Software Requirements Specification

Whack-a-Prof

Version 1.1

Prepared by Team 2 – Specifications Group
CISC 3140 Project • Brooklyn College
May 8, 2025

Contents

1. Introduction

1.1. Purpose

This document specifies the requirements for the browser-based game *Whack-a-Prof*, covering functionality, user interfaces, constraints, and external interactions.

1.2. Document Conventions

The structure follows IEEE Std 830-1998 (SRS).

1.3. Intended Audience and Reading Suggestions

- **Development Team**: Chapters 2–5
- **QA Testers**: Chapters 3–5
- **Evaluators**: All chapters

1.4. Project Scope

Whack-a-Prof is an arcade-style browser game inspired by *Whack-a-Mole*. Players earn points by hitting professors as they pop up from the holes. The game was developed for CISC 3140 at Brooklyn College.

1.5. References

- IEEE SRS Standard 830-1998
- K. Wiegers, "Software Requirements," `http://karlwiegers.com`

2. Overall Description

2.1. Product Perspective

Whack-a-Prof is a standalone, client-side web application built with HTML5, JavaScript, and CSS.

2.2. Product Functions

- Start, pause, and end gameplay
- Score points by clicking characters, including professors and trustees
- Randomised character appearance
- Local-storage leaderboard (highest score)
- Customizable mallet cursor (selectable at game start)
- Special "trustee" character with unique explosion animation and bonus points when hit

2.3. User Classes and Characteristics

- **Primary**: Project evaluators / professors
- **Secondary**: QA testers
- **Tertiary**: Development team
- **End-users**: General players

2.4. Operating Environment

- *Hardware*: PC, laptop, or mobile device capable of running a modern web browser, equipped with mouse, trackpad, or touchscreen input, and audio output capability.
- *Software*: A modern web browser supporting HTML5, CSS3, and JavaScript (See Section 2.7 for specific target browsers and versions).
- *Display Requirements*:
  - **Responsive Layout**: The game utilizes a responsive design with dynamic scaling. The user interface elements, particularly the game board, dynamically adapt to the available browser viewport size.
  - **Minimum Usable Viewport**: While the layout adapts fluidly, a minimum viewport size of 375 $\times$ 667 pixels (typical older portrait smartphone) is recommended to ensure comfortable interaction and readability. Functionality on significantly smaller viewports is not guaranteed.

  - **Physical Size Limitations**: The game requires sufficient physical screen size for accurate targeting. Devices with physical displays smaller than 4 inches (diagonal) are not supported, regardless of resolution or scaling techniques. Even with responsive design, extremely small displays (such as 2 × 2 inch screens) provide inadequate target sizes for the precision required in gameplay.
  - **Pixel Density**: The application is designed to render correctly on both standard-resolution and high-DPI displays (such as Apple Retina displays).

- *Audio Requirements*:
  - **Output Device**: The system must have functional audio output capability to experience the full game.
  - **Browser Audio Support**: The browser must support the HTML5 Audio API.
  - **Note**: The game is playable without audio, but this is not the intended experience.

2.5. Design and Implementation Constraints

- Implemented entirely in JavaScript (approved libraries permitted)
- Source repository hosted on RiouxSVN, accessible at `https://svn.riouxsvn.com/whackgame`

2.6. User Documentation

- In-game interactive tutorial
- Contextual help prompts / tooltips

2.7. Assumptions and Dependencies

- JavaScript and local-storage enabled in browser
- Target browsers:
  - Chrome 135+
  - Firefox 137+
  - Safari 17.x+
  - Edge 135+
- External libraries may be adopted later (TBD)

