Wonder Hall — Text Adventure Game

Sarthak Bhot · B.Sc. Computational Science, Western University · London, Ontario, Canada

1. Project Overview

Wonder Hall is a text-based adventure game set in a Wonderland-inspired world. The player explores interconnected rooms, collects three hidden treasure items, and must stash them all in the safe room — Wonder Hall — to win. Along the way the world reacts to the player's choices: hazards such as the Queen's flamingo trap in the Queen's Garden can end the game in a loss, so every command carries consequence.

The project was designed and built as a capstone-style final project, covering the full lifecycle from game design and implementation to a structured test plan validating every win, loss, and edge-case path.

2. Technology & Tools

Python | Object-Oriented Design | State Management | CLI Development | Git & GitHub | Unit Testing Concepts | Test Planning

- Python — core game engine: command parsing, room navigation, inventory, and win/loss logic.
- Object-oriented design — rooms, items, and player modelled as classes with clear state transitions.
- Git & GitHub — version control with incremental commits throughout development.
- Structured testing — a written test plan with defined cases, expected results, and pass criteria.

3. Key Features

- World exploration — multiple connected rooms (including the Queen's Garden and Wonder Hall) with descriptive narration.
- Inventory & objectives — collect three treasure items and stash them in the safe room to trigger the win condition.
- Consequence engine — risky actions (e.g. taking and using the flamingo) trigger the Queen's reaction and an immediate loss.
- Command parser — natural text input handling with feedback for invalid or impossible actions.
- Win/loss messaging — clear end-state messages once all treasures are secured or a fatal mistake is made.

4. Test Plan

Objective: verify that every gameplay path — victories, defeats, and invalid inputs — behaves exactly as designed, with no dead ends or unhandled states.

Approach: manual playthrough testing against scripted cases, covering the happy path, failure states, and boundary inputs. Each case was executed and marked pass/fail against its expected result.

ID	Test Case	Expected Result	Result
TC-01	Collect all three treasures and stash them in Wonder Hall	Win message is printed; game ends in victory	Pass
TC-02	Take the flamingo in the Queen's Garden and use it	Queen reacts; game ends in a loss	Pass
TC-03	Attempt to stash treasures before collecting all three	Win is not triggered; game continues	Pass
TC-04	Enter invalid or unrecognized commands	Helpful feedback shown; game state unchanged	Pass
TC-05	Revisit rooms and re-check inventory mid-game	Room descriptions and inventory remain consistent	Pass
TC-06	Full playthrough: explore, collect, stash, win	Complete path runs end-to-end with the win message	Pass

5. Skills Demonstrated

- Breaking a creative idea into implementable software components.
- Designing stateful logic with clear win/loss conditions.
- Writing a structured test plan and executing it methodically.
- Debugging edge cases in interactive, input-driven software.
- Using Git for version control across the development lifecycle.