included more emojis and markdown formatting, which I removed (and re-formatted) so that my document would compile and be legible):

"For a game like this — especially one that spans **modern consoles and a 16-bit relic like the Sega Genesis** — testing needs to cover functionality, performance, and platform constraints very carefully.

Here are **three distinct and appropriate types of testing**:

### 1. Functional Testing (Gameplay & Mechanics)

**Purpose:** Ensure the game works according to design.

This would verify:

- Players can explore the post-apocalyptic McMaster campus without falling through geometry
- Zombies, mutated professors, angry Deans, and killer squirrels spawn correctly
- Combat, health, inventory, and abilities function as intended
- Competitive and cooperative modes behave correctly
- Up to eight players connect and interact properly

Examples of functional test cases:

- If 8 players join a session, all characters spawn correctly.
- If a Dean is defeated, loot drops trigger.
- Friendly fire behaves correctly depending on mode (co-op vs competitive).

- Save/load restores correct state.

This ensures the **core gameplay logic works**.

## 2. Performance & Stress Testing (Multiplayer & AI Load)

**Purpose:** Ensure the game runs smoothly under load.

Your game supports:

- Up to 8 players
- Multiple enemy types
- Large explorable campus
- Both modern and legacy hardware

Testing would include:

- Frame rate stability (60 FPS on PS5/Xbox, acceptable performance on Switch)
- Mass zombie swarm scenarios
- Network latency under 8-player online sessions
- Memory usage limits

Especially important:

- The **Nintendo Switch** has weaker hardware.
- The **Sega Genesis** has extreme memory and CPU constraints (16-bit, limited RAM, cartridge storage).

Stress testing would reveal:

- Frame drops when 50 squirrels attack simultaneously
- Network desync in competitive mode
- AI pathfinding breakdowns under heavy load

## 3. Compatibility & Platform Testing (Cross-Console + 16-bit Constraints)

**Purpose:** Ensure the game works correctly across very different hardware environments.

You are releasing on:

- PlayStation 5
- Xbox Series X
- Nintendo Switch
- Sega Genesis

Key concerns:

Modern Consoles (PS5 / Xbox Series X)

- 4K rendering compatibility
- Ray tracing support (if implemented)
- SSD load speed optimization
- Controller haptics

Nintendo Switch

- Lower resolution scaling
- Docked vs handheld mode behavior
- Limited GPU power

Sega Genesis

- 16-bit graphics limitations
- Severe RAM restrictions
- Cartridge size constraints
- No modern multiplayer networking
- Likely redesigned gameplay mechanics

Compatibility testing ensures:

- UI scaling works at different resolutions
- Input mapping works on each controller
- Save systems comply with platform standards
- The Genesis version doesn't exceed hardware limits

Summary

The three appropriate testing types are:

1. **Functional Testing** – Does the game behave correctly?
2. **Performance & Stress Testing** – Does it run smoothly under load?
3. **Compatibility/Platform Testing** – Does it work properly across PS5, Xbox, Switch, and Genesis?

Given the extreme hardware diversity (especially including a 1990s 16-bit console), **platform compatibility testing would be especially critical** for this project."

- 
- 2 Task 2 - Testing with Junit
  - 3 Task 3 - Testing parts of large systems
  - 4 Task 4 - Test driven development (TDD)
  - 5 Task 5 - Test coverage and AI