3. External Interface Requirements

3.1. User Interfaces

- **Home Screen:**
  - The Home Screen shall display the game title.
  - The Home Screen shall provide a primary control to initiate gameplay.
  - The Home Screen shall provide controls to access the tutorial and leaderboard.
- **Tutorial:**
  - The system shall provide a tutorial that explains game mechanics, scoring, and controls to the player.
- **Leaderboard:**
  - The system shall display a leaderboard listing high scores (e.g., top 100). If all scores cannot be viewed simultaneously, the list must be scrollable.
  - The player's personal highest score and rank shall be clearly identifiable on the leaderboard.
- **Start/Weapon Selection:**
  - Upon initiating a new game, the system shall present an interface allowing the player to select their desired weapon/mallet from available options.
- **Game Field:**
  - The game field shall present a grid of interactive locations from which characters appear.
- **Dynamic Timer and Score Display:**
  - During gameplay, the interface shall continuously display the remaining game time and the player's current score.
- **Pause/Resume and Exit Buttons:**
  - The game interface shall provide controls for the player to pause and resume gameplay, and to exit or end the current game session.
- **Mute Button:**
  - The game interface shall include a control to mute and unmute game audio.
- **Game Over Screen:**
  - Upon game completion, an interface (e.g., "Game Over" screen) shall display the player's score for the completed session and their personal best score.
  - The "Game Over" interface shall provide controls for the player to return to the Home Screen and to start a new game.

Sketches and mock-ups will be supplied separately. For more detailed preliminary UI design concepts and layout ideas, please refer to Appendix C.

3.2. Hardware Interfaces

- Mouse / track-pad
- Touchscreen

3.3. Software Interfaces

- HTML5, CSS3, JavaScript libraries
- Browser Local Storage API

3.4. Communication Interfaces

None (client-side only).

4. System Features

4.1. Gameplay and Scoring Mechanics

4.1.1. Description

A fast-paced game in which professors randomly pop up from any of the 16 holes. Players click them to earn points; an on-screen score updates immediately. Top scores persist locally.

4.1.2. Stimulus/Response Sequences

1. Door opens; professor character appears.
2. Player clicks / taps character.
3. Game increments score.
4. Successful hit: +10 points.
5. Miss or inactivity: –5 points.
6. Trustee character (appearing approximately once per game) triggers a brief explosion animation ($\approx$ 1 s) and awards +20 points.

4.1.3. Functional Requirements

- **REQ-1.1**: Characters appear at uniformly random intervals of 0.5–1.5 s.
- **REQ-1.2**: Trustee character must appear randomly with approximately 5-10% probability. Trustee must trigger a special explosion animation visibly overlaying the screen for $\approx$ 1 s and play an accompanying scream sound effect.
- **REQ-1.3**: Characters vanish after 2 s if not clicked.
- **REQ-1.4**: Player's cursor must be visually replaced by a selected mallet graphic which appears to "whack" when clicked.
- **REQ-2.1**: Score updates in real-time and after each interaction.
- **REQ-2.2**: Top scores are stored via Local Storage.
- **REQ-2.3**: Sound effect plays on character clicks, misses, and trustee hits. Specific sound to be determined during implementation.

4.2. Weapon Selection Interface

4.2.1. Description

Before gameplay begins, players are presented with a weapon selection interface where they can choose their preferred mallet style. This interface enhances personalization and engagement, allowing players to select the whacking implement that best suits their style.

4.2.2. Stimulus/Response Sequences

1. Player clicks "Play" on the main menu.
2. Weapon selection screen appears with visual representation of available mallets.
3. Player selects desired mallet.
4. Game begins with the player's cursor replaced by their selected mallet.

4.2.3. Functional Requirements

- **REQ-WS-1**: The game must offer at least two visually distinct mallet options.
- **REQ-WS-2**: Each mallet must have both a selection image and a corresponding cursor representation.
- **REQ-WS-3**: The selected mallet must visually transform (scale down briefly) when clicked to simulate impact.
- **REQ-WS-4**: Player's mallet selection must persist for the entire gameplay session.
- **REQ-WS-5**: Weapon selection interface must include preview images that clearly represent how each mallet will appear during gameplay.

4.3. Audio and Sound Effects

4.3.1. Description

The game implements a comprehensive sound system to provide audio feedback for game events and enhance the user experience. All sounds follow a consistent style that matches the game's lighthearted theme.

4.3.2. Stimulus/Response Sequences

1. Game start: Plays introductory sound.
2. Professor hit: Plays "hit" sound.
3. Miss: Plays "miss" sound.
4. Trustee hit: Plays unique "explosion" sound with scream effect.
5. Game over: Plays concluding sound.
6. New high score: Plays celebratory sound.

