Development Journal

Santa's Workshop Puzzle – Project 3

---

1. Project Overview

The goal of this project was to design and implement an interactive sliding puzzle game (15-Puzzle) with graduate-level enhancements. The application supports multiple puzzle sizes, tracks user performance (time and moves), and maintains a size-specific leaderboard using backend persistence.

The project was implemented using:

- **Frontend:** HTML, CSS, JavaScript
- **Backend:** PHP
- **Database:** MySQL

The focus of development was on correctness, usability, state management, and integration between frontend and backend systems.

---

2. Initial Design & Approach

My initial approach was to build the core sliding puzzle functionality entirely on the frontend. This included:

- Rendering a dynamic grid based on puzzle size
- Implementing tile movement logic
- Detecting win conditions
- Tracking time and move count

Once the basic gameplay was functional, I expanded the design to meet graduate-level requirements by introducing:

- User authentication
- Backend session tracking
- Persistent leaderboards filtered by puzzle size

At this stage, the project transitioned from a frontend-only game to a full-stack web application.

---

3. Challenges Faced

Puzzle State & Size Switching

One major challenge was handling puzzle size changes correctly. When switching between sizes (e.g., 3×3 to 4×4), the puzzle state, timer, and leaderboard needed to reset and synchronize properly. Initially, size changes did not always start a fresh session, causing inconsistencies in leaderboard updates.

Solution:
I refactored the game initialization logic so that every size change explicitly resets the puzzle, timer, and backend session before allowing a new run to begin.

Leaderboard Not Updating on First Completion

Another issue occurred where the leaderboard was not updating after the first successful puzzle completion, but worked correctly on subsequent completions.

Root Cause:
The backend session was being finalized before all frontend state updates completed.

Solution:
I adjusted the win handling logic so that the session is properly ended on the backend *before* triggering a leaderboard refresh, ensuring the first completion is stored correctly.

Frontend and Backend Synchronization

Coordinating real-time frontend actions (moves, timer start/stop) with backend session tracking required careful sequencing. Improper timing caused move counts or durations to be incorrect.

Solution:
I introduced clear session lifecycle boundaries:

- Session start → first move
- Move tracking → per tile movement
- Session end → on win only

This ensured accurate and reliable data persistence.

---

4. UI & UX Iterations

The user interface went through several iterations. Initial designs were functional but visually plain. I later introduced a themed design inspired by a Christmas/Santa aesthetic while ensuring readability and usability.

Key improvements included:

- Clear visual distinction between movable and empty tiles
- Consistent styling across puzzle board and leaderboard panels
- Improved feedback for user actions (win overlay, move highlights)
- Removal of distracting visual elements after usability testing

This iterative design process balanced visual appeal with clarity and performance.

---

5. Final Architecture

Frontend Responsibilities

- Puzzle rendering and tile movement logic
- Timer and move tracking

- User interaction handling
- API calls to backend endpoints

Backend Responsibilities

- User authentication
- Session creation and termination
- Move and duration persistence
- Leaderboard queries filtered by puzzle size

The separation of concerns improved maintainability and made debugging significantly easier.

---

6. What I Learned

Through this project, I gained hands-on experience with:

- Managing complex application state
- Debugging asynchronous frontend/backend interactions
- Designing REST-style backend APIs
- Persisting and querying game data effectively
- Iterative UI refinement based on user experience

Most importantly, I learned how small sequencing issues in full-stack applications can lead to subtle bugs, and how structured debugging leads to reliable solutions.

---

7. Conclusion

This project strengthened my understanding of full-stack web development and reinforced the importance of thoughtful architecture and testing. The final implementation meets all functional and graduate-level requirements while remaining extensible for future enhancements.