4.3.3. Functional Requirements

- **REQ-3.1**: Game must provide audio feedback for all major user interactions and game events.
- **REQ-3.2**: Distinct sounds must play for:
  - Game start
  - Successful professor hits
  - Missed attempts

  - Trustee character hits (unique explosion sound)
  - Game over
  - Achievement of new high score

- **REQ-3.3**: Sound effects must synchronize with their corresponding visual events with latency ≤ 50 ms.
- **REQ-3.4**: Game must include a mute/unmute toggle button that persists user preference across sessions via Local Storage.
- **REQ-3.5**: Volume level must be consistent across all sound effects to prevent unexpected loud sounds.
- **REQ-3.6**: Sound format must be MP3 with WAV fallback for maximum browser compatibility.
- **REQ-3.7**: Individual sound effect files must not exceed 100 KB to ensure quick loading times.
- **REQ-3.8**: Game must remain fully playable with audio disabled for accessibility.
- **REQ-3.9**: In addition to mute/unmute, the game should provide volume adjustment with settings persisted in Local Storage.
- **REQ-3.10**: Audio system must efficiently pre-load and cache sound effects to prevent performance degradation.
- **REQ-3.11**: Game must gracefully handle scenarios where audio playback is not supported or permission is denied.

5. Non-functional Requirements

5.1. Performance

- Initial page load $\leq$ 5 s (on broadband).
- Animation renders at 60 fps on supported hardware.
- Audio playback must begin within 50 ms of triggering events.

5.2. Security

No sensitive data processed. All data remain local to the browser.

5.3. Software Quality Attributes

- Readable, maintainable codebase
- Robust gameplay with graceful error handling

5.4. Error Handling

- Detect and report Local Storage quota issues.
- Provide clear feedback for unsupported browsers.

A. Glossary

Professor Standard clickable target.

Trustee Special character with higher point value (20 points) that appears less frequently and triggers a unique explosion animation with scream effect.

Mallet The player's cursor, visually represented as a whacking implement. Players can choose between different mallet styles before starting the game.

FPS Frames per second.

Local Storage Browser-side key-value store.

B. To Be Determined

- Final UI mock-ups and design specifics
- Final JavaScript library selection
- Precise animation specification for trustee effect

C. Preliminary UI Design Concepts and Layout Notes

This appendix contains preliminary design concepts, layout ideas, and specific visual notes for the user interface of *Whack-a-Prof*. These descriptions are intended to supplement the formal UI requirements in Section 3.1 and serve as a starting point for detailed UI design and mock-up creation. They are not formal requirements themselves but represent current thinking and suggestions.

Home Screen Design Concepts

- The game title, "Whack A Prof!", could be prominently displayed, perhaps at the top center of the screen.
- For primary navigation, a "Start" button might be positioned in the center of the screen.
- "Tutorial" and "Leaderboard" access buttons could be grouped logically, for instance, below the "Start" button.
- Thematic imagery, such as depictions of the professor characters, could be considered for the sides of the screen to enhance visual appeal.

Tutorial Presentation Ideas

- The tutorial explaining game mechanics could be presented using either video or a sequence of static images. The final choice would depend on what the graphics/design resources determine to be most effective.

Leaderboard Display Concepts

- The leaderboard could display a list of the top 100 scores.
- The player's own highest score might be highlighted or positioned at the very top of the displayed list for easy visibility, along with their rank.

Weapon Selection Interface Concepts

- When the player initiates a new game, an interface could display a variety of mallet weapon visuals in the center of the screen for selection. The specific designs of these mallets are to be determined by the graphics/design team.

Game Field Layout Concept

- A potential layout for the game field involves a 4x4 grid of 16 holes, arranged to spread across the available screen width.
- During gameplay, professor characters would then appear randomly from these holes.

In-Game Displays Layout Ideas

- The dynamic game timer could be positioned at the top-left of the screen.
- The player's current score display could be placed near the timer, for example, slightly to its right.

In-Game Controls Layout Ideas

- A Pause/Resume icon button might be located at the top-right of the screen.
- An Exit/End game icon button could be positioned adjacent to the Pause/Resume button (e.g., slightly to its right).
- A toggle Mute button could be placed near these controls, for instance, slightly to the left of the Pause/Resume button.

Game Over Screen Concepts

- Upon game completion, a distinct visual element, such as a square box, could appear in the center of the screen.
- This element would display both the player's score for the just-completed session and their overall best score.
- Below the score display, "Home" and "Play Again" buttons would allow the player to navigate accordingly.